**Упражнение 9**

CREATE PROCEDURE greet IS

today DATE:=SYSDATE;

tomorrow today%TYPE;

BEGIN

tomorrow:=today +1;

DBMS\_OUTPUT.PUT\_LINE(' Hello World ');

DBMS\_OUTPUT.PUT\_LINE('TODAY IS : '|| today);

DBMS\_OUTPUT.PUT\_LINE('TOMORROW IS : ' || tomorrow);

END;

BEGIN

greet;

END;

DROP PROCEDURE greet;

CREATE PROCEDURE greet (p\_name VARCHAR2) IS

today DATE:=SYSDATE;

tomorrow today%TYPE;

BEGIN

tomorrow:=today +1;

DBMS\_OUTPUT.PUT\_LINE(' Hello '|| p\_name);

DBMS\_OUTPUT.PUT\_LINE('TODAY IS : '|| today);

DBMS\_OUTPUT.PUT\_LINE('TOMORROW IS : ' || tomorrow);

END;

BEGIN

greet('Friend');

END;

CREATE OR REPLACE FUNCTION get\_job (p\_jobid IN jobs.job\_id%type)

RETURN jobs.job\_title%type IS

v\_title jobs.job\_title%type;

BEGIN

SELECT job\_title INTO v\_title FROM jobs WHERE job\_id = p\_jobid;

RETURN v\_title;

END get\_job;

VARIABLE b\_title VARCHAR2(35)

EXECUTE :b\_title := get\_job ('SA\_REP');

PRINT b\_title

**Извикайте функцията get\_job с PL/SQL променлива**

CREATE OR REPLACE FUNCTION get\_annual\_comp(

p\_sal IN employees.salary%TYPE,

p\_comm IN employees.commission\_pct%TYPE)

RETURN NUMBER IS

BEGIN

RETURN (NVL(p\_sal,0) \* 12 + (NVL(p\_comm,0) \* NVL(p\_sal,0) \* 12));

END get\_annual\_comp;

SELECT employee\_id, last\_name,

get\_annual\_comp(salary,commission\_pct) "Annual Compensation"

FROM employees

WHERE department\_id=30;

CREATE OR REPLACE FUNCTION valid\_deptid( p\_deptid IN departments.department\_id%TYPE)

RETURN BOOLEAN IS

v\_dummy PLS\_INTEGER;

BEGIN

SELECT 1 INTO v\_dummy FROM departments WHERE department\_id = p\_deptid;

RETURN TRUE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END valid\_deptid;

CREATE OR REPLACE PROCEDURE add\_employee(

p\_first\_name employees.first\_name%TYPE,

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

p\_email employees.email%TYPE,

p\_job employees.job\_id%TYPE DEFAULT 'SA\_REP',

p\_mgr employees.manager\_id%TYPE DEFAULT 145,

p\_sal employees.salary%TYPE DEFAULT 1000,

p\_comm employees.commission\_pct%TYPE DEFAULT 0,

p\_deptid employees.department\_id%TYPE DEFAULT 30) IS

BEGIN

IF valid\_deptid(p\_deptid) THEN

INSERT INTO employees(employee\_id, first\_name, last\_name, email,

job\_id, manager\_id, hire\_date, salary, commission\_pct, department\_id)

VALUES (employees\_seq.NEXTVAL, p\_first\_name, p\_last\_name, p\_email,

p\_job, p\_mgr, TRUNC(SYSDATE), p\_sal, p\_comm, p\_deptid);

ELSE

RAISE\_APPLICATION\_ERROR (-20204, 'Invalid department ID. Try again.');

END IF;

END add\_employee;

EXECUTE add\_employee('Jane', 'Harris', 'JAHARRIS', p\_deptid=> 15)

EXECUTE add\_employee('Joe', 'Harris', 'JAHARRIS', p\_deptid=> 80)

**Променете извикването на add\_employee, така че да се въведе служителя в таблица employees!**

SELECT \* FROM employees WHERE last\_name='Harris';

Изпълнете

**ROLLBACK**

**1 зад.** Напишете PL/SQL функция за извеждане броя на смените на длъжност за даден номер на служител, който е параметър на функцията. Осигурете подходящ текст при извикването на функцията.

**2 зад.** Напишете PL/SQL функция за увеличаване на заплатата с опреден бонус за даден служител и длъжност, като номерът на служителя, бонусът и идентификаторът на длъжността са параметри на функцията. Преди обновяването направете проверка дали новата заплата е между минималната и максимална заплата за дадената длъжност. Осигурете прихващане на грешки и подходящ текст при извикването на функцията.