

# Database Programming with PL/SQL 8-1: Creating Procedures Practice Activities **Vocabulary**

Identify the vocabulary word for each definition below:

|  |  |
| --- | --- |
| **Subprograme** | Named PL/SQL blocks that are compiled and stored in the database. |
| **IS or AS** | Indicates the DECLARE section of a subprogram. |
| **Anonymus Blocks** | Unnamed executable PL/SQL blocks that cannot be reused or stored in the database for later use. |
| **Procedure** | 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?

## 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;

Prima reprezinta un bloc anonim, cea de-a doua reprezinta o procedura.

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

Subprogramele ajuta la modularizarea codului, adica vor face mai usoara mentenanta si codul va fi reutilizabil.Totodata imbunatatesc securitatea datelor si performanta.

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

O procedura este un bloc PL/SQL care accepta parametri si realizeaza o actiune.

1. 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;

* 1. 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;

* 1. Execute the procedure by running the following anonymous block:

BEGIN name\_change; END;

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

SELECT first\_name, department\_id

FROM employees\_dup;

1. 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;

SELECT first\_name, salary

FROM employees\_dup;

1. etrieve 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?

**ORA-00955: name is already used by an existing object**

1. 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?

Procedure created.

1. 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 update\_salary IS

BEGIN

UPDATE employees\_dup

SET salary = 3000;

UPDATE employees\_dup

SET salary = 1000

WHERE department\_id = 80;

UPDATE employees\_dup

SET salary = 2000

WHERE department\_id = 50;

END update\_salary;

BEGIN

update\_salary;

END;

SELECT first\_name, salary, department\_id

FROM employees\_dup