Practices for Lesson 4: Writing Executable Statements

Chapter 4

Practice 4: Writing Executable Statements

Note: If you have executed the code examples for this lesson, make sure that you execute the following code before starting this practice:

```
DROP sequence my_seq;
```

In this practice, you examine and write executable statements.

```
DECLARE
 v_weight    NUMBER(3) := 600;
 v_message   VARCHAR2(255) := 'Product 10012';
BEGIN
  DECLARE
   v_weight    NUMBER(3) := 1;
   v_message   VARCHAR2(255) := 'Product 11001';
   v_new_locn  VARCHAR2(50) := 'Europe';
  BEGIN
   v_weight := v_weight + 1;
   v_new_locn := 'Western ' || v_new_locn;
1 →
  END;
 v_weight := v_weight + 1;
 v_message := v_message || ' is in stock';
 v_new_locn := 'Western ' || v_new_locn;
2 →
END;
/
```

1. Evaluate the preceding PL/SQL block and determine the data type and value of each of the following variables, according to the rules of scoping.
   a. The value of `v_weight` at position 1 is:
   b. The value of `v_new_locn` at position 1 is:
   c. The value of `v_weight` at position 2 is:
   d. The value of `v_message` at position 2 is:
   e. The value of `v_new_locn` at position 2 is:

```
DECLARE
   v_customer     VARCHAR2(50) := 'Womansport';
   v_credit_rating   VARCHAR2(50) := 'EXCELLENT';
BEGIN
    DECLARE
       v_customer  NUMBER(7) := 201;
       v_name VARCHAR2(25) := 'Unisports';
    BEGIN
       v_credit_rating :='GOOD';
       …
    END;
  …
END;
```

2. In the preceding PL/SQL block, determine the value and data type of each of the following cases:
   a. The value of `v_customer` in the nested block is:
   b. The value of `v_name` in the nested block is:
   c. The value of `v_credit_rating` in the nested block is:
   d. The value of `v_customer` in the main block is:
   e. The value of `v_name` in the main block is:
   f. The value of `v_credit_rating` in the main block is:
3. Use the same session that you used to execute the practices in the lesson titled "Declaring PL/SQL Variables." If you have opened a new session, execute `lab_03_05_soln.sql`. Then, edit `lab_03_05_soln.sql` as follows:
   a. Use single-line comment syntax to comment the lines that create the bind variables, and turn on `SERVEROUTPUT`.
   b. Use multiple-line comments in the executable section to comment the lines that assign values to the bind variables.
   c. In the declaration section:
      1) Declare and initialize two temporary variables to replace the commented out bind variables
      2) Declare two additional variables: `v_fname` of type `VARCHAR2` and size 15, and `v_emp_sal` of type `NUMBER` and size 10
   d. Include the following SQL statement in the executable section:

```
SELECT first_name, salary INTO v_fname, v_emp_sal
  FROM employees WHERE employee_id=110;
```

   e. Change the line that prints "Hello World" to print "Hello" and the first name. Then, comment the lines that display the dates and print the bind variables.
   f. Calculate the contribution of an employee toward the provident fund (PF).
   PF is 12% of the basic salary, and the basic salary is 45% of the salary. Use local variables for the calculation. Try to use only one expression to calculate the PF. Print the employee's salary and his or her contribution toward PF.
   g. Execute and save your script as `lab_04_03_soln.sql`. The sample output is as follows:

```
PL/SQL procedure successfully completed.

 Hello John
YOUR SALARY IS : 8200
YOUR CONTRIBUTION TOWARDS PF:
     442.8
```

Solution 4: Writing Executable Statements

In this practice, you examine and write executable statements.

```
DECLARE
 v_weight    NUMBER(3) := 600;
 v_message   VARCHAR2(255) := 'Product 10012';
BEGIN
  DECLARE
   v_weight    NUMBER(3) := 1;
   v_message   VARCHAR2(255) := 'Product 11001';
   v_new_locn  VARCHAR2(50) := 'Europe';
  BEGIN
   v_weight := v_weight + 1;
   v_new_locn := 'Western ' || v_new_locn;
1 →
  END;
 v_weight := v_weight + 1;
 v_message := v_message || ' is in stock';
 v_new_locn := 'Western ' || v_new_locn;
2 →
END;
/
```

1. Evaluate the preceding PL/SQL block and determine the data type and value of each of the following variables, according to the rules of scoping:
   a. The value of `v_weight` at position 1 is:
      2
      The data type is `NUMBER`.
   b. The value of `v_new_locn` at position 1 is:
      Western Europe
      The data type is `VARCHAR2`.
   c. The value of `v_weight` at position 2 is:
      601
      The data type is `NUMBER`.
   d. The value of `v_message` at position 2 is:
      Product 10012 is in stock
      The data type is `VARCHAR2`.
   e. The value of `v_new_locn` at position 2 is:
      Illegal because `v_new_locn` is not visible outside the subblock

```
DECLARE
   v_customer     VARCHAR2(50) := 'Womansport';
   v_credit_rating   VARCHAR2(50) := 'EXCELLENT';
BEGIN
    DECLARE
       v_customer  NUMBER(7) := 201;
       v_name VARCHAR2(25) := 'Unisports';
    BEGIN
       v_credit_rating :='GOOD';
       …
    END;
  …
END;
```

2. In the preceding PL/SQL block, determine the value and data type for each of the following cases:
   a. The value of `v_customer` in the nested block is:
      201
      The data type is `NUMBER`.
   b. The value of `v_name` in the nested block is:
      Unisports
      The data type is `VARCHAR2`.
   c. The value of `v_credit_rating` in the nested block is:
      GOOD
      The data type is `VARCHAR2`.
   d. The value of `v_customer` in the main block is:
      Womansport
      The data type is `VARCHAR2`.
   e. The value of `v_name` in the main block is:
      Null. **`name` is not visible in the main block and you would see an error.**
   f. The value of `v_credit_rating` in the main block is:
      EXCELLENT
      The data type is `VARCHAR2`.
3. Use the same session that you used to execute the practices in the lesson titled "Declaring PL/SQL Variables." If you have opened a new session, execute `lab_03_05_soln.sql`. Then, edit `lab_03_05_soln.sql` as follows:
   a. Use single-line comment syntax to comment the lines that create the bind variables, and turn on `SERVEROUTPUT`.

```
-- VARIABLE b_basic_percent NUMBER
-- VARIABLE b_pf_percent NUMBER
SET SERVEROUTPUT ON
```

b. Use multiple-line comments in the executable section to comment the lines that assign values to the bind variables.

```
   /*:b_basic_percent:=45;
   :b_pf_percent:=12;*/
```

c. In the declaration section:
   1) Declare and initialize two temporary variables to replace the commented out bind variables
   2) Declare two additional variables: `v_fname` of type `VARCHAR2` and size 15, and `v_emp_sal` of type `NUMBER` and size 10

```
DECLARE
   v_basic_percent NUMBER:=45;
   v_pf_percent NUMBER:=12;
   v_fname VARCHAR2(15);
   v_emp_sal NUMBER(10);
```

d. Include the following SQL statement in the executable section:

```
   SELECT first_name, salary INTO v_fname, v_emp_sal
     FROM employees WHERE employee_id=110;
```

e. Change the line that prints “Hello World” to print “Hello” and the first name. Then, comment the lines that display the dates and print the bind variables.

```
DBMS_OUTPUT.PUT_LINE(' Hello '|| v_fname);
/*   DBMS_OUTPUT.PUT_LINE('TODAY IS : '|| v_today);
DBMS_OUTPUT.PUT_LINE('TOMORROW IS : ' || v_tomorrow);*/
...
...

/
--PRINT b_basic_percent
--PRINT b_basic_percent
```

f. Calculate the contribution of an employee toward the provident fund (PF).
PF is 12% of the basic salary, and the basic salary is 45% of the salary. Use local variables for the calculation. Try to use only one expression to calculate the PF. Print the employee’s salary and his or her contribution toward PF.

```
   DBMS_OUTPUT.PUT_LINE('YOUR SALARY IS : '||v_emp_sal);
   DBMS_OUTPUT.PUT_LINE('YOUR CONTRIBUTION TOWARDS PF:
     '||v_emp_sal*v_basic_percent/100*v_pf_percent/100);
END;
```

g. Execute and save your script as `lab_04_03_soln.sql`. The sample output is as follows:

```
PL/SQL procedure successfully completed.

 Hello John
YOUR SALARY IS : 8200
YOUR CONTRIBUTION TOWARDS PF:
     442.8
```