Database Programming with PL/SQL
8-2: Using Parameters in Procedures
Practice Activities

Vocabulary

Identify the vocabulary word for each definition below:

Parameters	Pass or communicate data between the caller and subprogram.
Argument	The actual value assigned to a parameter.
Actual parameter	Can be literal values, variables, or expressions that are provided in the parameter list of a called subprogram.
Formal Paramter	A parameter name declared in the procedure heading.

Try It / Solve It

1. In your own words, describe parameters and the purpose they serve in PL/SQL subprograms.
   Parameters are variables passed from the caller program to the subprogram.
   Parameters are used to make procedures more flexible, so that it can be used for more than one purpose or more than one piece of data.
2. Using the COUNTRIES table:

   A. Create a procedure that accepts a country_id as a parameter and displays the name of the country and its capitol city. Name your procedure get_country_info. Save your procedure definition for later use.

   B. Execute your procedure from an anonymous block, using country_id 90.

   C. Re-execute the procedure from the anonymous block, this time using country_id 95. What happens?

   D. Retrieve your procedure code from Saved SQL and modify it to trap the NO_DATA_FOUND exception in an exception handler. Execute the modified procedure using country_id 95 again. Now what happens?

3. In your own words, describe what a formal parameter is and what an actual parameter is. Also, name three variations for an actual parameter.

The formal parameter is a parameter declared in the procedure heading.
The actual parameter is the parameter or value in the calling environment.
Actual parameters can be literal values, variables or expressions.

4. Procedure Exercise:

   A. Write a procedure that displays the number of countries in a given region whose highest elevations exceed a given value. The procedure should accept two formal parameters, one for a region_id and the other for an elevation value for comparison. Use DBMS_OUTPUT.PUT_LINE to display the results in a message. Save your procedure code.

   B. Execute your procedure using the value 5 for the region_id and 2000 for the highest elevation.

   C. DESCRIBE your procedure to check the names and datatypes of its formal parameters.

   D. Retrieve your procedure code from Saved SQL and modify it to accept a third formal parameter of datatype CHAR. The procedure should display a count of the number of countries in a given region whose highest elevations exceed a given value and whose country name starts with a given alphabetic character. Your SELECT statement should include a WHERE condition to compare the first character of each country's name with the third parameter value (Hint: use SUBSTR). Save your work again and DESCRIBE the modified procedure.

   E. Write an anonymous block which declares three variables to store actual parameter values for the region_id, elevation, and area, and then executes the procedure passing these values. Execute the block using values 5, 2000, and 'B'.

   F. Modify your anonymous block to use the same actual parameter values but pass them to the procedure in a different order: (5, 'B', 2000). Execute the block. What happens and why?