Database Programming with PL/SQL
3-2: Retrieving Data in PL/SQL
Practice Activities

Vocabulary

No new vocabulary for this lesson

Try It / Solve It

1. State whether each of the following SQL statements can be included directly in a PL/SQL block.

Statement	Valid in PL/SQL	Not Valid in PL/SQL
ALTER USER SET password = 'oracle';		x
CREATE TABLE test (a NUMBER);		x
DROP TABLE test;		x
SELECT emp_id INTO v_id FROM employees;	x	
GRANT SELECT ON employees TO PUBLIC;		x
INSERT INTO grocery_items (product_id, brand, description) VALUES (199, 'Coke', 'Soda');	x	
REVOKE UPDATE ON employees FROM PUBLIC;		x
ALTER TABLE employees RENAME COLUMN employee_id TO emp_id;		x
DELETE FROM grocery_items WHERE description = 'Soap';	x	

2. Create a PL/SQL block that selects the maximum department_id in the departments table and stores it in the v_max_deptno variable. Display the maximum department_id. Declare v_max_deptno to be the same datatype as the department_id column. Include a SELECT statement to retrieve the highest department_id from the departments table. Display the variable v_max_deptno.

```
DECLARE
v_max_deptno departments.department_id%TYPE;
BEGIN
SELECT MAX(department_id) INTO v_max_deptno FROM departments;
DBMS_OUTPUT.PUT_LINE('Max id: ' || v_max_deptno);

END;
```

3. The following code is supposed to display the lowest and highest elevations for a country name entered by the user. However, the code does not work. Fix the code by following the guidelines for retrieving data that you learned in this lesson.

```
DECLARE
  v_country_name       countries.country_name%TYPE := Federative Republic of Brazil;
  v_lowest_elevation   countries.lowest_elevation%TYPE;
  v_highest_elevation  countries.highest_elevation%TYPE;
BEGIN
  SELECT lowest_elevation, highest_elevation
    FROM countries;
  DBMS_OUTPUT.PUT_LINE('The lowest elevation in '
    || v_country_name || ' is ' || v_lowest_elevation
    || ' and the highest elevation is ' || v_highest_elevation || '.');
END;
```

```
DECLARE
v_country_name countries.country_name%TYPE:='Federative Republic of Brazil';
v_lowest_elevation number(10);
v_highest_elevation number(10);
BEGIN
SELECT MIN(v_lowest_elevation), MAX (v_highest_elevation) INTO
v_lowest_elevation ,v_highest_elevation
FROM countries;
DBMS_OUTPUT.PUT_LINE('The lowest elevation in ' || v_country_name || ' is ' ||v_lowest_elevation || ' and the highest elevation is ' || v_highest_elevation || '.');
END;
```

4. Run the following anonymous block. It should execute successfully.

```
DECLARE
   v_emp_lname    employees.last_name%TYPE;
   v_emp_salary   employees.salary%TYPE;
BEGIN
   SELECT last_name, salary INTO v_emp_lname, v_emp_salary
     FROM employees
     WHERE job_id = 'AD_PRES';
   DBMS_OUTPUT.PUT_LINE(v_emp_lname || ' ' || v_emp_salary);
END;
```

A. Now modify the block to use 'IT_PROG' instead of 'AD_PRES' and re-run it. Why does it fail this time? ORA-01422: exact fetch returns more than requested number of rows
It fails because it returns more than one row and we want to put the values in v_emp_lname and v_emp_salary, so we only need one row.

B. Now modify the block to use 'IT_PRAG' instead of 'IT_PROG' and re-run it. Why does it still fail? ORA-01403: no data found
There is no data with job_id 'IT_PRAG' so it fails because you have nothing to assign to our variables.

5. Use (but don't execute) the following code to answer this question:

```
DECLARE
  last_name VARCHAR2(25) := 'Fay';
BEGIN
  UPDATE emp_dup
    SET first_name = 'Jennifer'
    WHERE last_name = last_name;
END;
```

What do you think would happen if you ran the above code? Write your answer here and then follow the steps below to test your theory.

A. Create a table called emp_dup that is a duplicate of employees.

```
CREATE TABLE emp_dup AS (SELECT * FROM employees);
```

B. Select the first_name and last_name values for all rows in emp_dup.

```
select first_name, last_name from emp_dup;
```

C. Run the anonymous PLSQL block shown at the beginning of this question.

```
C.
DECLARE
last_name VARCHAR2(25) := 'Fay';
BEGIN
UPDATE emp_dup SET
first_name = 'Jennifer' WHERE
last_name = last_name;
END;
```

D. Select the first_name and last_name columns from emp_dup again to confirm your theory.

```
select first_name, last_name from emp_dup;
```

E. Now we are going to correct the code so that it changes only the first name for the employee whose last name is “Fay”. Drop emp_dup and re-create it.

```
drop table emp_dup;
CREATE TABLE emp_dup AS (SELECT * FROM employees);
```

F. Modify the code shown at the beginning of this question so that for the employee whose last_name = ”Fay”, the first_name is updated to Jennifer. Run your modified block.

```
DECLARE
last_name VARCHAR2(25) :='Fay';
BEGIN
END;
```

```
UPDATE emp_dup SET first_name = 'Jennifer' WHERE last_name = 'Fay';
```

G. Confirm that your update statement worked correctly.

```
select first_name, last_name from emp_dup where last_name = 'Fay';
```

6. Is it possible to name a column in a table the same name as the table? Create a table to test this question. Don't forget to populate the table with data.

```
Yes, it is possible
```

7. Is it possible to have a column, table, and variable, all with the same name? Using the table you created in the question above, write a PL/SQL block to test your theory.

```
6.
CREATE TABLE test(
testID NUMBER(4) NOT NULL,
test VARCHAR2(50),
PRIMARY KEY(testID));

INSERT INTO test(testID, test) values (1, 'Sir de test');
select test from test;
```

```
7.
Yes, it is possible

DECLARE
test VARCHAR2(25) := 'Test';
BEGIN
UPDATE test SET test = 'Test'
WHERE testID = 1;
END;
```