**Tema 4**

**Opran Andrei 242**

-- Definiți un tip colecție denumit tip\_orase\_\*\*\*. Creați tabelul excursie\_\*\*\* cu următoarea

-- structură: cod\_excursie NUMBER(4), denumire VARCHAR2(20), orase tip\_orase\_\*\*\* (ce va

-- conține lista orașelor care se vizitează într-o excursie, într-o ordine stabilită; de exemplu, primul

-- oraș din listă va fi primul oraș vizitat), status (disponibilă sau anulată).

-- c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.

-- d. Pentru fiecare excursie afișați lista orașelor vizitate.

-- e. Anulați excursiile cu cele mai puține orașe vizitate.

DECLARE

    TYPE tip\_orase\_ao IS TABLE OF VARCHAR(256);

    TYPE excursie\_ao IS RECORD(

        cod\_excursie NUMBER(4),

        denumire VARCHAR2(20),

        orase tip\_orase\_ao,

        status BOOLEAN DEFAULT TRUE -- TRUE = DISPONIBILA, FALSE = ANULATA

    );

    TYPE tabel\_excursie\_ao IS TABLE OF excursie\_ao INDEX BY BINARY\_INTEGER;

    t tabel\_excursie\_ao;

    cod\_excursie NUMBER(4) := &cod\_tastatura;

    exista\_excursie BOOLEAN := FALSE;

    nu\_exista\_excursie EXCEPTION;

    minim NUMBER(4) := 9999;

BEGIN

    t(1).cod\_excursie := 1;

    t(1).denumire := 'Excursie 1';

    t(1).orase := tip\_orase\_ao('Bucuresti', 'Cluj-Napoca', 'Iasi');

    t(1).status := TRUE;

    t(2).cod\_excursie := 2;

    t(2).denumire := 'Excursie 2';

    t(2).orase := tip\_orase\_ao('Constanta', 'Timisoara', 'Brasov', 'Craiova');

    t(2).status := TRUE;

    t(3).cod\_excursie := 3;

    t(3).denumire := 'Excursie 3';

    t(3).orase := tip\_orase\_ao('Galati', 'Oradea');

    t(3).status := TRUE;

    t(4).cod\_excursie := 4;

    t(4).denumire := 'Excursie 4';

    t(4).orase := tip\_orase\_ao('Ploiesti', 'Campina', 'Busteni');

    t(4).status := TRUE;

    t(5).cod\_excursie := 5;

    t(5).denumire := 'Excursie 5';

    t(5).orase := tip\_orase\_ao('Bacau', 'Focsani');

    t(5).status := TRUE;

    -- c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.

    -- verificare existenta excursie cu codul dat de la tastatura

    FOR i IN 1..t.COUNT LOOP

        IF t(i).cod\_excursie = cod\_excursie THEN

            exista\_excursie := TRUE;

            EXIT;

        END IF;

    END LOOP;

    IF NOT exista\_excursie THEN

        RAISE nu\_exista\_excursie;

    ELSE

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA ' || cod\_excursie || ' ARE ' || t(cod\_excursie).orase.COUNT || ' ORASE VIZITATE:');

        DBMS\_OUTPUT.PUT('ORASELE VIZITATE SUNT: ');

        DBMS\_OUTPUT.PUT\_LINE(' ');

        FOR i IN 1..t(cod\_excursie).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE(t(cod\_excursie).orase(i));

        END LOOP;

    END IF;

    -- d. Pentru fiecare excursie afișați lista orașelor vizitate.

    IF t.COUNT = 0 then

        DBMS\_OUTPUT.PUT\_LINE('NU EXISTA EXCURSII.');

    ELSE

        FOR i in 1..t.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE('LISTA DE ORASE VIZITATE PENTRU EXCURSIA CU CODUL ' || t(i).cod\_excursie || ':');

            FOR j in 1..t(i).orase.COUNT LOOP

                DBMS\_OUTPUT.PUT(t(i).orase(j));

                DBMS\_OUTPUT.PUT(' ');

            END LOOP;

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

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

        END LOOP;

    END IF;

    -- e. Anulați excursiile cu cele mai puține orașe vizitate.

    -- determinare numar minim de orase vizitate

    FOR i in 1..t.COUNT LOOP

        IF t(i).orase.COUNT < minim THEN

            minim := t(i).orase.COUNT;

        END IF;

    END LOOP;

    -- anulare excursii cu numar minim de orase vizitate

    FOR i in 1..t.COUNT LOOP

        IF t(i).orase.COUNT = minim THEN

            t(i).status := FALSE;

        END IF;

    END LOOP;

    -- afisare

    FOR i IN 1..t.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA CU CODUL ' || t(i).cod\_excursie || ':');

        DBMS\_OUTPUT.PUT\_LINE('DENUMIRE: ' || t(i).denumire);

        DBMS\_OUTPUT.PUT('ORASE: ');

        FOR j IN 1..t(i).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT(t(i).orase(j) || ' ');

        END LOOP;

        DBMS\_OUTPUT.NEW\_LINE;

        DBMS\_OUTPUT.PUT\_LINE('STATUS: ' || CASE WHEN t(i).status THEN 'Disponibila' ELSE 'Anulata' END);

        DBMS\_OUTPUT.PUT\_line(' ');

    END LOOP;

EXCEPTION

    WHEN nu\_exista\_excursie THEN

        DBMS\_OUTPUT.PUT\_LINE('Excursia cu codul ' || cod\_excursie || ' nu exista.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || SQLERRM || ' ' || SQLCODE);

END;

/

    -- c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.

    -- verificare existenta excursie cu codul dat de la tastatura

    FOR i IN 1..t.COUNT LOOP

        IF t(i).cod\_excursie = cod\_excursie THEN

            exista\_excursie := TRUE;

            EXIT;

        END IF;

    END LOOP;

    IF NOT exista\_excursie THEN

        RAISE nu\_exista\_excursie;

    ELSE

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA ' || cod\_excursie || ' ARE ' || t(cod\_excursie).orase.COUNT || ' ORASE VIZITATE:');

        DBMS\_OUTPUT.PUT('ORASELE VIZITATE SUNT: ');

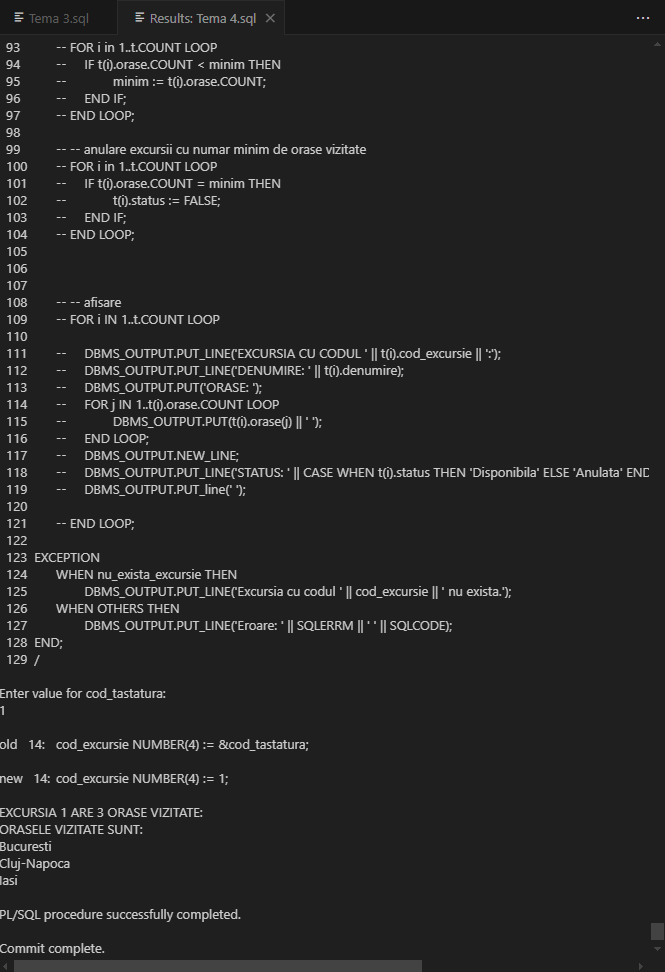
        DBMS\_OUTPUT.PUT\_LINE(' ');

        FOR i IN 1..t(cod\_excursie).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE(t(cod\_excursie).orase(i));

        END LOOP;

    END IF;

****

    -- d. Pentru fiecare excursie afișați lista orașelor vizitate.

    IF t.COUNT = 0 then

        DBMS\_OUTPUT.PUT\_LINE('NU EXISTA EXCURSII.');

    ELSE

        FOR i in 1..t.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE('LISTA DE ORASE VIZITATE PENTRU EXCURSIA CU CODUL ' || t(i).cod\_excursie || ':');

            FOR j in 1..t(i).orase.COUNT LOOP

                DBMS\_OUTPUT.PUT(t(i).orase(j));

                DBMS\_OUTPUT.PUT(' ');

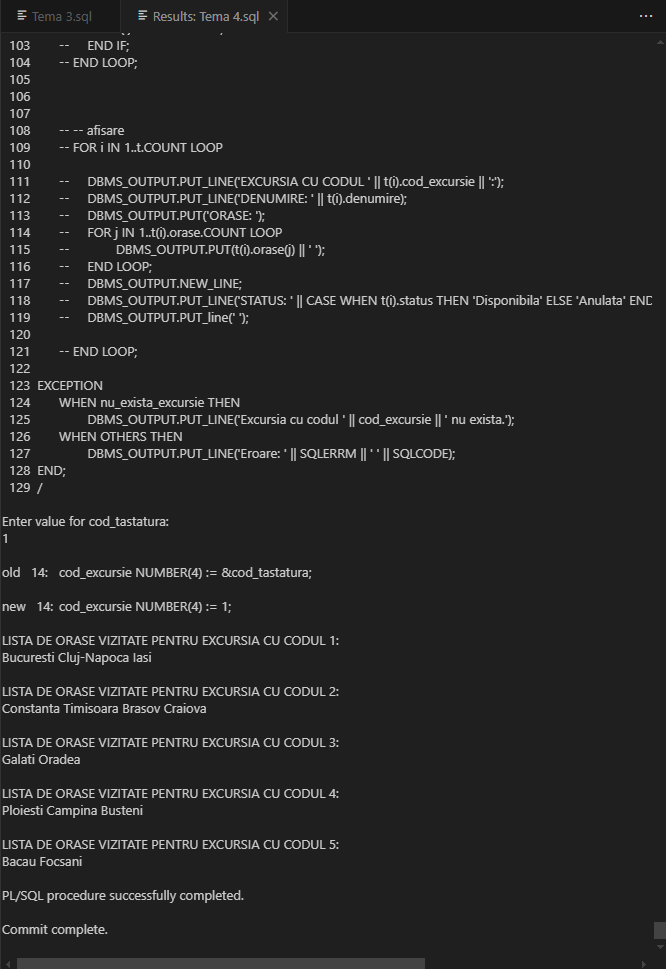
            END LOOP;

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

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

        END LOOP;

    END IF;

****

-- e. Anulați excursiile cu cele mai puține orașe vizitate.

    -- determinare numar minim de orase vizitate

    FOR i in 1..t.COUNT LOOP

        IF t(i).orase.COUNT < minim THEN

            minim := t(i).orase.COUNT;

        END IF;

    END LOOP;

    -- anulare excursii cu numar minim de orase vizitate

    FOR i in 1..t.COUNT LOOP

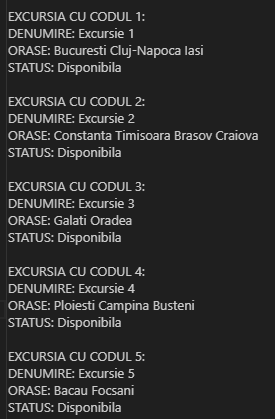
        IF t(i).orase.COUNT = minim THEN

            t(i).status := FALSE;

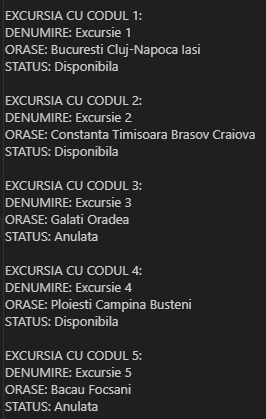
        END IF;

    END LOOP;

**INAINTE DE RULARE:**

****

**DUPA RULARE:**

****

-- E3. Rezolvați problema anterioară folosind un alt tip de colecție studiat

-- a. Inserați 5 înregistrări în tabel.

-- b. Actualizați coloana orase pentru o excursie specificată:

-- - adăugați un oraș nou în listă, ce va fi ultimul vizitat în excursia respectivă;

-- - adăugați un oraș nou în listă, ce va fi al doilea oraș vizitat în excursia respectivă;

-- - inversați ordinea de vizitare a două dintre orașe al căror nume este specificat;

-- - eliminați din listă un oraș al cărui nume este specificat.

-- c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele

-- orașelor.

-- d. Pentru fiecare excursie afișați lista orașelor vizitate.

-- e. Anulați excursiile cu cele mai puține orașe vizitate.

DECLARE

    TYPE varray\_tip\_orase\_ao IS VARRAY(20) OF VARCHAR(256);

    TYPE excursie\_ao IS RECORD(

        cod\_excursie NUMBER(4),

        denumire VARCHAR2(20),

        orase varray\_tip\_orase\_ao,

        status BOOLEAN DEFAULT TRUE -- TRUE = DISPONIBILA, FALSE = ANULATA

    );

    TYPE varray\_excursie\_ao IS VARRAY(20) OF excursie\_ao;

    v varray\_excursie\_ao := varray\_excursie\_ao();

    -- b. variabile

    cod\_excursie NUMBER(4) := &varray\_cod\_tastatura;

    oras\_nou\_1 varchar2(20) := '&varray\_primul\_oras\_nou';

    oras\_nou\_2 varchar2(20) := '&varray\_al\_doilea\_oras\_nou';

    inversare\_1 varchar2(20) := '&varray\_inversare\_1';

    inversare\_2 varchar2(20) := '&varray\_inversare\_2';

    index\_inversare\_1 NUMBER(2) := -1;

    index\_inversare\_2 NUMBER(2) := -1;

    ok1 BOOLEAN := FALSE;

    ok2 BOOLEAN := FALSE;

    eroare\_inversare EXCEPTION;

    oras\_de\_sters varchar2(20) := '&varray\_oras\_de\_sters';

    ok3 BOOLEAN := FALSE;

    eroare\_stergere EXCEPTION;

    -- c. variabile

    exista\_excursie BOOLEAN := FALSE;

    nu\_exista\_excursie EXCEPTION;

    -- e. variabile

    minim NUMBER(4) := 9999;

BEGIN

    -- a. Inserați 5 înregistrări în tabel.

    v.EXTEND(5);

    v(1).cod\_excursie := 1;

    v(1).denumire := 'Excursie 1';

    v(1).orase := varray\_tip\_orase\_ao('Bucuresti', 'Cluj-Napoca', 'Iasi');

    v(1).status := TRUE;

    v(2).cod\_excursie := 2;

    v(2).denumire := 'Excursie 2';

    v(2).orase := varray\_tip\_orase\_ao('Constanta', 'Timisoara', 'Brasov', 'Craiova');

    v(2).status := TRUE;

    v(3).cod\_excursie := 3;

    v(3).denumire := 'Excursie 3';

    v(3).orase := varray\_tip\_orase\_ao('Galati', 'Oradea');

    v(3).status := TRUE;

    v(4).cod\_excursie := 4;

    v(4).denumire := 'Excursie 4';

    v(4).orase := varray\_tip\_orase\_ao('Ploiesti', 'Campina', 'Busteni');

    v(4).status := TRUE;

    v(5).cod\_excursie := 5;

    v(5).denumire := 'Excursie 5';

    v(5).orase := varray\_tip\_orase\_ao('Bacau', 'Focsani');

    v(5).status := TRUE;

    -- b. Actualizați coloana orase pentru o excursie specificată:

    -- - adăugați un oraș nou în listă, ce va fi ultimul vizitat în excursia respectivă;

    v(cod\_excursie).orase.EXTEND;

    v(cod\_excursie).orase(v(cod\_excursie).orase.LAST) := oras\_nou\_1;

    -- - adăugați un oraș nou în listă, ce va fi al doilea oraș vizitat în excursia respectivă;

    v(cod\_excursie).orase.EXTEND;

    v(cod\_excursie).orase(v(cod\_excursie).orase.LAST) := oras\_nou\_2;

    -- - inversați ordinea de vizitare a două dintre orașe al căror nume este specificat;

    FOR i in 1..v(cod\_excursie).orase.COUNT LOOP

        IF upper(v(cod\_excursie).orase(i)) = upper(inversare\_1) THEN

            ok1 := TRUE;

            index\_inversare\_1 := i; -- salvam doar index-ul pentru ca daca modificam direct cum am facut la tema 3, se poate ca inversarea sa nu se faca pana la final

        ELSIF upper(v(cod\_excursie).orase(i)) = upper(inversare\_2) THEN

            ok2 := TRUE;

            index\_inversare\_2 := i; -- salvam doar index-ul pentru ca daca modificam direct cum am facut la tema 3, se poate ca inversarea sa nu se faca pana la final

        END IF;

    END LOOP;

    IF ok1 AND ok2 THEN

        v(cod\_excursie).orase(index\_inversare\_1) := inversare\_2; -- facem inversarea folosind indexii memorati

        v(cod\_excursie).orase(index\_inversare\_2) := inversare\_1;

    ELSE

        RAISE eroare\_inversare;

    END IF;

    - eliminați din listă un oraș al cărui nume este specificat.

    -- verificare existenta oras dat de la tastatura

    FOR i in 1..v.COUNT LOOP

        IF upper(v(cod\_excursie).orase(i)) = upper(oras\_de\_sters) THEN

            ok3 := TRUE;

            FOR j in i..v(cod\_excursie).orase.COUNT-1 LOOP

                v(cod\_excursie).orase(j) := v(cod\_excursie).orase(j+1);

            END LOOP;

            v(cod\_excursie).orase.TRIM;

            EXIT;

        END IF;

    END LOOP;

    IF NOT ok3 THEN

        RAISE eroare\_stergere;

    END IF;

    c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.

    -- verificare existenta excursie cu codul dat de la tastatura

    FOR i IN 1..v.COUNT LOOP

        IF v(i).cod\_excursie = cod\_excursie THEN

            exista\_excursie := TRUE;

            EXIT;

        END IF;

    END LOOP;

    IF NOT exista\_excursie THEN

        RAISE nu\_exista\_excursie;

    ELSE

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA ' || cod\_excursie || ' ARE ' || v(cod\_excursie).orase.COUNT || ' ORASE VIZITATE:');

        DBMS\_OUTPUT.PUT('ORASELE VIZITATE SUNT: ');

        DBMS\_OUTPUT.PUT\_LINE(' ');

        FOR i IN 1..v(cod\_excursie).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE(v(cod\_excursie).orase(i));

        END LOOP;

    END IF;

    -- d. Pentru fiecare excursie afișați lista orașelor vizitate.

    IF v.COUNT = 0 then

        DBMS\_OUTPUT.PUT\_LINE('NU EXISTA EXCURSII.');

    ELSE

        FOR i in 1..v.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE('LISTA DE ORASE VIZITATE PENTRU EXCURSIA CU CODUL ' || v(i).cod\_excursie || ':');

            FOR j in 1..v(i).orase.COUNT LOOP

                DBMS\_OUTPUT.PUT(v(i).orase(j));

                DBMS\_OUTPUT.PUT(' ');

            END LOOP;

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

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

        END LOOP;

    END IF;

    -- e. Anulați excursiile cu cele mai puține orașe vizitate.

    -- determinare numar minim de orase vizitate

    FOR i in 1..v.COUNT LOOP

        IF v(i).orase.COUNT < minim THEN

            minim := v(i).orase.COUNT;

        END IF;

    END LOOP;

    -- anulare excursii cu numar minim de orase vizitate

    FOR i in 1..v.COUNT LOOP

        IF v(i).orase.COUNT = minim THEN

            v(i).status := FALSE;

        END IF;

    END LOOP;

    -- afisare excursii

    FOR i IN 1..v.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA CU CODUL ' || v(i).cod\_excursie || ':');

        DBMS\_OUTPUT.PUT\_LINE('DENUMIRE: ' || v(i).denumire);

        DBMS\_OUTPUT.PUT('ORASE: ');

        FOR j IN 1..v(i).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT(v(i).orase(j) || ' ');

        END LOOP;

        DBMS\_OUTPUT.NEW\_LINE;

        DBMS\_OUTPUT.PUT\_LINE('STATUS: ' || CASE WHEN v(i).status THEN 'Disponibila' ELSE 'Anulata' END);

        DBMS\_OUTPUT.PUT\_LINE(' ');

    END LOOP;

EXCEPTION

EXCEPTION

    WHEN eroare\_inversare THEN

        DBMS\_OUTPUT.PUT\_LINE('Nu s-a putut efectua inversarea deoarece nu exista orasele specificate in excursia cu codul ' || cod\_excursie);

    WHEN eroare\_stergere THEN

        DBMS\_OUTPUT.PUT\_LINE('Nu s-a putut efectua stergerea deoarece nu exista orasul specificat in excursia cu codul ' || cod\_excursie);

    WHEN nu\_exista\_excursie THEN

        DBMS\_OUTPUT.PUT\_LINE('Excursia cu codul ' || cod\_excursie || ' nu exista.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || SQLERRM || ' ' || SQLCODE);

END;

/

    WHEN nu\_exista\_excursie THEN

        DBMS\_OUTPUT.PUT\_LINE('Excursia cu codul ' || cod\_excursie || ' nu exista.');

    WHEN OTHERS THEN

        DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || SQLERRM || ' ' || SQLCODE);

END;

/

-- b. Actualizați coloana orase pentru o excursie specificată:

    -- - adăugați un oraș nou în listă, ce va fi ultimul vizitat în excursia respectivă;

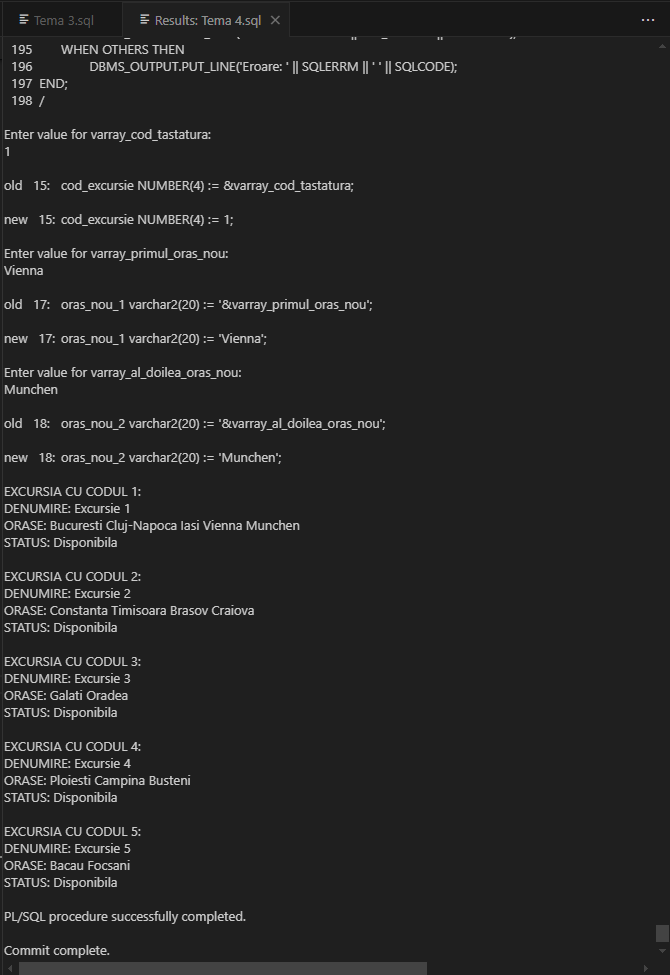
    v(cod\_excursie).orase.EXTEND;

    v(cod\_excursie).orase(v(cod\_excursie).orase.LAST) := oras\_nou\_1;

    -- - adăugați un oraș nou în listă, ce va fi al doilea oraș vizitat în excursia respectivă;

    v(cod\_excursie).orase.EXTEND;

    v(cod\_excursie).orase(v(cod\_excursie).orase.LAST) := oras\_nou\_2;

****

    -- - inversați ordinea de vizitare a două dintre orașe al căror nume este specificat;

    FOR i in 1..v(cod\_excursie).orase.COUNT LOOP

        IF upper(v(cod\_excursie).orase(i)) = upper(inversare\_1) THEN

            ok1 := TRUE;

            index\_inversare\_1 := i; -- salvam doar index-ul pentru ca daca modificam direct cum am facut la tema 3, se poate ca inversarea sa nu se faca pana la final

        ELSIF upper(v(cod\_excursie).orase(i)) = upper(inversare\_2) THEN

            ok2 := TRUE;

            index\_inversare\_2 := i; -- salvam doar index-ul pentru ca daca modificam direct cum am facut la tema 3, se poate ca inversarea sa nu se faca pana la final

        END IF;

    END LOOP;

    IF ok1 AND ok2 THEN

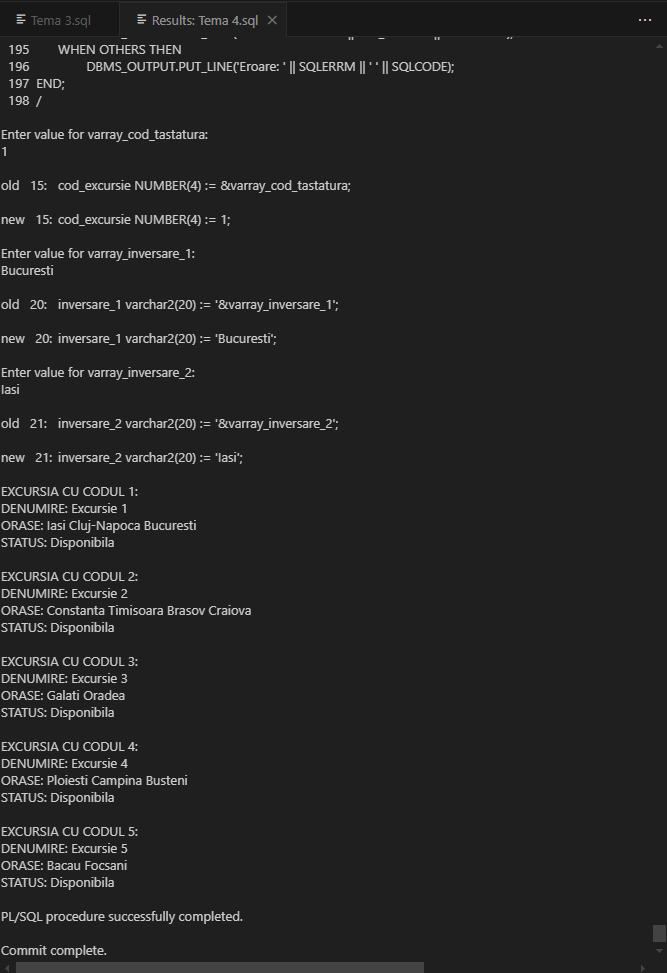
        v(cod\_excursie).orase(index\_inversare\_1) := inversare\_2; -- facem inversarea folosind indexii memorati

        v(cod\_excursie).orase(index\_inversare\_2) := inversare\_1;

    ELSE

        RAISE eroare\_inversare;

    END IF;

****

    -- eliminați din listă un oraș al cărui nume este specificat.

    -- verificare existenta oras dat de la tastatura

    FOR i in 1..v.COUNT LOOP

        IF upper(v(cod\_excursie).orase(i)) = upper(oras\_de\_sters) THEN

            ok3 := TRUE;

            FOR j in i..v(cod\_excursie).orase.COUNT-1 LOOP

                v(cod\_excursie).orase(j) := v(cod\_excursie).orase(j+1);

            END LOOP;

            v(cod\_excursie).orase.TRIM;

            EXIT;

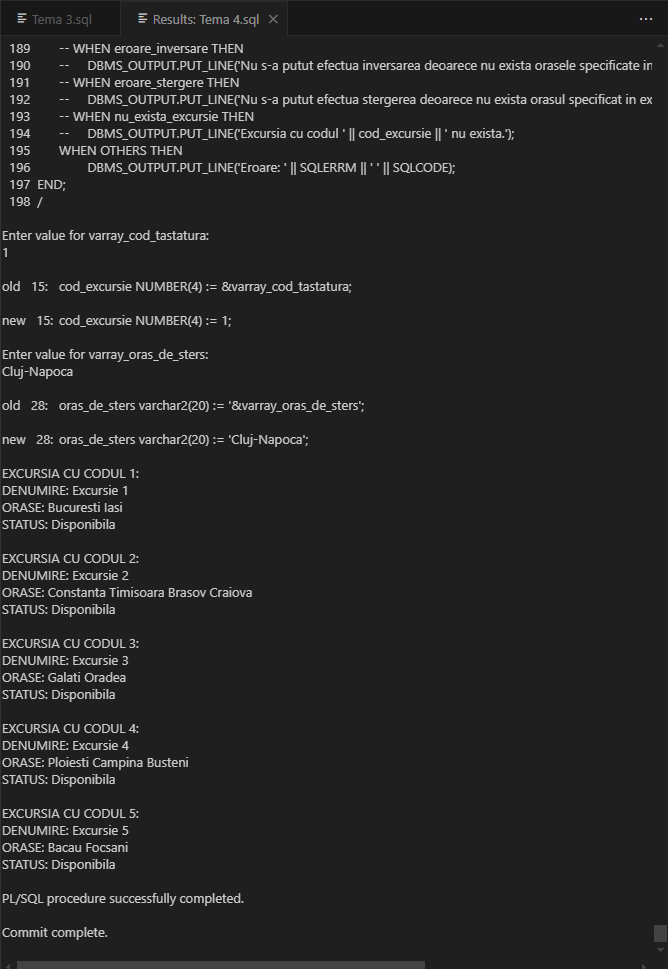
        END IF;

    END LOOP;

    IF NOT ok3 THEN

        RAISE eroare\_stergere;

    END IF;

****

    -- c. Pentru o excursie al cărui cod este dat, afișați numărul de orașe vizitate, respectiv numele orașelor.

    -- verificare existenta excursie cu codul dat de la tastatura

    FOR i IN 1..v.COUNT LOOP

        IF v(i).cod\_excursie = cod\_excursie THEN

            exista\_excursie := TRUE;

            EXIT;

        END IF;

    END LOOP;

    IF NOT exista\_excursie THEN

        RAISE nu\_exista\_excursie;

    ELSE

        DBMS\_OUTPUT.PUT\_LINE('EXCURSIA ' || cod\_excursie || ' ARE ' || v(cod\_excursie).orase.COUNT || ' ORASE VIZITATE:');

        DBMS\_OUTPUT.PUT('ORASELE VIZITATE SUNT: ');

        DBMS\_OUTPUT.PUT\_LINE(' ');

        FOR i IN 1..v(cod\_excursie).orase.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE(v(cod\_excursie).orase(i));

        END LOOP;

    END IF;

****

    -- d. Pentru fiecare excursie afișați lista orașelor vizitate.

    IF v.COUNT = 0 then

        DBMS\_OUTPUT.PUT\_LINE('NU EXISTA EXCURSII.');

    ELSE

        FOR i in 1..v.COUNT LOOP

            DBMS\_OUTPUT.PUT\_LINE('LISTA DE ORASE VIZITATE PENTRU EXCURSIA CU CODUL ' || v(i).cod\_excursie || ':');

            FOR j in 1..v(i).orase.COUNT LOOP

                DBMS\_OUTPUT.PUT(v(i).orase(j));

                DBMS\_OUTPUT.PUT(' ');

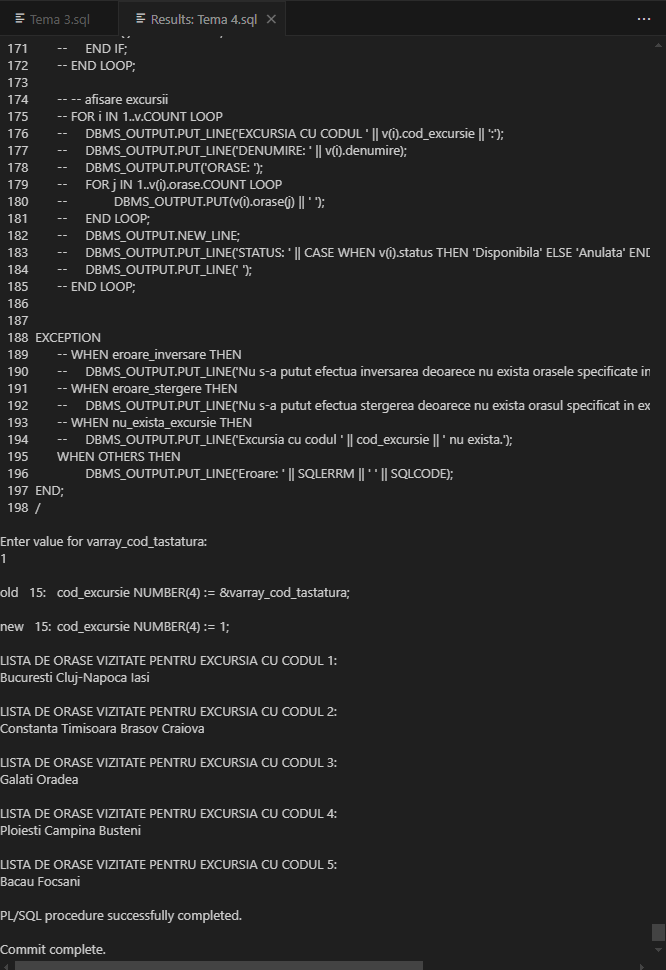
            END LOOP;

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

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

        END LOOP;

    END IF;

****

    -- e. Anulați excursiile cu cele mai puține orașe vizitate.

    -- determinare numar minim de orase vizitate

    FOR i in 1..v.COUNT LOOP

        IF v(i).orase.COUNT < minim THEN

            minim := v(i).orase.COUNT;

        END IF;

    END LOOP;

    -- anulare excursii cu numar minim de orase vizitate

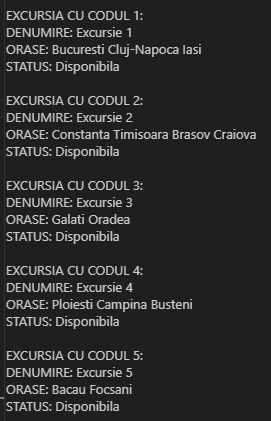
    FOR i in 1..v.COUNT LOOP

        IF v(i).orase.COUNT = minim THEN

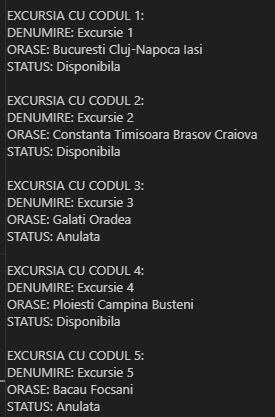
            v(i).status := FALSE;

        END IF;

    END LOOP;

**INAINTE RULARE:  
**

**DUPA RULARE:**

****