|  |
| --- |
| /\*   1. Să se ilustreze printr-un exemplu de program PL/SQL multi-bloc modul de propagare al excepțiilor.  Vor fi ilustrate cel putin situațiile în care o excepție este tratată sau nu în blocul curent și în care controlul programului va fi transmis blocului următor din secvență sau blocului exterior.  \*/  BEGIN      BEGIN          BEGIN              BEGIN                  BEGIN                      BEGIN                          BEGIN                              dbms\_output.PUT\_LINE('Împărțirea la zero ridică o excepție predefinită TOO\_MANY\_ROWS.');                              dbms\_output.PUT\_LINE(1 / 0);                          END;                      END;                  END;              EXCEPTION                  WHEN TOO\_MANY\_ROWS THEN                      dbms\_output.PUT\_LINE('Nu se va rula acest handler deoarece excepția nu este de tipul TOO\_MANY\_ROWS. Excepția se va propaga în blocurile părinte.');              END;          END;      END;  EXCEPTION      WHEN ZERO\_DIVIDE THEN          dbms\_output.PUT\_LINE('Excepțiile se progagă până la primul handler potrivit, indiferent de adâncimea imbricării.');  END;  /  BEGIN      DECLARE          v\_data BINARY\_INTEGER;      BEGIN          dbms\_output.PUT\_LINE('Excepția predefinită NO\_DATA\_FOUND este ridicată când un SELECT INTO returnează 0 rânduri.');          SELECT 1          INTO v\_data          FROM DUAL          WHERE rownum = 2;      EXCEPTION          WHEN NO\_DATA\_FOUND THEN              dbms\_output.PUT\_LINE('După ce o excepție este tratată, execuția iese din bloc. Codul din afara blocului nu este afectat de excepții ridicate și tratate în bloc.');      END;      DECLARE          v\_data BINARY\_INTEGER;      BEGIN          dbms\_output.PUT\_LINE('Acest text va fi printat deoarece excepția NO\_DATA\_FOUND a fost tratată, deci nu va afecta acest bloc.');          SELECT 1          INTO v\_data          FROM DUAL          UNION          SELECT 2          FROM DUAL;      EXCEPTION          WHEN NO\_DATA\_FOUND THEN              dbms\_output.PUT\_LINE('Nu se va printa acest text deoarece excepția nu este NO\_DATA\_FOUND. Următorul handler va fi rulat deoarece excepția este TOO\_MANY\_ROWS.');          WHEN TOO\_MANY\_ROWS THEN              dbms\_output.PUT\_LINE('Excepția predefinită TOO\_MANY\_ROWS este ridicată când un SELECT INTO returnează multiple rânduri.');      END;  END;  /  /\*   2. Să se ilustreze prin exemple folosirea instrucțiunii RAISE pentru a ridica atât o excepție predefinită cât și o excepție definită de utilizator.  În cazul excepțiilor predefinite, să se explice cum anume folosirea instrucțiunii RAISE schimbă funcționalitatea programului (față de cazul când această instrucțiune nu există).  \*/  BEGIN      RAISE\_APPLICATION\_ERROR(-20000, 'RAISE\_APPLICATION\_ERROR folosește pentru a ridica excepții cu un cod și mesaj personalizat.');  EXCEPTION      WHEN OTHERS THEN          dbms\_output.PUT\_LINE('Orice se prinde cu WHEN OTHERS, chiar și un cod care nu are o excepție asociată. SQLCODE = '||SQLCODE||', SQLERRM = '||SQLERRM);  END;  /  DECLARE      custom\_exception EXCEPTION;      custom\_exception\_with\_custom\_code EXCEPTION;      row\_overflow EXCEPTION;      too\_many\_varray\_values EXCEPTION;      PRAGMA EXCEPTION\_INIT (custom\_exception\_with\_custom\_code, -20000);      PRAGMA EXCEPTION\_INIT (row\_overflow, -01422);      PRAGMA EXCEPTION\_INIT (too\_many\_varray\_values, -22165);  BEGIN      BEGIN          dbms\_output.PUT\_LINE('Expecțiile definite de utilizatori se definesc similar cu variabilele.');          dbms\_output.PUT\_LINE('Se poate asocia un cod de eroare folosing PRAGMA EXCEPTION\_INIT');          RAISE custom\_exception\_with\_custom\_code;      EXCEPTION          WHEN custom\_exception\_with\_custom\_code THEN              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;      BEGIN          RAISE\_APPLICATION\_ERROR(-20000, 'Dacă excepția are un cod asociat, se poate folosi RAISE\_APPLICATION\_ERROR pentru a ridica eroarea, și WHEN o va prinde.');      EXCEPTION          WHEN custom\_exception\_with\_custom\_code THEN              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;      BEGIN          dbms\_output.PUT\_LINE('Implicit, excepțiile definite de utilizator nu au un cod asociat.');          RAISE custom\_exception;      EXCEPTION          WHEN custom\_exception THEN              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;      BEGIN          BEGIN              dbms\_output.PUT\_LINE(1 / 0);          EXCEPTION              WHEN ZERO\_DIVIDE THEN                  dbms\_output.PUT\_LINE('Excepțiile pot fi re-ridicate pentru a continua propagarea. Se poate folosi pentru a raporta detalii despre o eroare, apoi a trimite excepția mai sus.');                  RAISE;          END;          dbms\_output.PUT\_LINE('Deși excepția a fost tratată, nu se va printa acest text deoarece a fost re-ridicată.');      EXCEPTION          WHEN ZERO\_DIVIDE THEN              dbms\_output.PUT\_LINE('Excepția va fi propagată, la fel ca o excepție netratată.');      END;      DECLARE          v\_data BINARY\_INTEGER;      BEGIN          SELECT 1          INTO v\_data          FROM DUAL          UNION          SELECT 2          FROM DUAL;      EXCEPTION          WHEN row\_overflow THEN              dbms\_output.PUT\_LINE('Se poate defini un nume nou pentru excepțiile predefinite.');              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;      DECLARE          TYPE int\_varray IS VARRAY(10) OF BINARY\_INTEGER;          array int\_varray := int\_varray();      BEGIN          SELECT LEVEL          BULK COLLECT INTO array          FROM DUAL          CONNECT BY LEVEL <= 11;      EXCEPTION          WHEN too\_many\_varray\_values THEN              dbms\_output.PUT\_LINE('Unele excepții predefinite nu au un nume asociat predefinit. Totuși, se poate defini un nume folosing EXCEPTION\_INIT.');              dbms\_output.PUT\_LINE('Când o excepție predefinită este ridicată automat, se generează un mesaj de eroare în limba utilizatorului.');              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;      BEGIN          RAISE too\_many\_varray\_values;      EXCEPTION          WHEN too\_many\_varray\_values THEN              dbms\_output.PUT\_LINE('Când se folosește RAISE pentru a ridica o excepție predefinită, mesajul de eroare nu se va crea corect.');              dbms\_output.PUT\_LINE('SQLCODE: '||SQLCODE||' SQLERRM:'||SQLERRM);      END;  END;  / |

