

Recognizing the Scope of Variables

Terminology

1. _____ The portion of the program where the variable can be accessed without using a qualifier.
2. _____ A label given to a block.
3. _____ The portion of a program in which the variable is declared and is accessible.

Try It / Solve It

1. In your own words, explain the difference between the scope of a variable and the visibility of a variable. Give an example where a variable **is** in scope but **not** visible.

2. Enter and run the following PL/SQL code.

```
DECLARE
  v_last_name  employees.last_name%TYPE;
BEGIN
  SELECT last_name INTO v_last_name
  FROM employees WHERE employee_id = 100;
  /* This employee's last name is King */
DECLARE
  v_last_name  employees.last_name%TYPE;
BEGIN
  SELECT last_name INTO v_last_name
  FROM employees WHERE employee_id = 107;
  /* This employee's last name is Lorentz */
  DBMS_OUTPUT.PUT_LINE(v_last_name);
END;
DBMS_OUTPUT.PUT_LINE(v_last_name);
END;
```

- A. What output is displayed, and why? Save your code.
- B. Modify the code by using a block label so that the inner block displays both employees' last names while the outer block displays nothing. Execute and save your code.
- C. Modify the code again so that this time, the outer block displays both employees' last names while the inner block displays nothing. Execute your code.

3. Enter and execute the following block, which uses a cursor with a parameter to fetch and display the country name and area of countries in Central America. You should see eight countries displayed. Save your code.

```
DECLARE
  v_region_id  wf_world_regions.region_id%TYPE;
  CURSOR country_curs
    (p_region_id wf_world_regions.region_id%TYPE) IS
    SELECT country_name, area
      FROM    wf_countries
     WHERE    region_id = p_region_id
     ORDER BY country_name;
  country_rec  country_curs%ROWTYPE;

BEGIN
  SELECT region_id INTO v_region_id
    FROM wf_world_regions
   WHERE region_name = 'Central America';
  OPEN country_curs(v_region_id);
  LOOP
    FETCH country_curs INTO country_rec;
    EXIT WHEN country_curs%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE (country_rec.country_name ||
                          ' ' || country_rec.area);
  END LOOP;
  CLOSE country_curs;
END;
```

- A. Modify the code to place all the cursor references (DECLARE, OPEN, FETCH in a loop and CLOSE) in an inner nested block. Leave the declaration of v_region_id and the single-row SELECT in the outer block. Execute your code (you should see the same output as in step a). Save your code.
- B. Now modify the code again by moving the CLOSE *cursor-name* statement to the outer block. Execute your code. What happens and why ?

4. Enter and run the following code twice, using country_ids 5 (which does not exist) and 672 (Antarctica, which does exist but has no currency).

```
DECLARE
  v_country_name wf_countries.country_name%TYPE;
  v_currency_code wf_countries.currency_code%TYPE;
BEGIN
  DECLARE
    e_no_currency EXCEPTION;
  BEGIN
    SELECT country_name, currency_code
      INTO v_country_name, v_currency_code
    FROM wf_countries
      WHERE country_id = 5; -- repeat with 672
    IF v_currency_code = 'NONE' THEN
      RAISE e_no_currency;
    END IF;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('This country does not exist');
    WHEN e_no_currency THEN
      DBMS_OUTPUT.PUT_LINE('This country exists but has no currency');
  END;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Another type of error occurred');
END;
```

- A. Explain the output. Save your code.
- B. Modify the code to move the two exception handlers to the outer block. Leave the declaration of e_no_currency in the inner block. Execute twice, again using country_ids 5 and 672. Now what happens and why? Save your code.
- C. Modify the code again to move the declaration of e_no_currency to the outer block. Re-execute again using country_ids 5 and 672. Now what happens and why?