# Funcția stocată f2\_oras\_angjat

|  |
| --- |
| /\*      E7. Adaptați cerințele exercițiilor 2 și 4 (folosind ca bază cerințele exercițiilor 1, respectiv 3) pentru      diagrama proiectului prezentată la materia Baze de Date din anul I. Rezolvați aceste două exerciții      în PL/SQL, folosind baza de date proprie. (PARTIAL doar ex. 2)      1. (adaptat)      Definiți un subprogram prin care să obțineți orașul în care lucrează un angajat, folosind numele angajatului.      Funcția trebuie să funcționeze chiar dacă utilizatorul introduce doar numele de familie sau doar prenumele.      Tratați toate excepțiile ce pot fi generate.      2.      Rezolvați exercițiul 1 folosind o funcție stocată.  \*/  CREATE OR REPLACE FUNCTION f2\_oras\_angajat(v\_nume IN angajat.nume%TYPE)  RETURN oras.nume%TYPE IS      v\_id angajat.id\_angajat%TYPE;      v\_nume\_oras oras.nume%TYPE;      BEGIN          SELECT id\_angajat          INTO v\_id          FROM angajat          WHERE regexp\_like(nume, '(^| )'||v\_nume||'($| )', 'i');          -- Imposibil să avem expectie la join, datorită constrângerilor          -- FOREIGN KEY și NOT NULL pe id\_restaurant și id\_oras.          SELECT oras.nume          INTO v\_nume\_oras          FROM angajat          JOIN restaurant          USING (id\_restaurant)          JOIN oras          USING (id\_oras)          WHERE id\_angajat = v\_id;          RETURN v\_nume\_oras;      EXCEPTION          WHEN NO\_DATA\_FOUND THEN              RAISE\_APPLICATION\_ERROR(-20009, 'Nu s-a găsit un angajat cu nume sau prenume '||v\_nume||'.');              RETURN -1;          WHEN TOO\_MANY\_ROWS THEN              RAISE\_APPLICATION\_ERROR(-20009, 'Input ambiguu: multiplii angajați găsiți cu nume sau prenume '||v\_nume||'.');              RETURN -2;          WHEN OTHERS THEN              RAISE\_APPLICATION\_ERROR(-20009,'Alta eroare: '||SQLERRM);              RETURN -3;  END;  / |

# Găsirea inputurilor ambigue pentru testarea codului

|  |
| --- |
| /\*      Găsește un nume a.î. count >= 1 pentru a cauza eroarea  \*/  WITH fn AS (      SELECT regexp\_substr(nume, '(\w+)$', 1, 1, NULL, 1) AS nume FROM angajat  )  SELECT nume, COUNT(\*) FROM fn  GROUP BY nume  ORDER BY COUNT(\*) DESC;  /\*   Iulia si Anamaria ar trebui să cauzeze eroarea.  \*/ |

# Codul PL/SQL pentru a testa funcția stocată

|  |
| --- |
| DECLARE      TYPE test\_inputs IS VARRAY(10) OF angajat.nume%TYPE;      v\_test\_inputs test\_inputs := test\_inputs('Adrian', 'Iulia', 'Trandafir', 'Trandafi', 'Iorga Irina', 'Anamaria', 'Stefan');      esec\_f2\_oras\_angajat EXCEPTION;      PRAGMA EXCEPTION\_INIT (esec\_f2\_oras\_angajat, -20009);  BEGIN      FOR i IN 1..v\_test\_inputs.LAST      LOOP          dbms\_output.PUT(v\_test\_inputs(i) || ' => ');          BEGIN              dbms\_output.PUT\_LINE('Lucrează în '||f2\_oras\_angajat(v\_test\_inputs(i))||'.');          EXCEPTION              WHEN esec\_f2\_oras\_angajat THEN                  dbms\_output.put\_line(SQLERRM);          END;      END LOOP;  END;  / |

# Întregul fișier 251\_Negrescu\_Theodor\_tema\_lab\_8.sql

|  |
| --- |
| /\*      E7. Adaptați cerințele exercițiilor 2 și 4 (folosind ca bază cerințele exercițiilor 1, respectiv 3) pentru      diagrama proiectului prezentată la materia Baze de Date din anul I. Rezolvați aceste două exerciții      în PL/SQL, folosind baza de date proprie. (PARTIAL doar ex. 2)      1. (adaptat)      Definiți un subprogram prin care să obțineți orașul în care lucrează un angajat, folosind numele angajatului.      Funcția trebuie să funcționeze chiar dacă utilizatorul introduce doar numele de familie sau doar prenumele.      Tratați toate excepțiile ce pot fi generate.      2.      Rezolvați exercițiul 1 folosind o funcție stocată.  \*/  CREATE OR REPLACE FUNCTION f2\_oras\_angajat(v\_nume IN angajat.nume%TYPE)  RETURN oras.nume%TYPE IS      v\_id angajat.id\_angajat%TYPE;      v\_nume\_oras oras.nume%TYPE;      BEGIN          SELECT id\_angajat          INTO v\_id          FROM angajat          WHERE regexp\_like(nume, '(^| )'||v\_nume||'($| )', 'i');          -- Imposibil să avem expectie la join, datorită constrângerilor          -- FOREIGN KEY și NOT NULL pe id\_restaurant și id\_oras.          SELECT oras.nume          INTO v\_nume\_oras          FROM angajat          JOIN restaurant          USING (id\_restaurant)          JOIN oras          USING (id\_oras)          WHERE id\_angajat = v\_id;          RETURN v\_nume\_oras;      EXCEPTION          WHEN NO\_DATA\_FOUND THEN              RAISE\_APPLICATION\_ERROR(-20009, 'Nu s-a găsit un angajat cu nume sau prenume '||v\_nume||'.');              RETURN -1;          WHEN TOO\_MANY\_ROWS THEN              RAISE\_APPLICATION\_ERROR(-20009, 'Input ambiguu: multiplii angajați găsiți cu nume sau prenume '||v\_nume||'.');              RETURN -2;          WHEN OTHERS THEN              RAISE\_APPLICATION\_ERROR(-20009,'Alta eroare: '||SQLERRM);              RETURN -3;  END;  /  /\*      Găsește un nume a.î. count >= 1 pentru a cauza eroarea  \*/  WITH fn AS (      SELECT regexp\_substr(nume, '(\w+)$', 1, 1, NULL, 1) AS nume FROM angajat  )  SELECT nume, COUNT(\*) FROM fn  GROUP BY nume  ORDER BY COUNT(\*) DESC;  /\*   Iulia si Anamaria ar trebui să cauzeze eroarea.  \*/  DECLARE      TYPE test\_inputs IS VARRAY(10) OF angajat.nume%TYPE;      v\_test\_inputs test\_inputs := test\_inputs('Adrian', 'Iulia', 'Trandafir', 'Trandafi', 'Iorga Irina', 'Anamaria', 'Stefan');      esec\_f2\_oras\_angajat EXCEPTION;      PRAGMA EXCEPTION\_INIT (esec\_f2\_oras\_angajat, -20009);  BEGIN      FOR i IN 1..v\_test\_inputs.LAST      LOOP          dbms\_output.PUT(v\_test\_inputs(i) || ' => ');          BEGIN              dbms\_output.PUT\_LINE('Lucrează în '||f2\_oras\_angajat(v\_test\_inputs(i))||'.');          EXCEPTION              WHEN esec\_f2\_oras\_angajat THEN                  dbms\_output.put\_line(SQLERRM);          END;      END LOOP;  END;  /  DROP FUNCTION f2\_oras\_angajat; |

# Screenshot-uri





