Database Programming with PL/SQL

10-1: Creating Packages

Practice Activities

Vocabulary

Identify the vocabulary word for each definition below:

	The interface to your applications that declares the constructs (procedures, functions, variables, and so on) which are visible to the calling environment.
	A two-part structure in which the detailed package body code is invisible to the calling environment, which can only see the specification. If changes to the code are needed, the body can be edited and recompiled without having to edit or recompile the specification.
	An option that drops and re-creates the package body.
	This contains the executable code of the subprograms which were declared in the package specification. It may also contain its own variable declarations.
	Containers that enable you to group together related PL/SQL subprograms, variables, cursors, and exceptions.

Try It / Solve It

1. Name at least four objects that can be put into a package.

2. Name the two components of a package and explain the difference between them.

3. Which component must be created first, and why?

4. Write a query against a Data Dictionary to display the list of packages you own.

5. Create the specification for the check_emp_pkg which you studied in this lesson. The specification should declare a constant and two procedures, as follows:

   - g_max_length_of_service, datatype NUMBER, initialized to 100
   - chk_hiredate with one input parameter having the same datatype as employees.hire_date
   - chk_dept_mgr with two input parameters having the same datatypes as employees.employee_id and employees.manager_id.

6. Create the package body for check_emp_pkg. Remember that the names and parameters of the procedures in the body must be identical to those in the specification, or the body will not compile.

   The code for chk_hiredate should RAISE_APPLICATION_ERROR if the employee was hired more than 100 years ago (hint: use MONTHS_BETWEEN, comparing with g_max_length_of_service * 12). The second procedure, chk_dept_mgr, accepts two input parameters: an employee_id and a manager_id. The code should find the manager of the employee's department and check whether this manager has the same manager_id as the second parameter. If the manager_id is the same, display a suitable "success" message; if they are different, raise an application error. Include an exception handler for NO_DATA_FOUND.

   The following sample data from the employees and departments tables may help you:

   Departments:

DEPARTMENT_ID	MANAGER_ID
80	149

Employees:

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
149	Zlotkey	80
174	Abel	80
176	Taylor	80
212	Hooper	80

Passing parameters (174,149) would be successful, while (174,176) would raise an error.

7. Test the chk_hiredate procedure using input value 17-Jan-1987 (it should succeed).

8. Test the chk_dept_mgr procedure twice using input values (174,149) and (174,176). The first should succeed while the second should fail.

9. Now you want to modify the package so that the chk_dept_mgr procedure displays a different error message if the two manager_ids are different. What do you need to recreate: the Specification, the Body, or both?