Database Programming with PL/SQL

3-2: Retrieving Data in PL/SQL

Practice Activities

Vocabulary

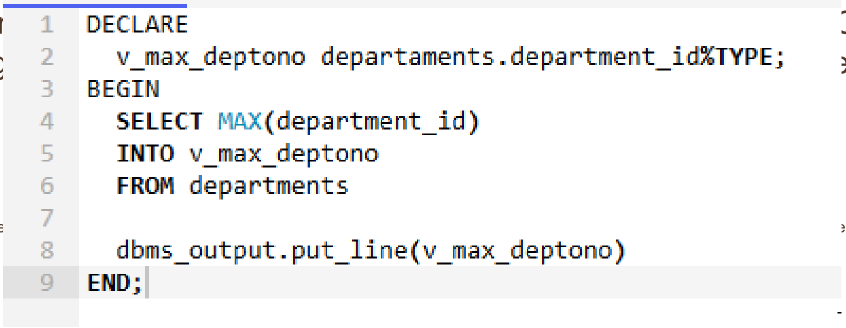
No new vocabulary for this lesson

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	

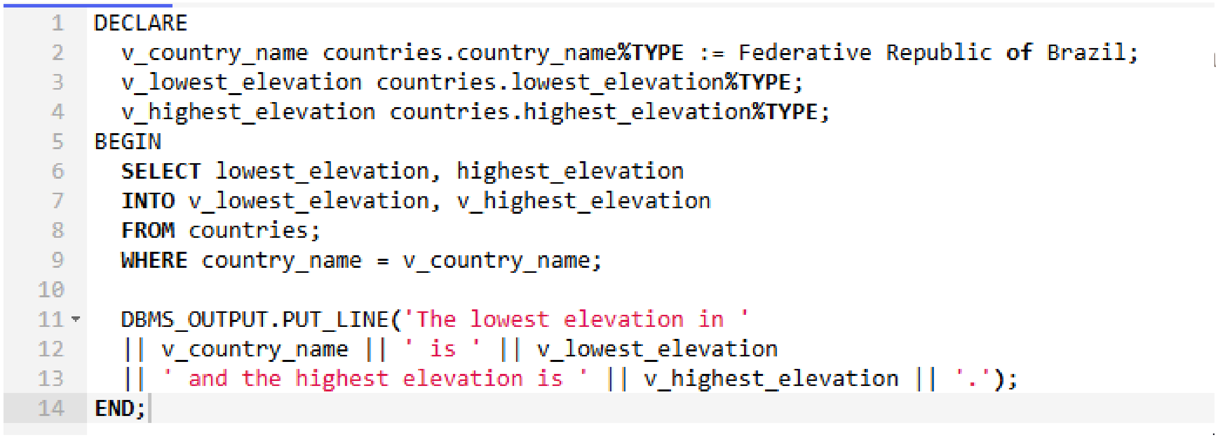
2. Create a PL/SQL block that selects the maximum department_id in the departments table and stores it in the v_max_deptno variable. Display the maximum department_id. Declare v_max_deptno to be the sar CT statement to retrieve the hig e variable v_max_deptno.



```
DECLARE
  v_max_deptono departaments.department_id%TYPE;
BEGIN
  SELECT MAX(department_id)
  INTO v_max_deptono
  FROM departments

  dbms_output.put_line(v_max_deptono)
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
  INTO v_lowest_elevation, v_highest_elevation
  FROM countries;
  WHERE country_name = v_country_name;

  DBMS_OUTPUT.PUT_LINE('The lowest elevation in '
  || v_country_name || ' is ' || v_lowest_elevation
  || ' and the highest elevation is ' || v_highest_elevation || '.');
END;
```

zil;

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?

   Because there are several employees with the same ID

B. Now modify the block to use 'IT_PRAG' instead of 'IT_PROG' and re-run it. Why does it still fail?

   Because IT_PRAG does not exist.

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.

Error, WHERE last_name = last_name;

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.

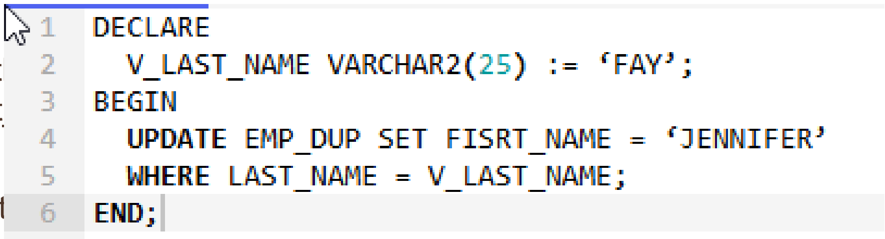
SELECT first_name, last_nameFROM emp_dup;

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

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

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;



```
DECLARE
  V_LAST_NAME VARCHAR2(25) := ‘FAY’;
BEGIN
  UPDATE EMP_DUP SET FISRT_NAME = ‘JENNIFER’
  WHERE LAST_NAME = V_LAST_NAME;
END;
```

F. Modify the code shown at t
last_name = ”Fay”, the first

G. Confirm that your update st

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.

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 GROCERY_ITEM (
   PRODUCT_ID NUMBER(6,0) PRIMARY KEY,
   BRAND VARCHAR2(100),
   GROCERY_ITEM VARCHAR2(100)
);

INSERT INTO GROCERY_ITEM (PRODUCT_ID, BRAND, GROCERY_ITEM)
VALUES (100, 'COLGATE', 'PASTA DENTAL');
```

Yes, it's possible.