FIAP GRADUAÇÃO

FIAP

Tecnologia em Análise e Desenvolvimento de Sistemas

Application Development For Databases

PROF. MILTON

Versão 1.4 – <fevereiro de 24>

FIAP

Criando Pacotes

Objetivos

Ao concluir esta lição, você será capaz de:

- Descrever pacotes e listar seus componentes
- Criar um pacote para agrupar variáveis, cursores, constantes, exceções, funções e procedures relacionados
- Designar uma estrutura de pacote como pública ou privada
- Chamar uma estrutura de pacote
- Descrever o uso de um pacote sem corpo do pacote

4

Objetivo da Lição

Nesta lição, você aprenderá o que é um pacote e quais são seus componentes. Você também aprenderá a criar e usar pacotes.

Agenda de Lições

- Identificando os benefícios e os componentes de pacotes
- Trabalhando com pacotes:
  - Criando a especificação e o corpo do pacote
  - Chamando os subprogramas de pacotes
  - Removendo um pacote
  - Exibindo as informações dos pacotes

O Que São Pacotes PL/SQL?

FIAP

- Um pacote é um objeto de esquema que agrupa logicamente tipos, variáveis e subprogramas PL/SQL relacionados.
- Geralmente, os pacotes têm duas partes:
  - Uma especificação (spec)
  - Um corpo
- A especificação é a interface com o pacote. Ela declara os tipos, variáveis, constantes, exceções, cursores e subprogramas que podem ser referenciados a partir de fora do pacote.
- O corpo define as consultas para os cursores e o código para os subprogramas.
- Permitem que o servidor Oracle leia vários objetos na memória de uma só vez.

6

Pacotes PL/SQL: Visão Geral

Os pacotes PL/SQL permitem agrupar tipos PL/SQL, variáveis, estruturas de dados, exceções e subprogramas relacionados em apenas um container. Por exemplo, um pacote de recursos humanos pode conter procedures de contratação e demissão, funções de comissão e bônus, e variáveis de isenção de impostos.

Em geral, um pacote consiste em duas partes armazenadas separadamente no banco de dados:

- Uma especificação
- Um corpo (opcional)

O pacote propriamente dito não pode ser chamado, parametrizado nem aninhado. Uma vez criado e compilado o pacote, o conteúdo poderá ser compartilhado com várias aplicações.

Quando uma estrutura de pacote PL/SQL é referenciada pela primeira vez, o pacote todo é carregado na memória. O acesso subsequente a estruturas do mesmo package não requer entrada/saída (E/S) de disco.

Vantagens do Uso de Pacotes

- Modularidade: Encapsulamento de estruturas relacionadas
- Manutenção mais fácil: Agrupamento de funcionalidades relacionadas logicamente
- Projeto de aplicação mais fácil: Codificação e compilação da especificação e do corpo separadas
- Ocultação de informações:
  - Apenas as declarações da especificação do pacote ficam visíveis e acessíveis para as aplicações
  - As estruturas privadas do corpo do pacote ficam ocultas e inacessíveis
  - Toda a codificação fica oculta no corpo do pacote

7

Vantagens do Uso de Pacotes

Os pacotes são uma alternativa à criação de procedures e funções como objetos de esquema stand-alone, oferecendo diversas vantagens.

Modularidade e facilidade de manutenção: Você encapsula estruturas de programação relacionadas logicamente em um módulo nomeado. Cada pacote é de fácil compreensão, e a interface entre os pacotes é simples, clara e bem definida.

Projeto de aplicação mais fácil: Inicialmente, você só precisa das informações da interface na especificação do pacote. Você pode codificar e compilar uma especificação sem o seu corpo correspondente. Em seguida, os subprogramas armazenados que fazem referência ao pacote também podem ser compilados. Você só precisará definir o corpo do pacote totalmente quando estiver pronto para concluir a aplicação.

Ocultação de informações: Você decide quais estruturas são públicas (visíveis e acessíveis) e quais são privadas (ocultas e inacessíveis). As declarações da especificação do pacote ficam visíveis e acessíveis para as aplicações. O corpo do pacote oculta a definição das estruturas privadas, de modo que, se a definição for alterada, apenas o pacote será afetado (e não a aplicação ou os programas que fizeram a chamada). Assim, é possível alterar a implementação sem ter de recompilar os programas que fizeram a chamada. Além disso, ao ocultar dos usuários os detalhes da implementação, você protege a integridade do pacote.

Vantagens do Uso de Pacotes

- Funcionalidade adicional: Persistência de variáveis públicas e cursores
- Melhor desempenho:
  - O pacote inteiro é carregado na memória quando é referenciado pela primeira vez.
  - Há apenas uma cópia na memória para todos os usuários.
  - A hierarquia de dependências é simplificada.
- Overloading: Vários subprogramas com o mesmo nome

8

Vantagens do Uso de Pacotes (continuação)

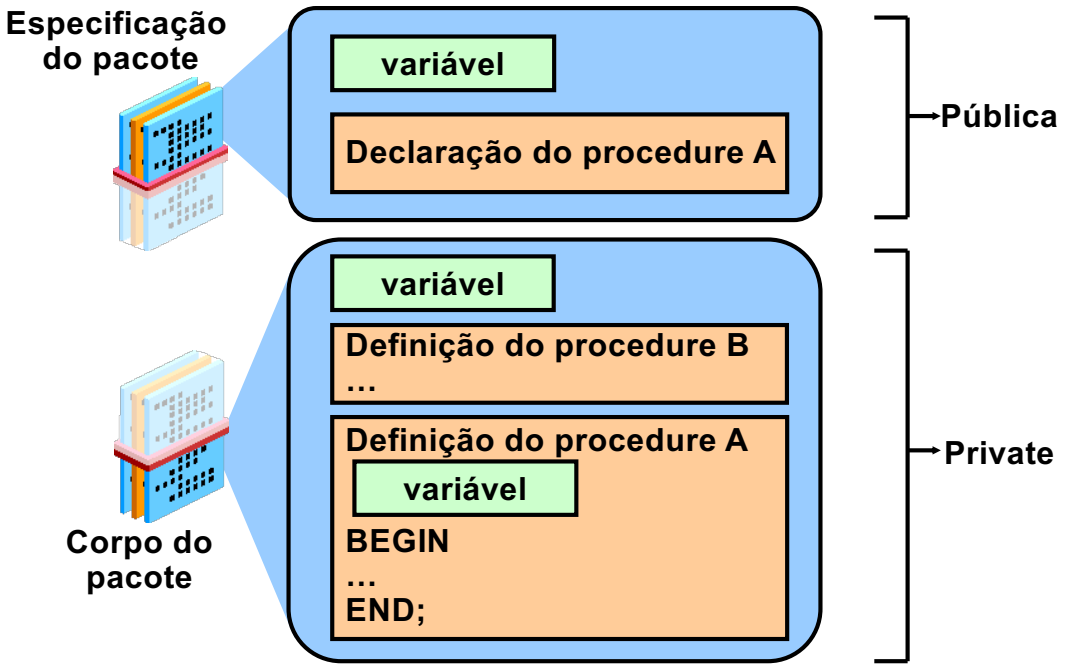
Funcionalidade adicional: As variáveis e os cursores públicos encapsulados persistem durante toda a sessão. Assim, eles podem ser compartilhados por todos os subprogramas em execução no ambiente. Eles também permitem manter dados entre transações sem a necessidade de armazená-los no banco de dados. As estruturas privadas também persistem durante a sessão, mas só podem ser acessadas dentro do pacote.

Melhor desempenho: Quando você chama um subprograma encapsulado pela primeira vez, o pacote inteiro é carregado na memória. Dessa forma, as chamadas posteriores para subprogramas relacionados do pacote não exigem mais operações de entrada/saída no disco. Os subprogramas encapsulados também interrompem dependências em cascata e, portanto, evitam compilações desnecessárias.

Overloading: Com os pacotes, você pode fazer overloading de procedures e funções, ou seja, pode criar vários subprogramas com o mesmo nome e no mesmo pacote, cada um deles utilizando diferentes parâmetros que têm quantidades e tipos de dados específicos.

Observação: As dependências são abordadas de maneira mais detalhada na lição “Gerenciando Dependências”.

Componentes de um Pacote PL/SQL



Componentes de um Pacote PL/SQL

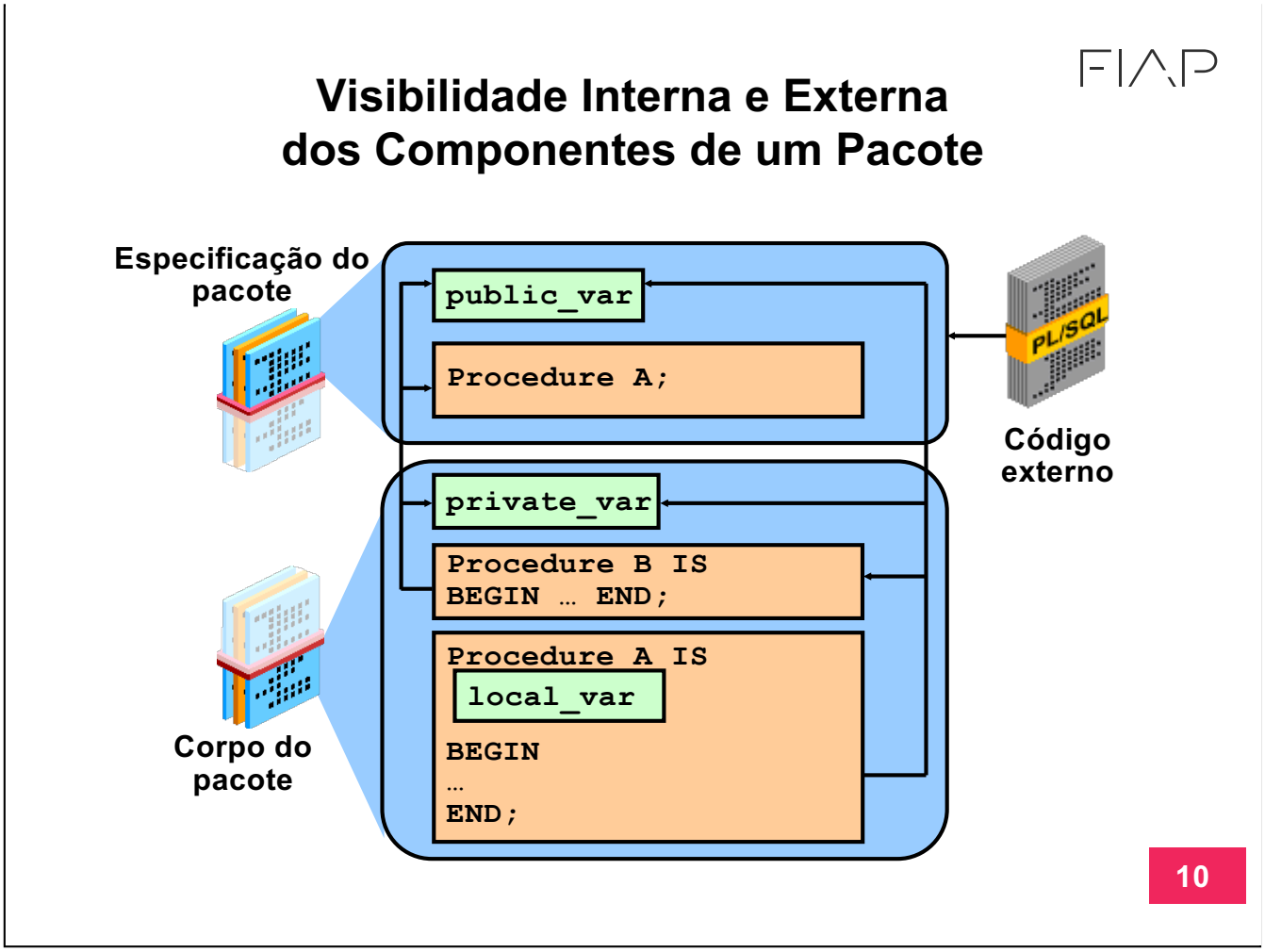
Os pacotes são criados com duas partes:

- A **especificação do** pacote é a interface com as aplicações. Ela declara as variáveis, as constantes, as exceções, os cursores, os subprogramas e os tipos públicos disponíveis para uso. A especificação do pacote também pode incluir `PRAGMA`s, que são diretivas para o compilador.
- O **corpo do pacote** define seus próprios subprogramas e deve implementar totalmente os subprogramas declarados na parte da especificação. O corpo do pacote também pode definir estruturas PL/SQL, como tipos, variáveis constantes, exceções e cursores.

Os componentes públicos são declarados na especificação do pacote. A especificação define uma API (application programming interface, interface de programação de aplicação) pública para usuários de recursos e funcionalidade do pacote, ou seja, os componentes públicos podem ser referenciados de qualquer ambiente de servidor Oracle externo ao pacote.

Os componentes privados são colocados no corpo do pacote e só podem ser referenciados por outras estruturas que estejam no mesmo corpo do pacote. Os componentes privados podem fazer referência aos componentes públicos de um pacote.

Observação: Se a especificação do pacote não contiver declarações de subprograma, não será necessário ter um corpo de pacote.



Visibilidade Interna e Externa dos Componentes de um Pacote

A visibilidade de um componente indica se ele pode ser visto, ou seja, referenciado e usado por outros componentes ou objetos. A visibilidade dos componentes depende da forma como eles estão *declarados, ou seja, local* ou *globalmente*

Os componentes locais são visíveis dentro da estrutura em que foram declarados, como indicado a seguir:

- As variáveis definidas em um subprograma podem ser referenciadas nesse subprograma, mas não são visíveis para os componentes externos. Por exemplo, a variável `local_var` pode ser usada no procedure A.
- As variáveis de pacote públicas, que são declaradas em um corpo de pacote, podem ser referenciadas por outros componentes do mesmo corpo de pacote. Elas não são visíveis para subprogramas ou objetos que estão fora do pacote. Por exemplo, `private_var`pode ser usada pelos procedures A e B no corpo do pacote, mas não fora do pacote.

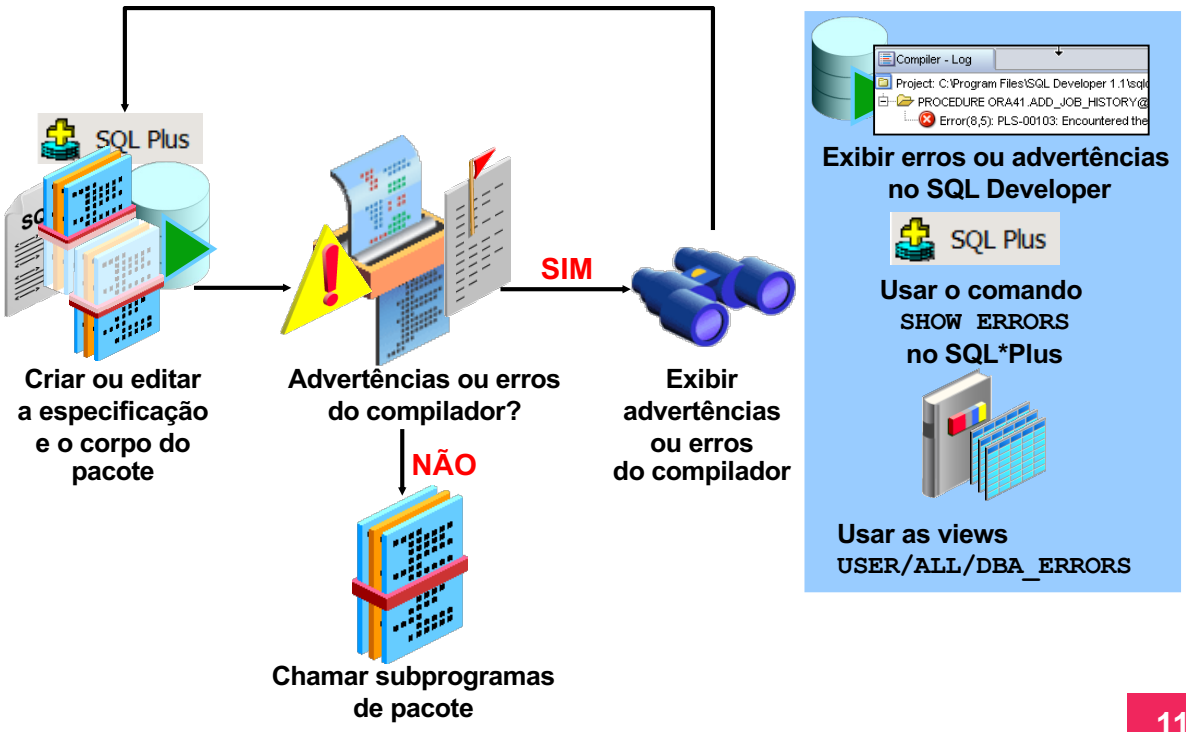
Os componentes declarados de forma global são visíveis dentro e fora do pacote. Por exemplo:

Uma variável pública, que é declarada em uma especificação de pacote, pode ser referenciada e alterada fora do pacote (por exemplo, a variável `public_var` pode ser referenciada externamente).

- Um subprograma de pacote contido na especificação pode ser chamado a partir de códigos-fontes externos (por exemplo, o procedure A pode ser chamado a partir de um ambiente externo ao pacote).

Observação: Os subprogramas privados, como o procedure B, só podem ser chamados com subprogramas públicos, como o procedure A, ou outras estruturas de pacote privadas. Uma variável pública declarada na especificação do pacote é uma variável global.

Desenvolvendo Pacotes PL/SQL: Visão Geral



11

Desenvolvendo Pacotes PL/SQL

O diagrama no slide ilustra as etapas básicas envolvidas no desenvolvimento e uso de pacotes:

1. Crie o procedure usando a árvore Object Navigation do SQL Developer ou a área SQL Worksheet.
2. Compile o pacote. O pacote é criado no banco de dados. A instrução `CREATE PACKAGE` cria e armazena no banco de dados o código-fonte e o *m-code* compilado. Para compilar o pacote, clique com o botão direito do mouse no nome do pacote na árvore Object Navigator e depois clique em Compile.
3. Se não houver advertências ou erros de compilação, execute qualquer estrutura pública na especificação do pacote em um ambiente de servidor Oracle Server.
4. Se houver advertências ou erros de compilação, será possível exibir (e corrigir) as advertências ou erros usando um dos seguintes métodos:
   - Usando a interface do SQL Developer (tab Compiler – Log)
   - Usando o comando `SHOW ERRORS` no SQL*Plus
   - Usando as views `USER/ALL/DBA_ERRORS`

Agenda de Lições

- Identificando os benefícios e os componentes de pacotes
- Trabalhando com pacotes:
  - Criando a especificação e o corpo do pacote
  - Chamando os subprogramas de pacotes
  - Removendo um pacote
  - Exibindo as informações dos pacotes

Criando a Especificação de Pacote: Usando a Instrução CREATE PACKAGE

```
CREATE [OR REPLACE] PACKAGE package_name IS|AS
    public type and variable declarations
    subprogram specifications
END [package_name];
```

- A opção OR REPLACE elimina e recria a especificação do pacote.
- Por default, as variáveis declaradas na especificação do pacote são inicializadas como NULL.
- Todas as estruturas declaradas em uma especificação de pacote são visíveis para os usuários que têm privilégios no pacote.

13

Criando a Especificação do Pacote

Para criar pacotes, declare todas as estruturas públicas dentro da especificação do pacote.

- Especifique a opção OR REPLACE se estiver sobregravando uma especificação de pacote existente.
- Inicialize uma variável com uma fórmula ou um valor constante na declaração, se necessário; caso contrário, a variável será inicializada implicitamente como NULL.

Veja a seguir as definições dos itens da sintaxe do pacote:

- **package_name** especifica um nome para o pacote, que deve ser exclusivo entre os objetos do esquema do proprietário. A inclusão do nome do pacote após a palavra-chave END é opcional.
- **public type and variable declarations** declara variáveis, constantes, cursores, exceções, tipos definidos pelo usuário e subtipos públicos.
- **subprogram specification** especifica as declarações de funções ou de procedures públicos.

A especificação do pacote deve conter os cabeçalhos de funções e de procedures encerrados com um ponto-e-vírgula, sem a palavra-chave IS (ou AS) e o bloco PL/SQL correspondente. A implementação de uma função ou de um procedure declarado em uma especificação de pacote é feita no corpo do pacote.

O banco de dados Oracle armazena a especificação e o corpo do pacote separadamente. Assim, é possível alterar a implementação de uma estrutura de programa no corpo do pacote sem invalidar outros objetos de esquema que chamam ou fazem referência a essa estrutura.

Criando a Especificação do Pacote: Usando o SQL Developer

Você pode usar o SQL Developer para criar a especificação de pacote como se segue:

1. Clique com o botão direito do mouse no nó **Packages** na árvore de navegação Connections.
2. Selecione **New Package** no menu de atalho.
3. Na janela **Create PL/SQL Package** , selecione o nome do esquema, insira o nome do novo pacote e depois clique em OK. Uma tab para o novo pacote é exibida juntamente com o shell do novo pacote.
4. Informe o código do novo pacote.
5. Compile ou salve (usando o ícone Save na barra de ferramentas principal) o novo pacote.
6. A tab **Messages – Log** exibe se a compilação foi ou não bem-sucedida.
7. O pacote recém-criado é exibido no nó **Packages** na árvore de navegação Connections.

Criando o Corpo do Pacote: Usando o SQL Developer

Criando o Corpo do Pacote: Usando o SQL Developer

Você pode usar o SQL Developer para criar o corpo do pacote como se segue:

1. Clique com o botão direito do mouse no nome do pacote para o qual você está criando um corpo no nó **Packages** na árvore de navegação Connections.
2. Selecione **Create Body** no menu de atalho. Uma tab para o novo pacote é exibida juntamente com o shell do novo corpo de pacote.
3. Informe o código do novo corpo de pacote.
4. Compile ou salve o corpo do pacote.
5. Uma tab **Messages – Log** exibe se a compilação foi ou não bem-sucedida.
6. O corpo do pacote recém-criado é exibido no nó **Packages** na árvore de navegação Connections.

Exemplo de Especificação de Pacote: comm_pkg

```
-- The package spec with a public variable and a
-- public procedure that are accessible from
-- outside the package.

CREATE OR REPLACE PACKAGE comm_pkg IS
  v_std_comm NUMBER := 0.10;  --initialized to 0.10
  PROCEDURE reset_comm(new_comm NUMBER);
END comm_pkg;
/
```

- V_STD_COMM é uma variável *pública* global inicializada como 0.10.
- RESET_COMM é um procedure *público* usado para redefinir a comissão padrão com base em algumas regras de negócios. Esse procedure é implementado no corpo do pacote.

16

Exemplo de Especificação de Pacote: comm_pkg

O exemplo do slide cria um pacote denominado comm_pkg que gerencia regras de processamento de negócios para cálculos de comissão.

The A variável pública (global)v_std_comm é declarada para armazenar o percentual máximo de comissão permitido para a sessão do usuário e é inicializada como 0.10 (isto é, 10%).

O procedure público reset_comm é declarado para aceitar um novo percentual de comissão que atualizará o percentual padrão se as regras de validação da comissão forem aceitas. As regras de validação para redefinir a comissão não são definidas como públicas nem aparecem na especificação do pacote. Essas regras de validação são gerenciadas por meio de uma função privada do corpo do pacote.

Criando o Corpo do Pacote

```
CREATE [OR REPLACE] PACKAGE BODY package_name IS|AS
    private type and variable declarations
    subprogram bodies
[BEGIN initialization statements]
END [package_name];
```

- A opção OR REPLACE elimina e recria o corpo do pacote.
- Os identificadores definidos no corpo do pacote são *privados* e não ficam visíveis fora dele.
- Todas as estruturas *privadas* devem ser declaradas antes de serem referenciadas.
- As estruturas públicas são visíveis para o corpo do pacote.

17

Criando o Corpo do Pacote

Crie um corpo de pacote para definir e implementar todos os subprogramas públicos e estruturas privadas de suporte. Ao criar um corpo de pacote, faça o seguinte:

- Especifique a opção OR REPLACE para sobregravar um corpo de pacote existente.
- Defina os subprogramas na ordem adequada. O princípio básico é que você deve declarar uma variável ou um subprograma para que estes possam ser referenciados por outros componentes do mesmo corpo de pacote. É comum encontrar todas as variáveis e os subprogramas privados definidos primeiro e os subprogramas públicos definidos por último no corpo do pacote.
- Conclua a implementação de todos os procedures ou funções declaradas na especificação do pacote dentro do corpo do pacote.

Veja a seguir as definições dos itens da sintaxe do corpo do pacote:

- **package_name** especifica um nome para o pacote, que deve ser igual à especificação do pacote. O uso do nome do pacote após a palavra-chave END é opcional.
- **private type and variable declarations** declara variáveis, constantes, cursores, exceções, tipos definidos pelo usuário e subtipos privados.
- **subprogram specification** especifica a implementação completa de todos os procedures ou funções privados e/ou públicos.
- **[BEGIN initialization statements]** é um bloco opcional de código de inicialização que é executado quando o pacote é referenciado pela primeira vez.

Exemplo de Corpo de Pacote: comm_pkg

```
CREATE OR REPLACE PACKAGE BODY comm_pkg IS
  FUNCTION validate(p_comm NUMBER) RETURN BOOLEAN IS
    v_max_comm    employees.commission_pct%type;
  BEGIN
    SELECT MAX(commission_pct) INTO v_max_comm
    FROM employees;
    RETURN (p_comm BETWEEN 0.0 AND v_max_comm);
  END validate;

  PROCEDURE reset_comm (p_new_comm NUMBER) IS
  BEGIN
    IF validate(p_new_comm) THEN
      v_std_comm := p_new_comm; -- reset public var
    ELSE  RAISE_APPLICATION_ERROR(
            -20210, 'Bad Commission');
    END IF;
  END reset_comm;
END comm_pkg;
```

18

Exemplo de Corpo de Pacote: comm_pkg

O slide mostra o corpo de pacote completo de `comm_pkg`, com uma função privada denominada `validate` para verificar se uma comissão é válida. A validação exige que a comissão seja positiva e menor que a comissão mais alta entre os funcionários existentes. O procedure `reset_comm` chama a função de validação privada antes de alterar a comissão padrão em `v_std_comm`. No exemplo, observe o seguinte:

- A variável `v_std_comm` referenciada no procedure `reset_comm` é pública. As variáveis declaradas na especificação do pacote, como `v_std_comm`, podem ser referenciadas diretamente sem qualificação.
- O procedure `reset_comm` implementa a definição pública na especificação.
- No corpo de pacote `comm_pkg`, a função `validate` é privada e referenciada diretamente a partir do procedure `reset_comm` sem qualificação.

Observação: A função `validate` aparece antes do procedure `reset_comm` porque `reset_comm` faz referência à função `validate`. Será possível criar declarações forward para subprogramas no corpo do pacote, se for necessário alterar sua ordem de exibição. Se uma especificação de pacote declarar apenas tipos, constantes, variáveis e exceções, sem nenhuma especificação de subprograma, o corpo do pacote será desnecessário. No entanto, o corpo do pacote poderá ser usado para inicializar os itens declarados na especificação do pacote.

Chamando os Subprogramas de Pacotes: Exemplos

```
-- Invoke a function within the same packages:
CREATE OR REPLACE PACKAGE BODY comm_pkg IS ...
  PROCEDURE reset_comm (p_new_comm NUMBER) IS
  BEGIN
    IF validate(p_new_comm) THEN
      v_std_comm := p_new_comm;
    ELSE ...
    END IF;
  END reset_comm;
END comm_pkg;
```

```
-- Invoke a package procedure from SQL*Plus:
EXECUTE comm_pkg.reset_comm(0.15)
```

```
-- Invoke a package procedure in a different schema:
EXECUTE scott.comm_pkg.reset_comm(0.15)
```

19

Chamando Subprogramas de Pacotes

Após o armazenamento do pacote no banco de dados, você poderá chamar subprogramas públicos ou privados que estejam no mesmo pacote, ou subprogramas públicos externos ao pacote. Qualifique totalmente o subprograma com o nome do pacote correspondente, quando ele for chamado fora do pacote. Use a sintaxe `package_name.subprogram`.

A qualificação total de um subprograma chamado dentro do mesmo pacote é opcional.

Exemplo 1: Chame a função `validate` a partir do procedure `reset_comm` dentro do mesmo pacote. O prefixo do nome do pacote é opcional.

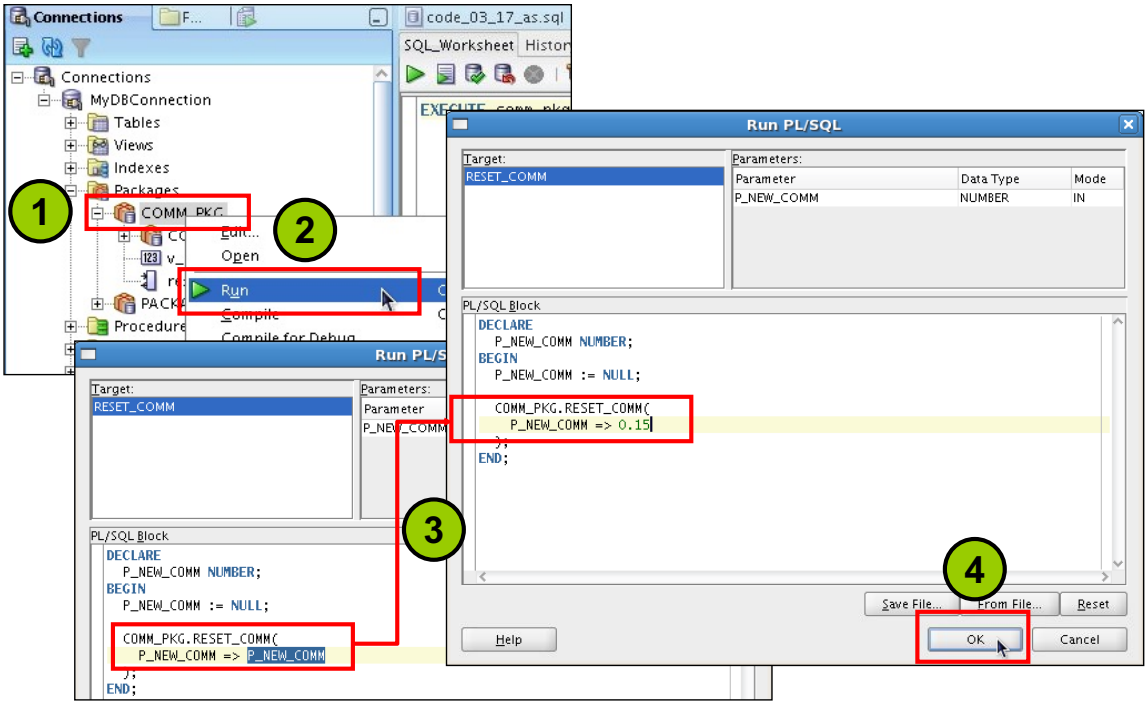
Exemplo 2: Chama o procedure `reset_comm` a partir do SQL*Plus (um ambiente externo ao pacote) a fim de redefinir a comissão atual como 0,15 para a sessão do usuário.

Exemplo 3: Chama o procedure `reset_comm` pertencente a um usuário do esquema denominado `SCOTT`. Usando o SQL*Plus, o procedure de pacote qualificado é prefixado com o nome do esquema. Para simplificar, você pode usar um sinônimo que faça referência a `schema.package_name`.

Vamos supor que um link de banco de dados denominado `NY` tenha sido criado para um banco de dados remoto, no qual o procedure de pacote `reset_comm` foi criado. Para chamar o procedure remoto, use:

```
EXECUTE comm_pkg.reset_comm@NY(0.15)
```

Chamando os Subprogramas de Pacotes: Usando o SQL Developer



Chamando os Subprogramas de Pacotes: Usando o SQL Developer

Você pode usar o SQL Developer para chamar o subprograma de um pacote como se segue:

1. Clique com o botão direito do mouse no nome do pacote no nó Packages na árvore Navigation.
2. Selecione **Run** no menu de flutuação. A janela **Run PL/SQL** é exibida. Você pode usar a janela **Run PL/SQL** para especificar os valores de parâmetro para executar uma função ou procedure PL/SQL. (Se você especificar um pacote, selecione uma função ou procedure no pacote.) Especifique o seguinte:
   a. **Target:** Selecione o nome da função ou procedure a executar.
   b. **Parameters:** Esta seção lista cada parâmetro para o alvo especificado. O modo de cada parâmetro pode ser `IN` (o valor é especificado), `OUT` (o valor é retornado), ou `IN/OUT` (o valor é especificado, e o resultado da ação da função ou procedure é armazenado no parâmetro).
3. Na seção **PL/SQL Block** , altere as especificações do parâmetro formal `IN` e `IN/OUT` neste bloco para valores reais que você deseja usar para executar a função ou procedure. Por exemplo, para especificar `0.15` como o valor para um parâmetro de entrada denominado `P_NEW_COMM`, altere `P_NEW_COMM => P_NEW_COMM` para `P_NEW_COMM => 0.15`.
4. Clique em **OK**. O SQL Developer executa a função ou procedure.

Criando e Usando Pacotes sem Corpo do Pacote

```
CREATE OR REPLACE PACKAGE global_consts IS
  c_mile_2_kilo    CONSTANT  NUMBER  :=  1.6093;
  c_kilo_2_mile    CONSTANT  NUMBER  :=  0.6214;
  c_yard_2_meter   CONSTANT  NUMBER  :=  0.9144;
  c_meter_2_yard   CONSTANT  NUMBER  :=  1.0936;
END global_consts;
```

```
SET SERVEROUTPUT ON
BEGIN
   DBMS_OUTPUT.PUT_LINE('20 miles = ' ||
        20 * global_consts.c_mile_2_kilo || ' km');
END;
```

```
SET SERVEROUTPUT ON
CREATE FUNCTION mtr2yrd(p_m NUMBER) RETURN NUMBER IS
BEGIN
  RETURN (p_m * global_consts.c_meter_2_yard);
END mtr2yrd;
/
EXECUTE DBMS_OUTPUT.PUT_LINE(mtr2yrd(1))
```

21

Criando e Usando Pacotes sem Corpo do Pacote

As variáveis e constantes declaradas em subprogramas stand-alone existem apenas durante a execução do subprograma. Para manter os dados durante a sessão do usuário, crie uma especificação de pacote contendo declarações de constantes e variáveis públicas (globais). Nesse caso, crie uma especificação de pacote sem corpo de pacote, conhecida como *pacote sem corpo do pacote*. Como vimos anteriormente nesta lição, se uma especificação declarar apenas tipos, constantes, variáveis e exceções, o corpo do pacote será desnecessário.

Exemplos

A primeira caixa de código do slide cria uma especificação de pacote sem corpo do pacote com várias constantes a serem usadas para as taxas de conversão. Essa especificação de pacote não exige um corpo do pacote. Supõe-se que a instrução `SET SERVEROUTPUT ON` foi emitida antes da execução dos exemplos de código no slide.

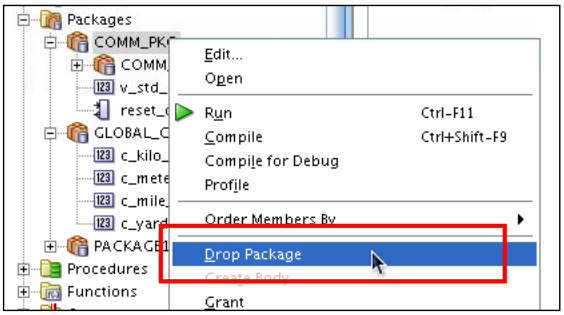
A segunda caixa de código faz referência à constante `mile_2_kilo` do pacote `global_consts`, prefixando o identificador da constante com o nome do pacote.

O terceiro exemplo cria uma função stand-alone `c_mtr2yrd` para converter metros em jardas e usa a taxa de conversão de constante `c_meter_2_yard` declarada no pacote `global_consts`. A função é chamada no parâmetro `DBMS_OUTPUT.PUT_LINE`.

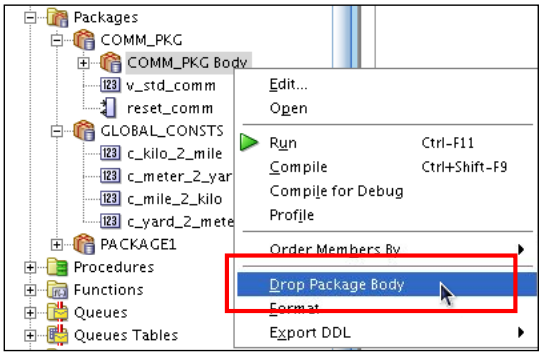
Regra a ser seguida: Ao fazer referência a variáveis, cursores, constantes ou exceções fora do pacote, você deve qualificá-los com o nome do pacote.

Removendo Pacotes: Usando o SQL Developer ou a Instrução do SQL DROP

Eliminar especificação e corpo do pacote



Eliminar somente corpo do pacote




```
-- Remove the package specification and body
DROP PACKAGE package_name;
```

```
-- Remove the package body only
DROP PACKAGE BODY package_name;
```

22

Removendo Pacotes

Quando um pacote não for mais necessário, você poderá usar uma instrução SQL no SQL Developer para removê-lo. Um pacote tem duas partes; portanto, você pode remover o pacote inteiro ou remover apenas o corpo do pacote e manter a especificação do pacote.

Exibindo Pacotes Usando o Dicionário de Dados

```
-- View the package specification.
SELECT text
FROM   user_source
WHERE  name = 'COMM_PKG' AND type = 'PACKAGE'
ORDER BY LINE;
```

	TEXT
1	PACKAGE comm_pkg IS
2	std_comm NUMBER := 0.10; --initialized to 0.10
3	PROCEDURE reset_comm(new_comm NUMBER);
4	END comm_pkg;

```
-- View the package body.
SELECT text
FROM   user_source
WHERE  name = 'COMM_PKG' AND type = 'PACKAGE BODY'
ORDER BY LINE;
```

	TEXT
1	PACKAGE BODY comm_pkg IS
2	FUNCTION validate(comm NUMBER) RETURN BOOLEAN IS
3	max_comm employees.commission_pct%type;
4	BEGIN
5	SELECT MAX(commission_pct) INTO max_comm
6	FROM employees;
7	RETURN (comm BETWEEN 0.0 AND max_comm);

Exibindo Pacotes no Dicionário de Dados

O código-fonte dos pacotes PL/SQL também é armazenado nas views de dicionário de dadosUSER_SOURCE ALL_SOURCE. A tabela USER_SOURCE é usada para exibir o código PL/SQL pertencente a você. A tabela ALL_SOURCE é usada para exibir o código PL/SQL para o qual você recebeu o direito EXECUTE concedido pelo proprietário do código desse subprograma e fornece uma coluna OWNER além das colunas anteriores.

Ao consultar o pacote, use uma condição em que a coluna TYPE seja:

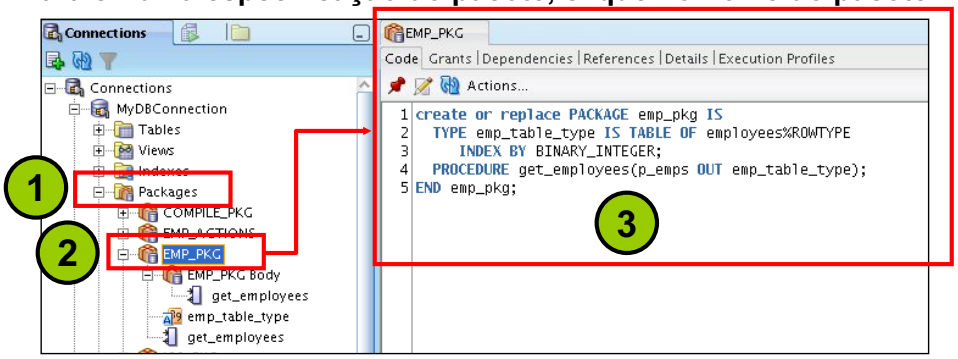
- Igual a ''PACKAGE' para exibir o código-fonte da especificação do pacote
- Igual a ''PACKAGE BODY' para exibir o código-fonte do corpo do pacote

Você também pode exibir a especificação do pacote e o corpo no SQL Developer usando o nome do pacote no nó Packages.

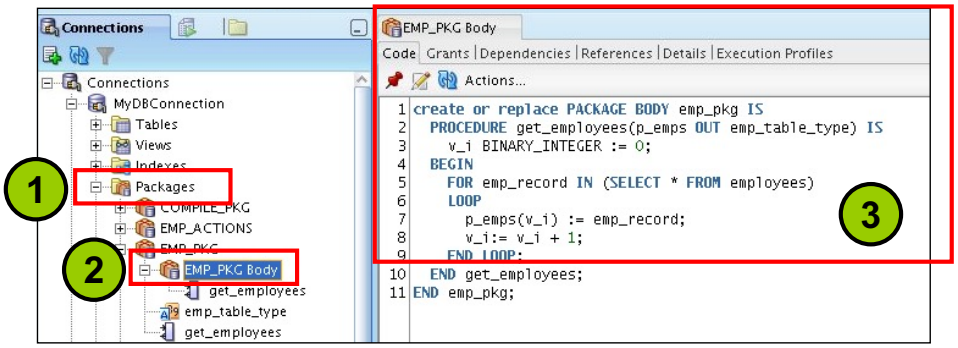
Observação:Não é possível exibir o código-fonte de pacotes PL/SQL Oracle incorporados, nem o código PL/SQL cujo código-fonte tenha sido encapsulado com o utilitário WRAPou ocultação. A ocultação e o encapsulamento do código-fonte PL/SQL são abordados em uma lição posterior. Clicar no ícone Execute Statement (F9) (em vez do ícone Run Script) na barra de ferramentas SQL Worksheet, às vezes exibe uma saída mais bem formatada na tab Results, conforme mostrado nos exemplos do slide.

Exibindo Pacotes Usando o SQL Developer

Para exibir a especificação do pacote, clique no nome do pacote



Para exibir o corpo do pacote, clique no corpo do pacote



24

Exibindo Pacotes Usando o SQL Developer

Para exibir a especificação de um pacote no SQL Developer, faça o seguinte:

1. Clique no nó **Packages** na tab **Connections** .
2. Clique no nome do pacote.
3. O código de especificação do pacote é exibido na tab **Code** conforme mostrado no slide.

Para exibir o corpo de um pacote no SQL Developer, faça o seguinte:

1. Clique no nó **Packages** na tab **Connections** .
2. Clique no corpo do pacote.
3. O código do corpo do pacote é exibido na tab **Code** conforme mostrado no slide.

Diretrizes para a Criação de Pacotes

- Crie pacotes para uso geral.
- Defina a especificação do pacote antes do corpo.
- Inclua na especificação do pacote apenas as estruturas que você deseja que sejam públicas.
- Coloque os itens na parte da declaração do corpo do pacote quando você precisar mantê-los durante uma sessão ou entre transações.
- O gerenciamento da dependência detalhada reduz a necessidade de recompilar os subprogramas de referenciamento quando uma especificação de pacote é alterada.
- Coloque na especificação do pacote o menor número possível de estruturas.

25

Diretrizes para a Criação de Pacotes

Mantenha os pacotes o mais genéricos que puder, de modo que possam ser reutilizados em aplicações futuras. Além disso, evite criar pacotes que dupliquem recursos fornecidos pelo servidor Oracle.

As especificações de pacotes refletem o projeto da aplicação; portanto, defina-as antes de definir os corpos do pacote. A especificação do pacote deve conter apenas as estruturas que precisam estar visíveis para os usuários do pacote. Assim, outros desenvolvedores não poderão utilizar o pacote de forma incorreta baseando o código em detalhes irrelevantes.

Coloque os itens na parte da declaração do corpo do pacote quando você precisar mantê-los durante uma sessão ou entre transações. Por exemplo, declare uma variável denominada `NUMBER_EMPLOYED` como privada, se for necessário manter cada chamada para um procedure que utilize essa variável. Quando a variável for declarada como global na especificação do pacote, o valor dessa variável global será inicializado em uma sessão na primeira vez que uma estrutura do pacote for chamada.

Antes do Oracle Database 11*g*, as alterações feitas no corpo do pacote não exigiam a recompilação das estruturas dependentes, enquanto as alterações feitas na especificação do pacote exigiam a recompilação de cada subprograma armazenado que fizesse referência ao pacote. O Oracle Database 11*g* reduz esta dependência. As dependências agora são controladas no nível do elemento dentro da unidade. O Gerenciamento de Dependência Detalhada é abordado em uma lição posterior.

Questionário

A especificação do pacote é a interface com as aplicações. Ela declara as variáveis, as constantes, as exceções, os cursores, os subprogramas e os tipos públicos disponíveis para uso. A especificação do pacote também pode incluir PRAGMAs, que são diretivas para o compilador.

1. Verdadeiro
2. Falso

26

Resposta: 1

Sumário

Nesta lição, você aprendeu a:

- Descrever pacotes e listar seus componentes
- Criar um pacote para agrupar variáveis, cursores, constantes, exceções, funções e procedures relacionados
- Designar uma estrutura de pacote como pública ou privada
- Chamar uma estrutura de pacote
- Descrever o uso de um pacote sem corpo

27

Sumário

Os pacotes são usados para agrupar funções e procedures relacionados. Eles melhoram a organização, o gerenciamento, a segurança e o desempenho do código.

Os pacotes consistem em dois componentes: especificação do pacote e corpo do pacote. Você pode alterar o corpo de um pacote sem afetar a especificação do pacote.

Os pacotes permitem ocultar o código-fonte dos usuários. Quando você chama um pacote pela primeira vez, o pacote inteiro é carregado na memória. Isso reduz o acesso ao disco para chamadas subsequentes.

Visão Geral do Exercício 11: Criando e Usando Pacotes

Este exercício aborda os seguintes tópicos:

- Criando pacotes
- Chamando unidades de programa de pacote

28

Exercício 11: Visão Geral

Neste exercício, você criará especificações de pacotes e corpos de pacote. Então você chama as estruturas dos pacotes, usando dados de amostra.

Observação: Se estiver usando o SQL Developer, os erros de tempo de compilação são exibidos na tab Message Log. Se estiver usando o SQL*Plus para criar seu código armazenado, use `SHOW ERRORS` para exibir erros de compilação.