Database Programming with PL/SQL

8-1: Creating Procedures Practice

Activities

Vocabulary

Identify the vocabulary word for each definition below:

SUBPROGRAMS	Named PL/SQL blocks that are compiled and stored in the database.
IS OR AS	Indicates the DECLARE section of a subprogram.
ANONYMOUS BLOCKS	Unnamed executable PL/SQL blocks that cannot be reused or stored in the database for later use.
PROCEDURES	Named PL/SQL blocks that can accept parameters and are compiled and stored in the database.

Try It / Solve It

1. What is the difference between the following two pieces of code?

El código A es un bloque anónimo y el código B es un procedimiento para actualizar el salario.

CODE SAMPLE A

```
DECLARE
  v_empid           employees.employee_id%TYPE := 100;
v_percent_increase      NUMBER(2,2) := .05;
BEGIN
  UPDATE employees
    SET salary = (salary * v_percent_increase) + salary
    WHERE employee_id = v_empid;
```

```
END;
```

CODE SAMPLE B

```
CREATE PROCEDURE pay_raise
    (p_empid employees.employee_id%TYPE,
p_percent_increase  NUMBER)
IS
BEGIN
  UPDATE employees
    SET salary = (salary * p_percent_increase) + salary
    WHERE employee_id = p_empid;
END pay_raise;
```

2. In your own words, list the benefits of subprograms.

Un subprograma puede ser reutilizado varias veces.

Es más fácil darle mantenimiento.

3. In your own words, describe a stored procedure.

Un procedimiento de almacenado, encapsula el código que se utiliza para actualizar o insertar datos a una tabla.

4. The remaining questions in this practice use a copy of the employees table. Create the copy by executing the following SQL statement:

```
CREATE TABLE employees_dup AS SELECT * from employees;
```

A. Use the code below to create a procedure in Application Express. Save the definition of your procedure in case you need to modify it later. In the “Save SQL” popup, name your saved work “My name change procedure.”

```
CREATE OR REPLACE PROCEDURE name_change IS
BEGIN
  UPDATE employees_dup
    SET first_name = 'Susan'
    WHERE department_id = 80;
END name_change;
```

B. Execute the procedure by running the following anonymous block:

```
BEGIN
name_change;
END;
```

C. SELECT from the table to check that the procedure has executed correctly and performed the UPDATE.

SELECT * FROM USER_PROCEDURES WHERE object_name = 'NAME_CHANGE'

5. Create a second procedure named pay_raise which changes the salary of all employees in employees_dup to a new value of 30000. Execute the procedure from an anonymous block, then SELECT from the table to check that the procedure has executed correctly.

```
CREATE OR REPLACE PROCEDURE pay_raise IS
BEGIN
  UPDATE employees_dup
    SET salary= 30000;
END pay_raise;

BEGIN
pay_raise;
END;
```

6. Retrieve your first name_change procedure by clicking on its name in the Saved SQL window. Modify the code to remove OR REPLACE from the CREATE statement, and introduce a deliberate error into the code, for example by misspelling a keyword: UPDAT employees_dup. Execute your code to recreate the procedure. What happens?

Marca un error en la compilación del código.

7. Now correct the procedure code by reinserting the OR REPLACE clause and correcting your deliberate spelling error. Execute your code to recreate the procedure. Now what happens?

   El procedimiento ya funciona

8. Create, save, and execute a procedure which updates the salary of employees in employees_dup according to the following rules:

   - if the employee is in department 80, the new salary = 1000
   - if the employee is in department 50, the new salary = 2000
   - if the employee is in any other department, the new salary = 3000.

   You will need to include three UPDATE statements, one for each of the above rules. In a later lesson you will learn how to avoid this. Execute your procedure from an anonymous block and verify that the updates have been performed correctly.

```
CREATE OR REPLACE PROCEDURE UP_SALARY IS
BEGIN

UPDATE employees_dup
    SET salary= 1000 WHERE department_id=80;
UPDATE employees_dup
      SET salary= 2000 WHERE department_id=50;
UPDATE employees_dup
      SET salary= 3000 WHERE department_id!=50 AND department_id!=80;
END UP_SALARY;
```