

PL/SQL 4-4: Iterative Control: WHILE and FOR Loops

Agustín Alejandro Mota Hinojosa

November 10, 2023

Contents

1	Vocabulary	1
2	Try it / Solve it	1

1 Vocabulary

Repeats a sequence of statements until the controlling condition is no longer TRUE.

WHILE Loop

Repeats a sequence of statements until a set number of iterations have been completed.

FOR Loop

2 Try it / Solve it

1. Write a PL/SQL block to display the `country_id` and `country_name` values from the `COUNTRIES` table for `country_id` whose values range from 51 through 55. Use a WHILE loop. Increment a variable from 51 through 55. Test your variable to see when it reaches 55. EXIT the loop after you have displayed the 5 countries.

```
DECLARE
  v_country_id wf_countries.country_id%TYPE := 51;
  v_country_name wf_countries.country_name%TYPE;
BEGIN
```

```

WHILE v_country_id <= 55 LOOP
    SELECT country_name INTO v_country_name
    FROM wf_countries
    WHERE country_id = v_country_id;
    DBMS_OUTPUT.PUT_LINE(v_country_id || ' ' || v_country_name);
    v_country_id := v_country_id + 1;
END LOOP;
END;

```

2. Write a PL/SQL block to display the country_{id} and country_{name} values from the COUNTRIES table for country_{id} whose values range from 51 through 55 in the reverse order. Use a FOR loop.

```

DECLARE
    v_country_id wf_countries.country_id%TYPE;
    v_country_name wf_countries.country_name%TYPE;
BEGIN
    FOR v_country_id IN REVERSE 51..55 LOOP
        SELECT country_name INTO v_country_name
        FROM wf_countries
        WHERE country_id = v_country_id;
        DBMS_OUTPUT.PUT_LINE(v_country_id || ' ' || v_country_name);
    END LOOP;
END;

```

3. Execute the following statements to build a new_{emps} table.

```

DROP TABLE new_emps;
CREATE TABLE new_emps AS SELECT * FROM employees;
ALTER TABLE new_emps ADD stars VARCHAR2(50);

```

- (a) Create a PL/SQL block that inserts an asterisk in the stars column for every whole \$1,000 of an employee's salary. For example, if an employee has a salary of \$7,800, the string "*****" would be inserted, and, if an employee has a salary of \$3,100, the string "*" would be inserted.

```

DECLARE
    v_empno new_emps.employee_id%TYPE := 124;
    v_asterisk new_emps.stars%TYPE := NULL;
    v_sal_in_thousands new_emps.salary%TYPE;

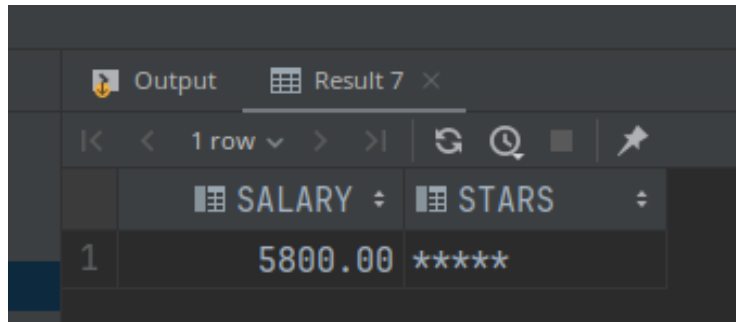
```

```

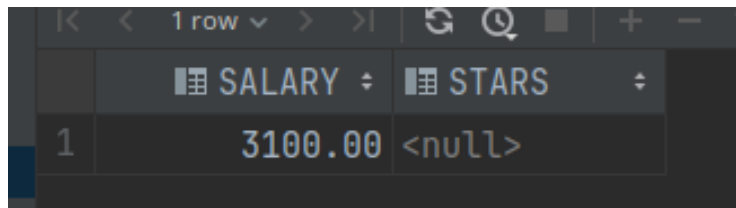
BEGIN
  SELECT NVL(TRUNC(salary/1000), 0) INTO v_sal_in_thousands
  FROM new_emps WHERE employee_id = v_empno;
  FOR i IN 1..v_sal_in_thousands LOOP
    v_asterisk := v_asterisk || '*';
  END LOOP;
  UPDATE new_emps
  SET stars = v_asterisk
  WHERE employee_id = v_empno;
END;

```

- (b) Test your code using employee_{ids} 124 and 142, then confirm the results.



	SALARY	STARS
1	5800.00	*****



	SALARY	STARS
1	3100.00	<null>