







*-- E2. 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).  
-- 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.  
  
-- a*CREATE TYPE lista\_orase IS TABLE OF VARCHAR2(20);  
CREATE TABLE excursie (  
 cod\_excursie NUMBER(4) PRIMARY KEY,  
 denumire VARCHAR2(20),  
 orase lista\_orase,  
 status VARCHAR(20) DEFAULT 'disponibila'  
)  
NESTED TABLE orase STORE AS NESTED\_excursie\_lista\_orase;  
  
INSERT INTO excursie (cod\_excursie, denumire, orase) VALUES (1, 'Excursie 1', lista\_orase('Bucuresti', 'Timisoara'));  
INSERT INTO excursie (cod\_excursie, denumire, orase) VALUES (2, 'Excursie 2', lista\_orase('Craiova', 'Timisoara'));  
INSERT INTO excursie (cod\_excursie, denumire, orase) VALUES (3, 'Excursie 3', lista\_orase('Craiova', 'Cluj'));  
INSERT INTO excursie (cod\_excursie, denumire, orase) VALUES (4, 'Excursie 4', lista\_orase('Bucuresti', 'Cluj'));  
INSERT INTO excursie (cod\_excursie, denumire, orase) VALUES (5, 'Excursie 5', lista\_orase('Craiova', 'Bucuresti', 'Cluj'));  
  
*-- b*DECLARE  
 SUBTYPE cod\_excursie IS excursie.cod\_excursie%TYPE;  
 SUBTYPE oras IS VARCHAR2(20);  
 v\_excursie\_de\_modificat cod\_excursie := &p\_excursive\_de\_modificat;  
 v\_oras\_add\_la\_sfarsit oras := &p\_oras\_add\_la\_sfarsit;  
 v\_oras\_add\_pozitie\_2 oras := &p\_oras\_add\_pozitie\_2;  
 v\_oras\_swap\_1 oras := &p\_oras\_swap\_1;  
 v\_oras\_swap\_2 oras := &p\_oras\_swap\_2;  
 v\_oras\_delete oras := &p\_oras\_delete;  
 v\_lista\_orase lista\_orase;  
 v\_lista\_orase\_crescuta lista\_orase := lista\_orase();  
 v\_oras oras;  
BEGIN  
 SELECT orase  
 INTO v\_lista\_orase  
 FROM excursie  
 WHERE cod\_excursie = v\_excursie\_de\_modificat  
 FOR UPDATE OF orase;  
  
 DBMS\_OUTPUT.*PUT*('Updating: {');  
 FOR i IN 1..v\_lista\_orase.LAST  
 LOOP  
 DBMS\_OUTPUT.*PUT*(' '||v\_lista\_orase(i));  
 END LOOP;  
 DBMS\_OUTPUT.*PUT\_LINE*(' }');  
  
 v\_lista\_orase.EXTEND;  
 v\_lista\_orase(v\_lista\_orase.LAST) := v\_oras\_add\_la\_sfarsit;  
  
 v\_lista\_orase\_crescuta.*EXTEND*(v\_lista\_orase.COUNT + 1);  
 v\_lista\_orase\_crescuta(1) := v\_lista\_orase(1);  
 v\_lista\_orase\_crescuta(2) := v\_oras\_add\_pozitie\_2;  
 FOR i IN 2..v\_lista\_orase.LAST  
 LOOP  
 *-- DataGrip a insistat sa folosesc DECODE, nu CASE* v\_lista\_orase\_crescuta(i + 1) := v\_lista\_orase(i);  
 END LOOP;  
  
 v\_lista\_orase.DELETE;  
 FOR i IN 1..v\_lista\_orase\_crescuta.LAST  
 LOOP  
 CONTINUE WHEN v\_lista\_orase\_crescuta(i) = v\_oras\_delete;  
  
 v\_oras := CASE v\_lista\_orase\_crescuta(i)  
 WHEN v\_oras\_swap\_1 THEN v\_oras\_swap\_2  
 WHEN v\_oras\_swap\_2 THEN v\_oras\_swap\_1  
 ELSE v\_lista\_orase\_crescuta(i)  
 END;  
  
 v\_lista\_orase.EXTEND;  
 v\_lista\_orase(v\_lista\_orase.LAST) := v\_oras;  
 END LOOP;  
  
 UPDATE excursie  
 SET orase = v\_lista\_orase  
 WHERE cod\_excursie = v\_excursie\_de\_modificat;  
  
 DBMS\_OUTPUT.*PUT*('Updated: {');  
 FOR i IN 1..v\_lista\_orase.LAST  
 LOOP  
 DBMS\_OUTPUT.*PUT*(' '||v\_lista\_orase(i));  
 END LOOP;  
 DBMS\_OUTPUT.*PUT\_LINE*(' }');  
EXCEPTION  
 WHEN NO\_DATA\_FOUND THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Cod excursie '||v\_excursie\_de\_modificat||' nu exista');  
END;  
/  
  
DROP TABLE excursie;  
DROP TYPE lista\_orase;  
  
*-- 10. Creați tabelul angajat\_telefoane cu coloanele id\_angajat și nume din tabelul angajat.  
-- Adăugați în acest tabel un nou câmp numit telefon de tip tablou imbricat. Acest tablou va menține  
-- pentru fiecare salariat toate numerele de telefon la care poate fi contactat. Inserați o linie nouă în  
-- tabel. Actualizați o linie din tabel. Afișați informațiile din tabel. Ștergeți tabelul și tipul.*CREATE TYPE lista\_telefoane IS TABLE OF VARCHAR2(20);  
CREATE TABLE angajat\_telefoane AS  
 SELECT id\_angajat, nume FROM ANGAJAT;  
  
ALTER TABLE angajat\_telefoane  
ADD (telefoane lista\_telefoane)  
NESTED TABLE telefoane STORE AS NESTED\_angajat\_telefoane;  
  
INSERT INTO angajat\_telefoane (id\_angajat, nume, telefoane)  
VALUES (999999, 'Angajat Test', lista\_telefoane('+40 123 491 1231', '+1 123 532 4213'));  
  
INSERT INTO TABLE(SELECT telefoane FROM angajat\_telefoane WHERE id\_angajat = 999999)  
VALUES ('+1 919 432 1233');  
  
SELECT id\_angajat, nume, telefoane FROM angajat\_telefoane;  
  
DROP TABLE angajat\_telefoane;  
DROP TYPE lista\_telefoane;