

Database Programming with PL/SQL

9-2: Using Functions in SQL Statements

Practice Activities

# Vocabulary

Identify the vocabulary word for each definition below:

|  |  |
| --- | --- |
|  | A function created by the PL/SQL programmer that can be used anywhere there is a value or function. |

# Try It / Solve It

The questions in this Practice use partial copies of the employees and departments tables. Create these copies by executing the following SQL statements:

CREATE TABLE f\_emps

AS SELECT employee\_id, last\_name, salary, department\_id

FROM employees;

CREATE TABLE f\_depts

AS SELECT department\_id, department\_name FROM departments;

1. Create and execute a function sal\_increase using the following two code samples. The first creates a function which returns an employee’s new salary if a percentage increase is granted. The second calls this function in a SELECT statement, using an increase of 5 percent.

CREATE OR REPLACE FUNCTION sal\_increase (p\_salary f\_emps.salary%TYPE,

p\_percent\_incr NUMBER) RETURN NUMBER

IS

BEGIN

RETURN (p\_salary + (p\_salary \* p\_percent\_incr / 100));

END;

SELECT last\_name, salary, sal\_increase(salary, 5) FROM f\_emps;

Now, suppose you want to see the same information in your SELECT statement, but only for those employees for whom the increased salary would be greater than 10000. Write and test two SELECT statements to do this. In the first, do NOT use your function. In the second, use your function. Use an increase of 5 percent. Which do you think is better, and why?

SELECT last\_name, salary

FROM f\_emps

WHERE salary > 10000;

SELECT last\_name, salary, sal\_increase(salary, 5)

FROM f\_emps

WHERE salary > 10000;

Ce-a de a 2-a in care folosim si functia pt ca se vede mai clar.

1. Name five places within a SQL statement where a function can be used. The first one has been done for you (think of four more).

• The column-list of a SELECT statement

• Conditional expressions in the WHERE and HAVING clauses

• The ORDER BY and GROUP BY clauses of a query

• THE VALUES clause of the INSERT statement

• The SET clause of the UPDATE statement

1. Modify your anonymous block from question 1 (the block with the calls to the sal\_increase function) to ORDER the results by the increased salary in descending order (i.e., highest increased salary first).

SELECT last\_name, salary, sal\_increase(salary, 5)

FROM f\_emps

ORDER BY sal\_increase(salary, 5) DESC;

1. Examine the following SELECT statement which lists the total salaries in each department for those departments whose total salary is greater than 20000.

SELECT department\_id, SUM(salary)

FROM f\_emps

GROUP BY department\_id

HAVING SUM(salary) > 20000;

Modify the statement so that it also lists the total salary in each department if a 5 percent increase is granted, and lists those departments whose increased total salary would be greater than 20000. Your modified statement should call the sal\_increase function twice, once in the column\_list and once in the HAVING clause. Test the modified statement.

SELECT department\_id, SUM(salary), sal\_increase(SUM(salary),5)

FROM f\_emps

GROUP BY department\_id

HAVING sal\_increase(SUM(salary), 5) > 20000;

1. The following function accepts a department id as an input parameter and checks whether the department exists in the f\_depts table. Run this code to create the check\_dept function.

CREATE OR REPLACE FUNCTION check\_dept

(p\_dept\_id f\_depts.department\_id%TYPE)

RETURN BOOLEAN IS

v\_dept\_id f\_depts.department\_id%TYPE;

BEGIN

SELECT department\_id INTO v\_dept\_id

FROM f\_depts

WHERE department\_id = p\_dept\_id;

RETURN TRUE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END;

Examine the above function and explain why it could not be used within a SQL statement. Could this function be used within a PL/SQL statement? Why or why not?

BEGIN

DBMS\_OUTPUT.PUT\_LINE(check\_dept(120));

END;

Apare o eroare atunci cand incerc sa utilizez functia check\_dept

**PLS-00306: wrong number or types of arguments in call to 'PUT\_LINE'**

1. Write a procedure called insert\_emp which inserts a new employee into f\_emps. Pass the employee id, last name, salary, and department id to the procedure as IN parameters. The procedure should call your check\_dept function to verify that the passed department id exists in

the f\_depts table. If it exists, insert the employee. If it does not exist, use

DBMS\_OUTPUT.PUT\_LINE to display a suitable error message. Save your code.

CREATE OR REPLACE PROCEDURE insert\_emp

(p\_employee\_id employees.employee\_id%TYPE,

p\_last\_name employees.last\_name%TYPE,

p\_salary employees.salary%TYPE,

p\_department\_id employees.department\_id%TYPE)

IS

a BOOLEAN;

b BOOLEAN;

c BOOLEAN;

d BOOLEAN;

e BOOLEAN;

BEGIN

a := check\_dept(p\_employee\_id);

b := check\_dept(p\_last\_name);

c := check\_dept(p\_salary);

d:= check\_dept(p\_department\_id);

e := a and b and c and d;

IF e = TRUE THEN

INSERT INTO f\_emps(employee\_id, last\_name, salary, department\_id)

VALUES

(p\_employee\_id, p\_last\_name, p\_salary, p\_department\_id);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insertion error');

END IF;

END;

1. Test your insert\_emp procedure from an anonymous block using the following IN parameter values: employee\_id = 800, last\_name = Jokinen, salary = 5000, and department\_id = 750. What happened and why?

BEGIN

insert\_emp(800, 'Jokinen', 5000, 750);

END;

Intervine eroarea: **ORA-06502: PL/SQL: numeric or value error: character to number conversion error**

1. Modify your insert\_emp procedure so that if the department does not exist, the procedure first inserts a new department with the non-existent department id and a department name of ‘Temporary’, and then inserts the employee. Test your procedure again with the same IN values used in the previous question.

CREATE OR REPLACE PROCEDURE insert\_emp

(p\_employee\_id employees.employee\_id%TYPE,

p\_last\_name employees.last\_name%TYPE,

p\_salary employees.salary%TYPE,

p\_department\_id employees.department\_id%TYPE)

IS

a BOOLEAN;

b BOOLEAN;

c BOOLEAN;

d BOOLEAN;

e BOOLEAN;

BEGIN

a := check\_dept(p\_employee\_id);

b := check\_dept(p\_last\_name);

c := check\_dept(p\_salary);

d:= check\_dept(p\_department\_id);

IF d = FALSE THEN

INSERT INTO f\_emps(department\_id)

VALUES

(p\_department\_id);

END IF;

e := a and b and c and d;

IF e = TRUE THEN

INSERT INTO f\_emps(employee\_id, last\_name, salary, department\_id)

VALUES

(p\_employee\_id, p\_last\_name, p\_salary, p\_department\_id);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insertion error');

END IF;

END;

1. Execute two SELECT statements to confirm department id 750 and employee id 800 were added to the F\_DEPTS and F\_EMPS tables, respectively.

1. Create the function get\_sal using the following code:

CREATE OR REPLACE FUNCTION get\_sal

(p\_emp\_id f\_emps.employee\_id%TYPE)

RETURN NUMBER

IS

v\_salary f\_emps.salary%TYPE;

BEGIN

SELECT salary INTO v\_salary

FROM f\_emps

WHERE employee\_id = p\_emp\_id;

RETURN v\_salary;

END;

Use the get\_sal function in the following SQL statement (which attempts to move all higher-salaried employees to department 50). What happens and why?

UPDATE f\_emps

SET department\_id = 50

WHERE get\_sal(employee\_id) > 10000;

Intervine eroarea de mai jos din cauza faptului ca incercam sa actualizam department\_id, in timp ce in clauza WHERE avem nevoie de el.

**RA-04091: table RO\_A849\_SQL\_S15.F\_EMPS is mutating, trigger/function may not see it**

11. Examine the following function (which doubles the salary of a chosen employee) and the SQL statement which uses it. What will happen when the SQL statement is executed? Why? Create the upd\_sal function, then run the SELECT statement to confirm your prediction.

CREATE OR REPLACE FUNCTION upd\_sal

(p\_emp\_id f\_emps.employee\_id%TYPE)

RETURN NUMBER

IS

v\_salary f\_emps.salary%TYPE;

BEGIN

SELECT salary INTO v\_salary

FROM f\_emps

WHERE employee\_id = p\_emp\_id; v\_salary := v\_salary \* 2; UPDATE f\_emps

SET salary = v\_salary

WHERE employee\_id = p\_emp\_id;

RETURN v\_salary;

END;

SELECT employee\_id, last\_name, salary, upd\_sal(employee\_id)

FROM f\_emps

WHERE employee\_id = 100;

Nu putem utiliza functia pentru a face o operatie DML in interiorul unei interogari.

ORA-14551: cannot perform a DML operation inside a query