Database Programming with PL/SQL

5-2: Using Explicit Cursor Attributes

Practice Activities

Vocabulary

|  |  |
| --- | --- |
| %ROWTYPE | Declares a record with the same fields as the cursor on which it  is based |
| RECORD | A composite data type in PL/SQL, consisting of a number of  fields each with their own name and data type |
| ISOPEN | Returns the status of the cursor |
| ROWCOUNT | An attribute that processes an exact number of rows or counts  the number of rows fetched in a loop |
| %NOTFOUND | An attribute used to determine whether the most recent FETCH  statement successfully returned a row |

Try It/Solve It

1. In your own words, explain the advantage of using %ROWTYPE to declare a record structure

based on a cursor declaration.

Nu tre sa ne cream a bunch of variables pt fiecare coloana care ne intereseaza din select. Zicem numa’ numecursor%rowtype si ne returneaza o lista generata automat de variabile cu tipurile bune.

Ele se acceseaza prin numevariabila.numecoloana

2. Write a PL/SQL block to read through rows in the countries table for all countries in region 5

(South America region). For each selected country, display the country\_name,

national\_holiday\_date, and national\_holiday\_name. Use a record structure to hold all the columns

selected from the countries table.

Hint: This exercise is similar to question 4G in the previous lesson. Use your solution as a starting

point for this exercise.

DECLARE

 CURSOR countries\_cur IS

 SELECT country\_name, national\_holiday\_date, national\_holiday\_name

 FROM wf\_countries

 WHERE region\_id = 5;

 v\_countries countries\_cur%ROWTYPE;

BEGIN

 OPEN countries\_cur;

 LOOP

FETCH countries\_cur INTO v\_countries;

EXIT WHEN countries\_cur%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_countries.country\_name || ' ' || v\_countries.national\_holiday\_date || ' ' || v\_countries.national\_holiday\_name);

END LOOP;

 CLOSE countries\_cur;

END;

3. For this exercise, you use the employees table. Create a PL/SQL block that fetches and displays

the six employees with the highest salary. For each of these employees, display the first name,

last name, job id, and salary. Order your output so that the employee with the highest salary is

displayed first. Use %ROWTYPE and the explicit cursor attribute %ROWCOUNT.

DECLARE

 CURSOR employees\_cursor IS

 SELECT \*

 FROM employees

 ORDER BY salary DESC;

 v\_emp\_rec employees\_cursor%ROWTYPE;

BEGIN

 OPEN employees\_cursor;

 LOOP

FETCH employees\_cursor INTO v\_emp\_rec;

EXIT WHEN employees\_cursor%ROWCOUNT > 6;

DBMS\_OUTPUT.PUT\_LINE(v\_emp\_rec.first\_name || ' ' || v\_emp\_rec.last\_name || ' ' || v\_emp\_rec.job\_id || ' ' || v\_emp\_rec.salary);

END LOOP;

 CLOSE employees\_cursor;

END;

4. Look again at the block you created in question 3. What if you wanted to display 21 employees instead of 6? There are only 20 rows in the employees table. What do you think would happen?

Ar trebui sa primim o eroare

5. In real life we would not know how many rows the table contained. Modify your block from question 3 so that it will exit from the loop when either 21 rows have been fetched and displayed, or when there are no more rows to fetch. Test the block again.

DECLARE

 CURSOR employees\_cursor IS

 SELECT \*

 FROM employees

 ORDER BY salary DESC;

 v\_emp\_rec employees\_cursor%ROWTYPE;

BEGIN

 OPEN employees\_cursor;

 LOOP

FETCH employees\_cursor INTO v\_emp\_rec;

EXIT WHEN employees\_cursor%ROWCOUNT > 6 OR employees\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_emp\_rec.first\_name || ' ' || v\_emp\_rec.last\_name || ' ' || v\_emp\_rec.job\_id || ' ' || v\_emp\_rec.salary);

END LOOP;

 CLOSE employees\_cursor;

END;