Blocuri anonime în PL/SQL

Gestiunea variabilelor. Blocuri imbricate

Să se realizeze şi testeze următoarele blocuri anonime în PL/SQL:

1. Afişaţi-vă numele complet;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Numele meu este Tiberiu Claudiu!');

END;

2. Afişaţi data care va fi peste exact 4 luni:

a. Utilizaţi doar o constantă;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Peste 4 luni va fi data de 24 Februarie 2024');

END;

b. Folosiţi o variabilă iniţializată la declarare

DECLARE

    v\_data DATE := ADD\_MONTHS(SYSDATE, 4);

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Peste 4 luni va fi data de '||v\_data);

END;

DECLARE

    v\_data DATE := ADD\_MONTHS(CURRENT\_DATE, 4);

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Peste 4 luni va fi data de '||v\_data);

END;

c. Folosiţi o variabilă calculată în secţiunea de execuţie

DECLARE

    v\_data DATE ;

BEGIN

    v\_data := ADD\_MONTHS(SYSDATE, 4);

    DBMS\_OUTPUT.PUT\_LINE('Peste 4 luni va fi data de '||v\_data);

END;

d. Utilizaţi o interogare

DECLARE

    v\_data DATE ;

BEGIN

    SELECT  ADD\_MONTHS(SYSDATE, 4)

    INTO v\_data

    FROM DUAL;

    DBMS\_OUTPUT.PUT\_LINE('Peste 4 luni va fi data de '||v\_data);

END;

//SQL

SELECT  ADD\_MONTHS(SYSDATE, 4)

FROM DUAL;

3. Sa se defineasca variabile si constante de diferite tipuri, cu diferite optiuni (NOT NULL, DEFAULT), apoi sa se afiseze valoarea lor in cadrul sectiunii de executie. Se vor afisa propozitii inteligibile

DECLARE

    v\_nume VARCHAR2(10) := 'Pop Ion';

    v\_disciplina VARCHAR2(30);

    v\_nota NUMBER := 9;

    v\_data DATE := ADD\_MONTHS(SYSDATE, -3);

BEGIN

    v\_disciplina := 'Informatica Economica';

    DBMS\_OUTPUT.PUT\_LINE('Elevul '||v\_nume||' a obtinut nota '||v\_nota||

                         ' la disciplina '|| v\_disciplina ||' in data de '|| v\_data);

END;

4. Faceţi o copie a tabelei EMPLOYEES sub numele ANGAJATI (în SQL). Analizaţi continutul tabelei, apoi afişaţi numele, prenumele si salariul lunar al Managerului general (în PL/SQL).

// SQL

CREATE TABLE angajati AS SELECT \* FROM employees

SELECT \* FROM angajati

SELECT first\_name, last\_name, salary/12 FROM angajati

WHERE manager\_id IS NULL

// PL/SQL

DECLARE

    v\_nume angajati.last\_name%TYPE;

    v\_prenume angajati.first\_name%TYPE;

    v\_salariu NUMBER(7,2);

BEGIN

    SELECT first\_name, last\_name, salary/12

    INTO  v\_prenume, v\_nume, v\_salariu

    FROM angajati

    WHERE manager\_id IS NULL;

    DBMS\_OUTPUT.PUT\_LINE('Managerul general este '|| v\_nume ||' '||v\_prenume||' si are un venit lunar de '||v\_salariu||' $.');

END;

5. Definiţi variabile/constante de mai multe tipuri de dată. Folosiţi câte o funcţie pentru fiecare variabilă şi afişaţi rezultatul. Mai intai utilizaţi funcţiile în cadrul unor interogări SQL, apoi fără utilizarea clauzei SELECT în cadrul blocului de execuţie

DECLARE

    v\_nume VARCHAR2(20) := 'Popescu';

    v\_prenume VARCHAR2(15) := '   Mihai   ';

    v\_data\_nastere DATE := TO\_DATE('12-11-00', 'dd-mm-yy');

    v\_e\_manager BOOLEAN := false;

    v\_varsta NUMBER(7,2);

    v\_lungime NUMBER(2);

BEGIN

    DBMS\_OUTPUT.PUT\_LINE(

        UPPER(v\_nume)||' ' ||

         TRIM(v\_prenume) ||

         ', cu varsta de ' ||

         TRUNC((SYSDATE - v\_data\_nastere) / 365.25) ||

         ' de ani, a implinit 18 de ani acum '

         || ROUND((SYSDATE - v\_data\_nastere) - (18 \* 365), 2) ||

         ' zile'

    );

    SELECT LENGTH(v\_nume || ' ' || v\_prenume), LPAD(v\_nume, 20, '-') INTO v\_lungime, v\_nume FROM DUAL;

    DBMS\_OUTPUT.PUT\_LINE('Lungimea numelui: ' || v\_lungime || '. Lpadded nume: ' || v\_nume);

END;

6. Realizaţi 3 blocuri imbricate – superior, intermediar şi inferior – iar în cadrul fiecăruia utilizaţi câte o variabilă. Testaţi vizibilitatea şi domeniul acestor variabile.

DECLARE

    v\_nume\_superior VARCHAR2(50) := 'Nume definit in blocul superior';

BEGIN

    DECLARE

        v\_nume\_intermediar VARCHAR2(50) := 'Nume definit in blocul intermediar';

    BEGIN

        DECLARE

            v\_nume\_inferior VARCHAR(50) := 'Nume definit in blocul inferior';

        BEGIN

            DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul inferior]: ' || v\_nume\_superior || ' | ' || v\_nume\_intermediar || ' | ' || v\_nume\_inferior);

        END;

        DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul intermediar]: ' || v\_nume\_superior || ' | ' || v\_nume\_intermediar );

    END;

    DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul superior]: ' || v\_nume\_superior );

END;

7. Realizaţi 3 blocuri imbricate, iar în cadrul fiecăruia utilizaţi variabile cu acelaşi nume, dar valori diferite. Afişaţi în cadrul fiecărui bloc valoarea tuturor variabilelor, calificându-le prin numele blocului doar pe acelea care solicit acest lucru. Pentru a şti din ce bloc s-a făcut afişarea, să se precizeze numele blocului. Ex. “Valori din blocul superior”

<<superior>>

DECLARE

    v\_nume VARCHAR2(50) := 'Nume definit in blocul superior';

BEGIN

    <<intermediar>>

    DECLARE

        v\_nume VARCHAR2(50) := 'nume definit in blocul intermediar';

    BEGIN

        <<inferior>>

        DECLARE

            v\_nume VARCHAR(50) := 'nume definit in blocul inferior';

        BEGIN

            DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul inferior] ' || superior.v\_nume || ' | ' || intermediar.v\_nume || ' | ' || inferior.v\_nume);

        END;

        DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul intermediar]: ' || superior.v\_nume || ' | ' || intermediar.v\_nume);

    END;

    DBMS\_OUTPUT.PUT\_LINE('[Afisari din blocul superior]: ' || superior.v\_nume );

END;

Instrucţiuni SQL în blocuri PL/SQL

Să se realizeze blocuri PL/SQL anonime cu următoarele funcţionalităţi:

1. “Angajaţi-vă” astăzi în cadrul tabelei ANGAJAŢI ca reprezentant de vânzări, cu salariul de 30.000$. Completaţi toate atributele obligatorii şi asiguraţi unicitatea cheii primare

// SQL

INSERT INTO angajati (employee\_id, first\_name, last\_name, email, phone\_number, hire\_date, job\_id, salary, commission\_pct, manager\_id, department\_id)

VALUES (237, 'Dascal', 'Tiberiu', 'tdascal', '123.456.789', TO\_DATE('2023-10-23', 'yyyy-mm-dd'), 'SA\_REP', 30000, null, 100, 80);

// PL/SQL

DECLARE

v\_empl\_id angajati.employee\_id%TYPE;

v\_nume angajati.last\_name%TYPE := 'Dascal';

v\_prenume angajati.first\_name%TYPE:= 'Tiberiu';

v\_email angajati.email%TYPE;

v\_phone angajati.phone\_number%TYPE := '123.456.789';

v\_hire\_date angajati.hire\_date%TYPE := SYSDATE;

v\_job\_id angajati.job\_id%TYPE := 'SA\_REP';

v\_salary angajati.salary%TYPE := 30000;

v\_manager\_id angajati.manager\_id%TYPE := 149;

v\_department\_id angajati.department\_id%TYPE := 80;

BEGIN

SELECT MAX(employee\_id) +1

INTO v\_empl\_id FROM angajati;

v\_email := UPPER(SUBSTR(v\_prenume,1,1) || v\_nume);

INSERT INTO angajati (employee\_id,

first\_name,

last\_name,

email,

phone\_number,

hire\_date,

job\_id,

salary,

commission\_pct,

manager\_id,

department\_id

)

VALUES (

v\_empl\_id,

v\_nume,

v\_prenume,

v\_email,

v\_phone,

v\_hire\_date,

v\_job\_id,

v\_salary,

null,

v\_manager\_id,

v\_department\_id

);

END;

2. Accidental, introduceţi din nou datele Dv., dar cu un alt ID

DECLARE

v\_empl\_id NUMBER := 237;

v\_nume VARCHAR2(50) := 'Tiberiu';

v\_prenume VARCHAR2(50) := 'Dascal';

v\_email VARCHAR(50) := 'tdascal';

v\_phone VARCHAR2(20) := '123.456.789';

v\_hire\_date VARCHAR(10) := '2023-10-23';

v\_job\_id VARCHAR2(10) := 'SA\_REP';

v\_salary NUMBER := 3000;

v\_manager\_id NUMBER :=100;

v\_department\_id NUMBER := 80;

BEGIN

INSERT INTO angajati (employee\_id,

first\_name,

last\_name,

email,

phone\_number,

hire\_date,

job\_id,

salary,

commission\_pct,

manager\_id,

department\_id

)

VALUES (

v\_empl\_id,

v\_nume,

v\_prenume,

v\_email,

v\_phone,

TO\_DATE(v\_hire\_date, 'yyyy-mm-dd'),

v\_job\_id,

v\_salary,

null,

v\_manager\_id,

v\_department\_id

);

END;

3. Ştergeţi cea de-a doua înregistrare adăugată şi creşteţi cu 15% salariul pentru înregistrarea rămasă

DECLARE

v\_empl\_id angajati.employee\_id%TYPE;

BEGIN

SELECT MAX(employee\_id)

INTO v\_empl\_id FROM angajati;

DELETE FROM angajati WHERE employee\_id = v\_empl\_id;

UPDATE angajati SET salary = salary \* 1.15 WHERE employee\_id = v\_empl\_id -1;

END;

SELECT \* FROM angajati ORDER BY employee\_id DESC;

4. Testaţi instrucţiunea MERGE pentru a actualiza tabela EMPLOYEES cu noul angajat introdus

BEGIN

MERGE INTO employees emp

USING angajati ang

ON (emp.employee\_id = ang.employee\_id)

WHEN MATCHED THEN

UPDATE SET emp.first\_name = ang.first\_name,

emp.last\_name = ang.last\_name,

emp.email = ang.email,

emp.phone\_number = ang.phone\_number,

emp.hire\_date = ang.hire\_date,

emp.job\_id = ang.job\_id,

emp.salary = ang.salary,

emp.commission\_pct = ang.commission\_pct,

emp.manager\_id = ang.manager\_id,

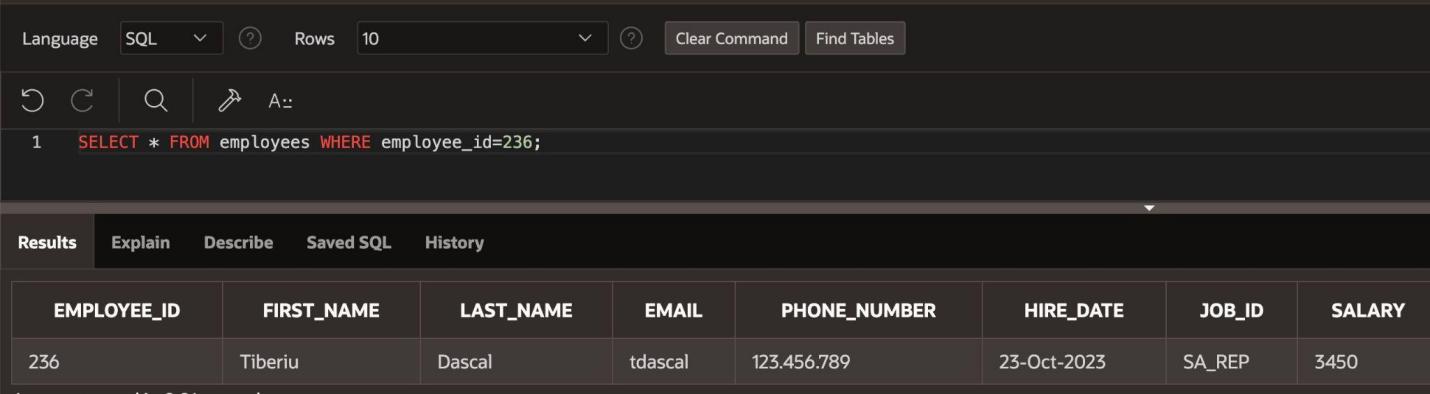
emp.department\_id = ang.department\_id

WHEN NOT MATCHED THEN

INSERT (employee\_id, first\_name, last\_name, email, phone\_number, hire\_date, job\_id, salary, commission\_pct, manager\_id, department\_id)

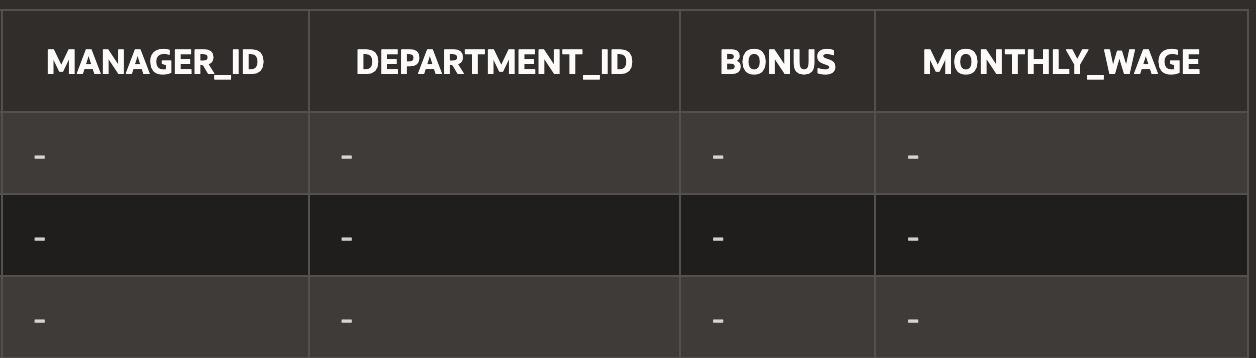
VALUES (ang.employee\_id, ang.first\_name, ang.last\_name, ang.email, ang.phone\_number, ang.hire\_date, ang.job\_id, ang.salary, ang.commission\_pct, ang.manager\_id, ang.department\_id);

END;



5. Modificaţi tabela ANGAJATI prin adaugarea unui nou atribut Monthly\_Wage Number(5)

ALTER TABLE angajati ADD monthly\_wage NUMBER(5);



6. Afişaţi numele, prenumele şi salariul Dv., informaţiile fiind extrase din tabela ANGAJATI

DECLARE

v\_search\_email VARCHAR2(100) := 'tdascal';

v\_nume VARCHAR2(100);

v\_prenume VARCHAR2(100);

v\_salar NUMBER;

BEGIN

SELECT first\_name, last\_name, salary

INTO v\_nume, v\_prenume, v\_salar

FROM angajati WHERE email=v\_search\_email;

DBMS\_OUTPUT.PUT\_LINE('Angajatul '|| v\_nume ||' '|| v\_prenume||' are salariul de '|| v\_salar ||' de RON/luna.');

END;



7. Extrageţi aceleaşi informaţii pentru toţi reprezentanţii de vânzări

DECLARE

v\_search\_criteria VARCHAR2(100) := 'SA\_REP';

BEGIN

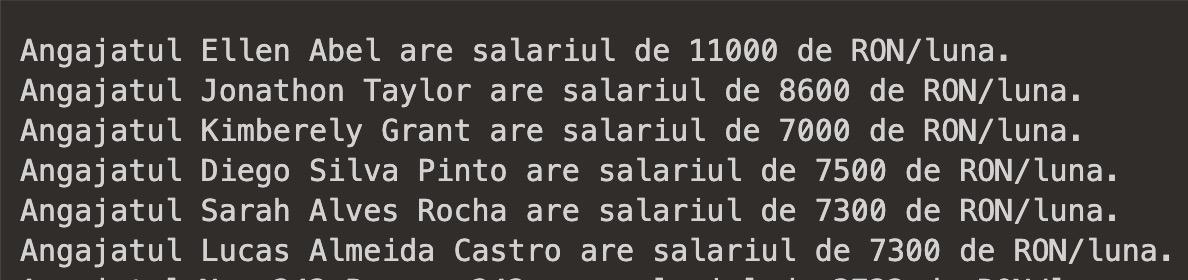
FOR v IN ( SELECT first\_name AS nume, last\_name AS prenume, salary AS salar FROM angajati WHERE job\_id=v\_search\_criteria)

LOOP

DBMS\_OUTPUT.PUT\_LINE('Angajatul '|| v.nume ||' '|| v.prenume||' are salariul de '|| v.salar ||' de RON/luna.');

END LOOP;

END;



8. Actualizaţi cu +10% salariile celor din administraţie şi afişaţi numărul angajaţilor care au fost bonificaţi

DECLARE

v\_counter NUMBER := 0;

BEGIN

FOR v IN (SELECT employee\_id AS admins FROM angajati WHERE job\_id LIKE 'AD\_%')

LOOP

UPDATE angajati SET salary = salary\*1.10 WHERE employee\_id = v.admins;

v\_counter := v\_counter + 1;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(v\_counter|| ' angajati au primit un bonus de 10%');

END;



9. Testaţi instrucţiunile TCL (COMMIT, ROLLBACK, SAVEPOINT) pe tabela ANGAJATI.

BEGIN

-- TESTING TCL Commands

SAVEPOINT my\_sp\_1;

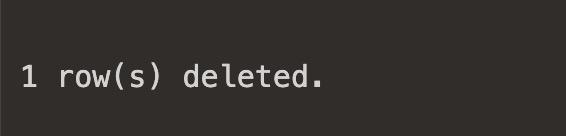
DELETE FROM angajati WHERE email='tdascal';

ROLLBACK to my\_sp\_1;

COMMIT;

END;

SELECT \* FROM angajati WHERE email='tdascal';



Structuri de programare în PL/SQL

Creaţi blocuri anonime pentru rezolvarea următoarelor exerciţii:

1. Testaţi dacă un număr – memorat într-o variabilă – este par;

DECLARE

v\_num\_even NUMBER := 8;

v\_num\_odd NUMBER := 9;

BEGIN

-- test even number

IF MOD(v\_num\_even, 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Numarul '|| v\_num\_even ||' este par!');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Numarul '|| v\_num\_even ||' este impar!');

END IF;

-- test odd number

IF MOD(v\_num\_odd, 2) = 0 THEN

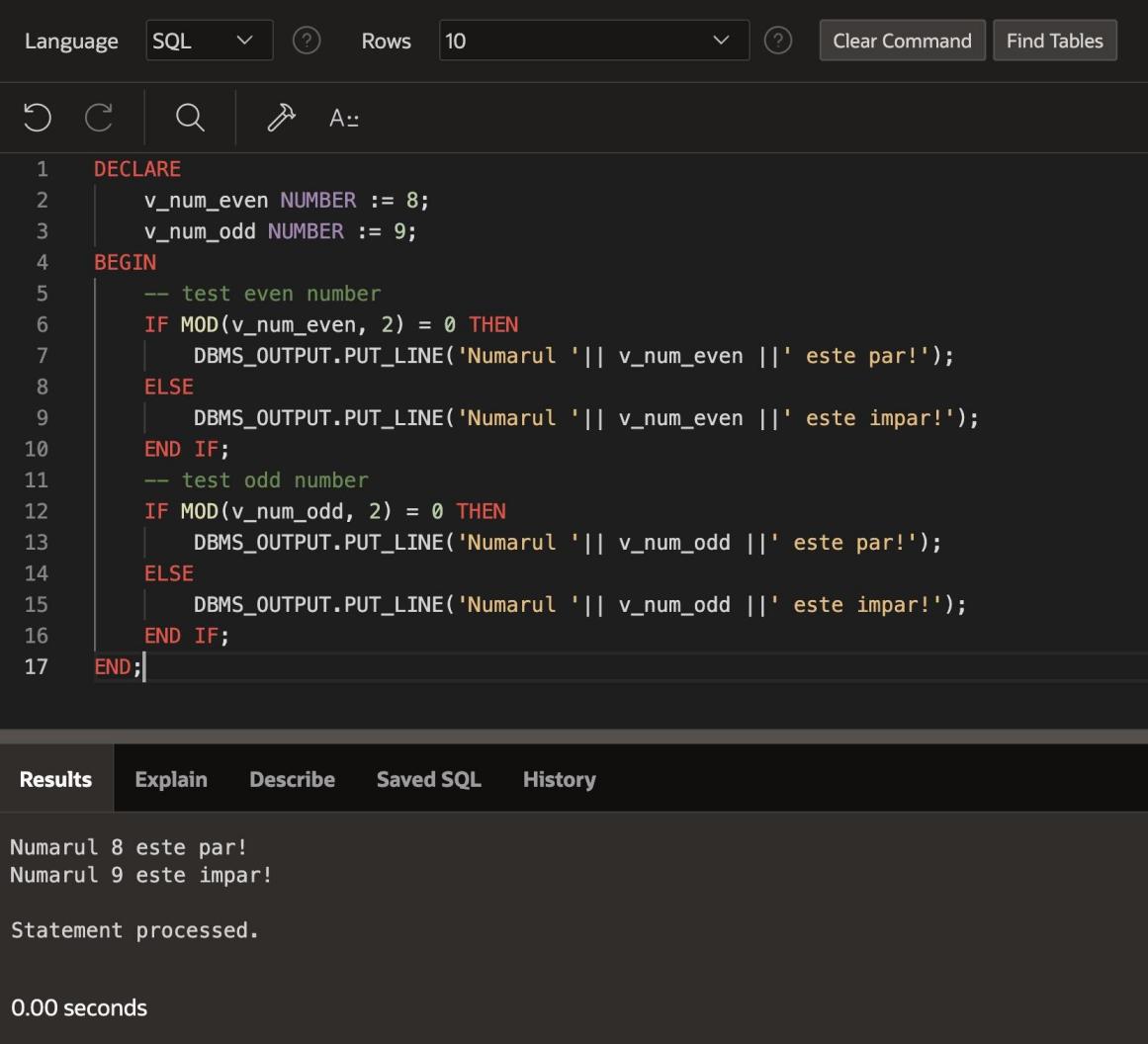
DBMS\_OUTPUT.PUT\_LINE('Numarul '|| v\_num\_odd ||' este par!');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Numarul '|| v\_num\_odd ||' este impar!');

END IF;

END;



2. Se da transcrierea în litere pentru numerele de la 1 la 9. Avem o variabilă care poate conţine un număr întreg între 11 şi 20, reprezentând sume în lei. Afişaţi suma în lei în litere (ca la bancă/poştă), găsind câteva reguli de compunere ale numeralelor şi folosind transcrierile cifrelor de la 1 la 9. Se va rezolva cu structuri IF;

DECLARE

v\_suma NUMBER := 20;

v\_cifra1 NUMBER;

v\_cifra2 NUMBER;

v\_suma\_litere VARCHAR2(100);

BEGIN

v\_cifra1 := FLOOR(v\_suma / 10);

v\_cifra2 := MOD(v\_suma, 10);

IF v\_cifra2 = 1 THEN

v\_suma\_litere := ' unsprezece';

END IF;

IF v\_cifra2 = 2 THEN

v\_suma\_litere := ' doisprezece';

END IF;

IF v\_cifra2 = 3 THEN

v\_suma\_litere := ' treisprezece';

END IF;

IF v\_cifra2 = 4 THEN

v\_suma\_litere := ' paisprezece';

END IF;

IF v\_cifra2 = 5 THEN

v\_suma\_litere := ' cincisprezece';

END IF;

IF v\_cifra2 = 6 THEN

v\_suma\_litere := ' saisprezece';

END IF;

IF v\_cifra2 = 7 THEN

v\_suma\_litere := ' saptesprezece';

END IF;

IF v\_cifra2 = 8 THEN

v\_suma\_litere := ' optsprezece';

END IF;

IF v\_cifra2 = 9 THEN

v\_suma\_litere := ' nouasprezece';

END IF;

IF v\_cifra2 = 2 THEN

v\_suma\_litere := ' doisprezece';

END IF;

-- Afișați suma în litere

DBMS\_OUTPUT.PUT\_LINE('Popescu depoziteaza suma de' ||v\_suma\_litere|| ' RON.');

END;



3. Rezolvaţi aceeaşi problemă cu o structură CASE;

DECLARE

v\_suma NUMBER := 19;

v\_cifra1 NUMBER;

v\_cifra2 NUMBER;

v\_suma\_litere VARCHAR2(100);

BEGIN

v\_cifra1 := FLOOR(v\_suma / 10);

v\_cifra2 := MOD(v\_suma, 10);

CASE v\_cifra2

WHEN 1 THEN v\_suma\_litere := ' unsprezece';

WHEN 2 THEN v\_suma\_litere := ' doisprezece';

WHEN 3 THEN v\_suma\_litere := ' treisprezece';

WHEN 4 THEN v\_suma\_litere := ' paisprezece';

WHEN 5 THEN v\_suma\_litere := ' cincisprezece';

WHEN 6 THEN v\_suma\_litere := ' saisprezece';

WHEN 7 THEN v\_suma\_litere := ' saptesprezece';

WHEN 8 THEN v\_suma\_litere := ' optsprezece';

WHEN 9 THEN v\_suma\_litere := ' nouasprezece';

ELSE v\_suma\_litere := ' douazeci';

END CASE;

-- Afișați suma în litere

DBMS\_OUTPUT.PUT\_LINE('Popescu depoziteaza suma de' ||v\_suma\_litere|| ' RON.');

END;



4. Apoi cu o expresie CASE;

DECLARE

v\_suma NUMBER := 15;

v\_cifra1 NUMBER;

v\_cifra2 NUMBER;

v\_suma\_litere VARCHAR2(100);

BEGIN

v\_cifra1 := FLOOR(v\_suma / 10);

v\_cifra2 := MOD(v\_suma, 10);

v\_suma\_litere :=

CASE v\_cifra2

WHEN 1 THEN ' unsprezece'

WHEN 2 THEN ' doisprezece'

WHEN 3 THEN ' treisprezece'

WHEN 4 THEN ' paisprezece'

WHEN 5 THEN ' cincisprezece'

WHEN 6 THEN ' saisprezece'

WHEN 7 THEN ' saptesprezece'

WHEN 8 THEN ' optsprezece'

WHEN 9 THEN ' nouasprezece'

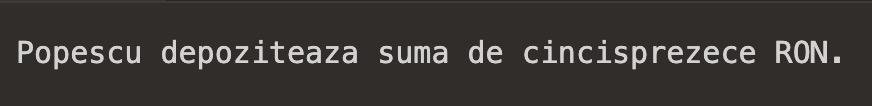
ELSE ' douazeci'

END;

-- Afișați suma în litere

DBMS\_OUTPUT.PUT\_LINE('Popescu depoziteaza suma de' ||v\_suma\_litere|| ' RON.');

END;



5. O persoană s-a născut între anii 1950-2000. În funcţie de anul naşterii să se afişeze mesaje precum: „Domnul X s-a născut în anii ’80.” Testaţi funcţionalitatea secvenţelor ELSIF, în cazul în care le schimbaţi ordinea;

DECLARE

v\_anul\_nasterii NUMBER := 1998;

v\_cifra2 NUMBER;

v\_anii NUMBER;

BEGIN

v\_cifra2 := MOD(MOD(floor(v\_anul\_nasterii / 10), 100), 10);

IF v\_cifra2 = 5 THEN

v\_anii := 50;

ELSIF v\_cifra2 = 6 THEN

v\_anii := 60;

ELSIF v\_cifra2 = 7 THEN

v\_anii := 70;

ELSIF v\_cifra2 = 8 THEN

v\_anii := 80;

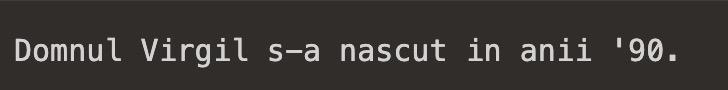
ELSE

v\_anii := 90;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Domnul Virgil s-a nascut in anii '''|| v\_anii||'.');

END;



6. Calculaţi factorialul unui număr memorat într-o variabilă. Se vor folosi toate variantele de structuri iterative cunoscute;

DECLARE

v\_start NUMBER;

v\_numar NUMBER;

v\_factorial NUMBER;

BEGIN

v\_start := 1;

v\_numar := 10;

v\_factorial := 1;

LOOP

v\_factorial := v\_start \* v\_factorial;

v\_start := v\_start + 1;

IF v\_start = v\_numar + 1 THEN

EXIT;

END IF;

END LOOP;

-- star = 1; fact = 2;

DBMS\_OUTPUT.PUT\_LINE(v\_numar||'! este egal cu '||v\_factorial);

END;



DECLARE

v\_start NUMBER;

v\_numar NUMBER;

v\_factorial NUMBER;

BEGIN

v\_start := 1;

v\_numar := 5;

v\_factorial := 1;

WHILE v\_start <= v\_numar LOOP

v\_factorial := v\_start \* v\_factorial;

v\_start := v\_start + 1;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(v\_numar||'! este egal cu '||v\_factorial);

END;



DECLARE

v\_numar NUMBER;

v\_factorial NUMBER;

BEGIN

v\_numar := 8;

v\_factorial := 1;

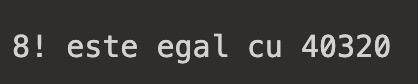
FOR i IN 1..v\_numar LOOP

v\_factorial := i \* v\_factorial;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(v\_numar||'! este egal cu '||v\_factorial);

END;



7. Să se introducă în cadrul tabelei Angajati 4 înregistrări: Id-urile să fie următoarele 4 numere luate secvenţial, numele şi prenumele vor fi generice „Nume”/”Prenume” concatenat cu id-ul. Toţi au fost angajaţi acum o lună ca reprezentanţi de vânzări cu salariul minim al acestei categorii, căruia i se adaugă un spor generat aleatoriu între 200 şi 500$.

DECLARE

v\_id NUMBER;

v\_nume VARCHAR2(50);

v\_prenume VARCHAR2(50);

v\_data\_angajare DATE;

v\_salariu NUMBER;

v\_spor NUMBER;

v\_job\_type VARCHAR2(7) := 'SA\_REP';

v\_email VARCHAR2(100);

v\_start\_id NUMBER;

v\_end\_id NUMBER;

BEGIN

SELECT MAX(employee\_id) INTO v\_start\_id FROM angajati;

v\_start\_id := v\_start\_id + 1;

v\_end\_id := v\_start\_id + 4;

FOR i IN v\_start\_id..v\_end\_id LOOP

v\_id := i;

v\_nume := 'Nume' || i;

v\_prenume := 'Prenume' || i;

v\_email := v\_nume || '.' || v\_prenume;

v\_data\_angajare := ADD\_MONTHS(SYSDATE, -1);

SELECT MIN(salary) INTO v\_salariu FROM angajati WHERE job\_id=v\_job\_type;

v\_spor := ROUND(DBMS\_RANDOM.VALUE(200, 500));

INSERT INTO angajati (employee\_id, first\_name, last\_name, hire\_date, salary, email, job\_id)

VALUES (v\_id, v\_nume, v\_prenume, v\_data\_angajare, v\_salariu + v\_spor, v\_email, v\_job\_type);

END LOOP;

COMMIT;

END;

