

Tratarea erorilor

Mecanismul de gestiune a erorilor permite utilizatorului să definească și să controleze comportamentul programului atunci când acesta generează o eroare. În acest fel, aplicația nu este oprită, revenind într-un regim normal de execuție.

Într-un program *PL/SQL* pot să apară erori la compilare sau erori la execuție.

Erorile care apar în timpul compilării sunt detectate de motorul *PL/SQL* și sunt comunicate programatorului care va face corecția acestora. Programul nu poate trata aceste erori deoarece nu a fost încă executat.

Erorile care apar în timpul execuției nu mai sunt tratate interactiv. În program trebuie prevăzută apariția unei astfel de erori și specificat modul concret de tratare a acesteia. Atunci când apare eroarea este declanșată o excepție, iar controlul trece la o secțiune separată a programului, unde va avea loc tratarea erorii.

Gestiunea erorilor în *PL/SQL* face referire la conceptul de excepție. Excepția este un eveniment particular (eroare sau avertisment) generat de *server-ul Oracle* sau de aplicație, care necesită o tratare specială. În *PL/SQL* mecanismul de tratare a excepțiilor permite programului să își continue execuția și în prezența anumitor erori.

Excepțiile pot fi definite, activate, tratate la nivelul fiecărui bloc din program (program principal, funcții și proceduri, blocuri interioare acestora). Execuția unui bloc se termină întotdeauna atunci când apare o excepție, dar se pot executa acțiuni ulterioare apariției acesteia, într-o secțiune specială de tratare a excepțiilor.

Posibilitatea de a da nume fiecărei excepții, de a izola tratarea erorilor într-o secțiune particulară, de a declanșa automat erori (în cazul excepțiilor interne) îmbunătățește lizibilitatea și fiabilitatea programului. Prin utilizarea excepțiilor și rutinelor de tratare a excepțiilor, un program *PL/SQL* devine robust și capabil să trateze atât erorile așteptate, cât și cele neașteptate ce pot apărea în timpul execuției.

Secțiunea de tratare a erorilor

Pentru a gestiona excepțiile, utilizatorul trebuie să scrie câteva comenzi care preiau controlul derulării blocului *PL/SQL*. Aceste comenzi sunt situate în secțiunea de tratare a erorilor dintr-un bloc *PL/SQL* și sunt cuprinse între cuvintele cheie *EXCEPTION* și *END*, conform următoarei sintaxe generale:

EXCEPTION

```

WHEN nume_excepție1 [OR nume_excepție2 ...] THEN
    secvența_de_instrucțiuni_1;
[WHEN nume_excepție3 [OR nume_excepție4 ...] THEN
    secvența_de_instrucțiuni_2;]
...
[WHEN OTHERS THEN
    secvența_de_instrucțiuni_n;]
END;

```

De remarcat că *WHEN OTHERS* trebuie să fie ultima clauză și trebuie să fie unică. Toate excepțiile care nu au fost analizate vor fi tratate prin această clauză. Evident, în practică nu se utilizează forma *WHEN OTHERS THEN NULL*.

În *PL/SQL* există două tipuri de excepții:

- excepții interne, care se produc atunci când un bloc *PL/SQL* nu respectă o regulă *Oracle* sau depășește o limită a sistemului de operare;
- excepții externe definite de utilizator (*user-defined error*), care sunt declarate în secțiunea declarativă a unui bloc, subprogram sau pachet și care sunt activate explicit în partea executabilă a blocului *PL/SQL*.

Excepțiile interne *PL/SQL* sunt de două tipuri:

- excepții interne predefinite (*predefined Oracle Server error*);
- excepții interne nepredefinite (*non-predefined Oracle Server error*).

Funcții pentru identificarea excepțiilor

Indiferent de tipul excepției, aceasta are asociate două elemente:

- un cod care o identifică;
- un mesaj cu ajutorul căruia se poate interpreta excepția respectivă.

Cu ajutorul funcțiilor *SQLCODE* și *SQLERRM* se pot obține codul și mesajul asociate excepției declanșate. Lungimea maximă a mesajului este de 512 caractere.

De exemplu, pentru eroarea predefinită *ZERO_DIVIDE*, codul *SQLCODE* asociat este -1476, iar mesajul corespunzător erorii, furnizat de *SQLERRM*, este „divide by zero error“.

Codul erorii este:

- un număr negativ, în cazul unei erori sistem;
- numărul +100, în cazul excepției *NO_DATA_FOUND*;
- numărul 0, în cazul unei execuții normale (fără excepții);
- numărul 1, în cazul unei excepții definite de utilizator.

Funcțiile *SQLCODE* și *SQLERRM* nu se pot utiliza direct ca parte a unei instrucțiuni *SQL*. Valorile acestora trebuie atribuite unor variabile locale.

Rezultatul funcției *SQLCODE* poate fi asignat unei variabile de tip numeric, iar cel al funcției *SQLERRM* unei variabile de tip caracter. Variabilele locale astfel definite pot fi utilizate în comenzi *SQL*.

Exemplu:

Să se scrie un bloc *PL/SQL* prin care să se exemplifice situația comentată.

```
DECLARE
    eroare_cod      NUMBER;
    eroare_mesaj    VARCHAR2(100);
BEGIN
    ...
EXCEPTION
    ...
    WHEN OTHERS THEN
        eroare_cod := SQLCODE;
        eroare_mesaj := SUBSTR(SQLERRM,1,100);
        INSERT INTO erori
        VALUES (eroare_cod, eroare_mesaj);
END;
```

Mesajul asociat excepției declanșate poate fi furnizat și de funcția *DBMS_UTILITY.FORMAT_ERROR_STACK*.

Excepții interne

Excepțiile interne se produc atunci când un bloc *PL/SQL* nu respectă o regulă *Oracle* sau depășește o limită a sistemului de exploatare.

Aceste excepții pot fi independente de structura bazei de date sau pot să apară datorită nerespectării constrângerilor statice implementate în structură (*PRIMARY KEY*, *FOREIGN KEY*, *NOT NULL*, *UNIQUE*, *CHECK*).

Atunci când apare o eroare *Oracle*, excepția asociată ei se declanșează implicit. De exemplu, dacă apare eroarea *ORA-01403* (deoarece o comandă *SELECT* nu returnează nici o linie), atunci implicit *PL/SQL* activează excepția *NO_DATA_FOUND*. Cu toate că fiecare astfel de excepție are asociat un cod specific, ele trebuie referite prin nume.

Excepții interne predefinite

Excepțiile interne predefinite nu trebuie declarate în secțiunea declarativă și sunt tratate implicit de către *server-ul Oracle*. Ele sunt referite prin nume (*CURSOR_ALREADY_OPEN*, *DUP_VAL_ON_INDEX*, *NO_DATA_FOUND* etc.). *PL/SQL* declară aceste excepții în pachetul *DBMS_STANDARD*.

Nume excepție	Cod eroare	Descriere
<i>ACCES_INT0_NULL</i>	<i>ORA-06530</i>	Asignare de valori atributelor unui obiect neinițializat.
<i>CASE_NOT_FOUND</i>	<i>ORA-06592</i>	Nu este selectată nici una din clauzele <i>WHEN</i> ale lui <i>CASE</i> și nu există nici clauza <i>ELSE</i> (excepție specifică lui <i>Oracle9i</i>).
<i>COLLECTION_IS_NULL</i>	<i>ORA-06531</i>	Aplicarea unei metode (diferite de <i>EXISTS</i>) unui tabel imbricat sau unui vector neinițializat.
<i>CURSOR_ALREADY_OPEN</i>	<i>ORA-06511</i>	Deschiderea unui cursor care este deja deschis.
<i>DUP_VAL_ON_INDEX</i>	<i>ORA-00001</i>	Detectarea unei dubluri într-o coloană unde acestea sunt interzise.
<i>INVALID_CURSOR</i>	<i>ORA-01001</i>	Operație ilegală asupra unui cursor.
<i>INVALID_NUMBER</i>	<i>ORA-01722</i>	Conversie nepermisă de la tipul șir de caractere la număr.
<i>LOGIN_DENIED</i>	<i>ORA-01017</i>	Nume sau parolă incorecte.
<i>NO_DATA_FOUND</i>	<i>ORA-01403</i>	Comanda <i>SELECT</i> nu returnează nici o înregistrare.
<i>NOT_LOGGED_ON</i>	<i>ORA-01012</i>	Programul <i>PL/SQL</i> apelează baza fără să fie conectat la <i>Oracle</i> .
<i>SELF_IS_NULL</i>	<i>ORA-30625</i>	Apelul unei metode când instanța este <i>NULL</i> .
<i>PROGRAM_ERROR</i>	<i>ORA-06501</i>	<i>PL/SQL</i> are o problemă internă.
<i>ROWTYPE_MISMATCH</i>	<i>ORA-06504</i>	Incompatibilitate între parametrii actuali și formali, la deschiderea unui cursor parametrizat.
<i>STORAGE_ERROR</i>	<i>ORA-06500</i>	<i>PL/SQL</i> are probleme cu spațiul de memorie.
<i>SUBSCRIPT_BEYOND_COUNT</i>	<i>ORA-06533</i>	Referire la o componentă a unui <i>nested table</i> sau <i>varray</i> , folosind un index mai mare decât numărul elementelor colecției respective.
<i>SUBSCRIPT_OUTSIDE_LIMIT</i>	<i>ORA-06532</i>	Referire la o componentă a unui tabel imbricat sau vector, folosind un index care este în afara domeniului (de exemplu, -1).
<i>SYS_INVALID_ROWID</i>	<i>ORA-01410</i>	Conversia unui șir de caractere într-un <i>ROWID</i> nu se poate face deoarece șirul nu reprezintă un <i>ROWID</i> valid.
<i>TIMEOUT_ON_RESOURCE</i>	<i>ORA-00051</i>	Expirarea timpului de așteptare pentru eliberarea unei resurse.
<i>TRANSACTION_BACKED_OUT</i>	<i>ORA-00061</i>	Tranzacția a fost anulată datorită unei interblocări.
<i>TOO_MANY_ROWS</i>	<i>ORA-01422</i>	<i>SELECT...INTO</i> întoarce mai multe linii.
<i>VALUE_ERROR</i>	<i>ORA-06502</i>	Apariția unor erori în conversii, constrângeri sau erori aritmetice.
<i>ZERO_DIVIDE</i>	<i>ORA-01476</i>	Sesizarea unei împărțiri la zero.

Exemplu:

Să se scrie un bloc *PL/SQL* prin care să se afișeze numele artiștilor de o anumită naționalitate care au opere de artă expuse în muzeu.

- 1) Dacă rezultatul interogării returnează mai mult decât o linie, atunci să se trateze excepția și să se insereze în tabelul *mesaje* textul „mai mulți creatori”.
- 2) Dacă rezultatul interogării nu returnează nici o linie, atunci să se trateze excepția și să se insereze în tabelul *mesaje* textul „nici un creator”.
- 3) Dacă rezultatul interogării este o singură linie, atunci să se insereze în tabelul *mesaje* numele artistului și pseudonimul acestuia.
- 4) Să se trateze orice altă eroare, inserând în tabelul *mesaje* textul „alte erori au apărut”.

```
SET VERIFY OFF
ACCEPT national PROMPT 'Introduceti nationalitatea:'
DECLARE
    v_num_artist    artist.nume%TYPE;
    v_pseudonim     artist.pseudonim%TYPE;
    v_national      artist.national%TYPE:='&national';
BEGIN
    SELECT    nume, pseudonim
    INTO      v_num_artist, v_pseudonim
    FROM      artist
    WHERE     national = v_national;
    INSERT    INTO mesaje (rezultate)
    VALUES   (v_num_artist||'-'||v_pseudonim);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        INSERT INTO mesaje (rezultate)
        VALUES ('nici un creator');
    WHEN TOO_MANY_ROWS THEN
        INSERT INTO mesaje (rezultate)
        VALUES ('mai multi creatori');
    WHEN OTHERS THEN
        INSERT INTO mesaje (rezultate)
        VALUES ('alte erori au aparut');
END;
/
SET VERIFY ON
```

Aceeași excepție poate să apară în diferite circumstanțe. De exemplu, excepția *NO_DATA_FOUND* poate fi generată fie pentru că o interogare nu întoarce un rezultat, fie pentru că se referă un element al unui tablou *PL/SQL* care nu a fost definit (nu are atribuită o valoare). Dacă într-un bloc *PL/SQL* apar ambele situații, este greu de stabilit care dintre ele a generat eroarea și este necesară restructurarea blocului, astfel încât acesta să poată diferenția cele două situații.

Excepții interne nepredefinite

Excepțiile interne nepredefinite sunt declarate în secțiunea declarativă și sunt tratate implicit de către *server-ul Oracle*. Ele pot fi gestionate prin clauza *OTHERS*, în secțiunea *EXCEPTION*.

Diferențierea acestor erori este posibilă doar cu ajutorul codului. După cum s-a mai specificat, codul unei excepții interne este un număr negativ, în afară de excepția *NO_DATA_FOUND*, care are codul +100.

O altă metodă pentru tratarea unei erori interne nepredefinite (diferită de folosirea clauzei *OTHERS* drept detector universal de excepții) este utilizarea directivei de compilare (pseudo-instrucțiune) *PRAGMA EXCEPTION_INIT*. Această directivă permite asocierea numelui unei excepții cu un cod de eroare intern. În felul acesta, orice excepție internă poate fi referită printr-un nume și se pot scrie rutine speciale pentru tratarea acesteia. Directiva este procesată în momentul compilării, și nu la execuție.

Directiva trebuie să apară în partea declarativă a unui bloc, pachet sau subprogram, după definirea numelui excepției. *PRAGMA EXCEPTION_INIT* poate să apară de mai multe ori într-un program. De asemenea, pot fi asignate mai multe nume pentru același cod de eroare.

În acest caz, tratarea erorii se face în următoarea manieră:

- 1) se declară numele excepției în partea declarativă sub forma:

nume_excepție **EXCEPTION**;

- 2) se asociază numele excepției cu un cod eroare standard *Oracle*, utilizând comanda:

PRAGMA EXCEPTION_INIT (*nume_excepție*, *cod_eroare*);

- 3) se referă excepția în secțiunea de gestiune a erorilor (excepția este tratată automat, fără a fi necesară comanda *RAISE*).

Exemplu:

Dacă există opere de artă create de un anumit artist, să se tipărească un mesaj prin care utilizatorul este anunțat că artistul respectiv nu poate fi șters din baza de date (violarea constrângerii de integritate având codul eroare *Oracle* - 2292).

```
SET VERIFY OFF
DEFINE p_nume = Monet
DECLARE
    opera_exista  EXCEPTION;
    PRAGMA EXCEPTION_INIT(opera_exista,-2292);
BEGIN
    DELETE FROM artist WHERE nume = '&p_nume';
```

```

COMMIT;
EXCEPTION
  WHEN opera_exista THEN
    DBMS_OUTPUT.PUT_LINE ('nu puteti sterge artistul cu
      numele ' || '&p_nume' || ' deoarece exista in
      muzeu opere de arta create de acesta');
END;
/
SET VERIFY ON

```

Excepții externe

PL/SQL permite utilizatorului să definească propriile sale excepții. Aceste excepții pot să apară în toate secțiunile unui bloc, subprogram sau pachet. Excepțiile externe sunt definite în partea declarativă a blocului, deci posibilitatea de referire la ele este asigurată. În mod implicit, toate excepțiile externe au asociat același cod (+1) și același mesaj (*USER DEFINED EXCEPTION*).

Tratarea unei astfel de erori se face într-o manieră similară modului de tratare descris anterior. Activarea excepției externe este făcută explicit, folosind comanda *RAISE* însoțită de numele excepției. Comanda oprește execuția normală a blocului *PL/SQL* și transferă controlul „administratorului” excepțiilor.

Declararea și prelucrarea excepțiilor externe respectă următoarea sintaxă:

```

DECLARE
  nume_excepție EXCEPTION; -- declarare excepție
BEGIN
  ...
  RAISE nume_excepție; --declanșare excepție
  -- codul care urmează nu mai este executat
  ...
EXCEPTION
  WHEN nume_excepție THEN
  -- definire mod de tratare a erorii
  ...
END;

```

Excepțiile trebuie privite ca niște variabile, în sensul că ele sunt active în secțiunea în care sunt declarate. Ele nu pot să apară în instrucțiuni de atribuire sau în comenzi *SQL*.

Este recomandat ca fiecare subprogram să aibă definită o zonă de tratare a excepțiilor. Dacă pe parcursul execuției programului intervine o eroare, atunci acesta generează o excepție și controlul se transferă blocului de tratare a erorilor.

Exemplu:

Să se scrie un bloc *PL/SQL* care afișează numărul creatorilor operelor de artă din muzeu care au valoarea mai mare sau mai mică cu 100000\$ decât o valoare specificată. Să se tipărească un mesaj adecvat, dacă nu există nici un artist care îndeplinește această condiție.

```
VARIABLE g_mesaj VARCHAR2(100)
SET VERIFY OFF
ACCEPT p_val PROMPT 'va rog specificati valoarea:'
DECLARE
    v_val          opera.valoare%TYPE := &p_val;
    v_inf          opera.valoare%TYPE := v_val - 100000;
    v_sup          opera.valoare%TYPE := v_val + 100000;
    v_numar        NUMBER(7);
    e_nimeni       EXCEPTION;
    e_mai_mult     EXCEPTION;
BEGIN
    SELECT  COUNT(DISTINCT cod_autor)
    INTO    v_numar
    FROM    opera
    WHERE   valoare BETWEEN v_inf AND v_sup;
    IF v_numar = 0 THEN
        RAISE e_nimeni;
    ELSIF v_numar > 0 THEN
        RAISE e_mai_mult;
    END IF;
EXCEPTION
    WHEN e_nimeni THEN
        :g_mesaj:='nu exista nici un artist cu valoarea
                operelor cuprinsa intre '||v_inf||' si '||v_sup;
    WHEN e_mai_mult THEN
        :g_mesaj:='exista '||v_numar||' artisti cu valoarea
                operelor cuprinsa intre '||v_inf||' si '||v_sup;
    WHEN OTHERS THEN
        :g_mesaj:='au aparut alte erori';
END;
/

SET VERIFY ON
PRINT g_mesaj
```

Activarea unei excepții externe poate fi făcută și cu ajutorul procedurii *RAISE_APPLICATION_ERROR*, furnizată de pachetul *DBMS_STANDARD*.

RAISE_APPLICATION_ERROR poate fi folosită pentru a returna un mesaj de eroare unității care o apelează, mesaj mai descriptiv (non standard) decât identificatorul erorii. Unitatea apelantă poate fi *SQL*Plus*, un subprogram *PL/SQL* sau o aplicație *client*.

Procedura are următorul antet:

RAISE_APPLICATION_ERROR (*numar_eroare* **IN NUMBER**,
mesaj_eroare **IN VARCHAR2**, [{**TRUE** / **FALSE**}]);

Atributul *numar_eroare* este un număr cuprins între -20000 și -20999, specificat de utilizator pentru excepția respectivă, iar *mesaj_eroare* este un text asociat erorii, care poate avea maximum 2048 octeți.

Parametrul boolean este opțional. Dacă acest parametru este *TRUE*, atunci noua eroare se va adăuga listei erorilor existente, iar dacă este *FALSE* (valoare implicită) atunci noua eroare va înlocui lista curentă a erorilor (se retine ultimul mesaj de eroare).

O aplicație poate apela *RAISE_APPLICATION_ERROR* numai dintr-un subprogram stocat (sau metodă). Dacă *RAISE_APPLICATION_ERROR* este apelată, atunci subprogramul se termină și sunt returnate codul și mesajul asociate erorii respective.

Procedura *RAISE_APPLICATION_ERROR* poate fi folosită în secțiunea executabilă, în secțiunea de tratare a erorilor și chiar simultan în ambele secțiuni.

- În secțiunea executabilă:

```
DELETE FROM opera WHERE material = 'carton';
IF SQL%NOTFOUND THEN
    RAISE_APPLICATION_ERROR(-20201,'info incorecta');
END IF;
```

- În secțiunea de tratare a erorilor:

```
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20202,'info invalida');
END;
```

- În ambele secțiuni:

```
DECLARE
    e_material EXCEPTION;
    PRAGMA EXCEPTION_INIT (e_material, -20777);
BEGIN
    ...
    DELETE FROM opera WHERE valoare < 100001;
    IF SQL%NOTFOUND THEN
        RAISE_APPLICATION_ERROR(-20777,
            'nu exista opera cu aceasta valoare');
    END IF;
EXCEPTION
    WHEN e_material THEN
        -- trateaza eroarea aceasta
    ...
END;
```

RAISE_APPLICATION_ERROR facilitează comunicarea dintre *client* și *server*, transmițând aplicației *client* erori specifice aplicației de pe *server* (de obicei, un declanșator). Prin urmare, procedura este doar un mecanism folosit pentru comunicarea *server* → *client* a unei erori definite de utilizator, care permite ca procesul *client* să trateze excepția.

Exemplu:

Să se implementeze un declanșator care nu permite acceptarea în muzeu a operelor de artă având valoarea mai mică de 100000\$.

```
CREATE OR REPLACE TRIGGER minim_valoare
BEFORE INSERT ON opera
FOR EACH ROW
BEGIN
    IF :NEW.valoare < 100000 THEN
        RAISE_APPLICATION_ERROR
            (-20005, 'operele de arta trebuie sa aiba valoare
                mai mare de 100000$');
    END IF;
END;
```

Pe stația *client* poate fi scris un program care detectează și tratează eroarea.

```
DECLARE
    /* declarare excepție */
    nu_accepta EXCEPTION;
    /* asociază nume, codului eroare folosit in trigger */
    PRAGMA EXCEPTION_INIT(nu_accepta, -20005);
BEGIN
    /* incercă sa inserezi */
    INSERT INTO opera ...;
EXCEPTION
    /* tratare excepție */
    WHEN nu_accepta THEN
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
    /* SQLERRM va returna mesaj din RAISE_APPLICATION_ERROR */
END;
```

Cazuri speciale în tratarea excepțiilor

Dacă se declanșează o excepție într-un bloc simplu, atunci se face saltul la partea de tratare (*handler*) a acesteia, iar după ce este terminată tratarea erorii se iese din bloc (instrucțiunea *END*).

Prin urmare, dacă excepția se propagă spre blocul care include blocul curent, restul acțiunilor executabile din subbloc sunt „pierdute“. Dacă după o eroare se dorește totuși continuarea prelucrării datelor, este suficient ca instrucțiunea care a declanșat excepția să fie inclusă într-un subbloc.

După ce subblocul a fost terminat, se continuă secvența de instrucțiuni din blocul principal.

Exemplu:

```
BEGIN
  DELETE ...
  SELECT ...--poate declansa exceptia A
           --nu poate fi efectuat INSERT care urmeaza
  INSERT INTO ...
EXCEPTION
  WHEN A THEN ...
END;
```

Deficiența anterioară se poate rezolva incluzând într-un subbloc comanda *SELECT* care a declanșat excepția.

```
BEGIN
  DELETE ...
  BEGIN
    SELECT ...
    ...
  EXCEPTION
    WHEN A THEN ...
    /* dupa ce se trateaza exceptia A, controlul este
       transferat blocului de nivel superior, de fapt
       comenzii INSERT */
  END;
  INSERT INTO ...
  ...
EXCEPTION
...
END;
```

Uneori este dificil de aflat care comandă *SQL* a determinat o anumită eroare, deoarece există o singură secțiune pentru tratarea erorilor unui bloc. Sunt sugerate două soluții pentru rezolvarea acestei probleme.

1) Introducerea unui contor care să identifice instrucțiunea *SQL*.

```
DECLARE
  v_sel_cont  NUMBER(2) :=1;
BEGIN
  SELECT ...
  v_sel_cont:=2;
  SELECT ...
  v_sel_cont:=3;
  SELECT ...
EXCEPTION
  WHEN NO_DATA_FOUND THEN
```

```

INSERT INTO log_table(info)
VALUES ('comandă SELECT ' || TO_CHAR(v_sel_cont) ||
      ' nu gaseste date');
END;
```

2) Introducerea fiecărei instrucțiuni *SQL* într-un subbloc.

```

BEGIN
  BEGIN
    SELECT ...
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      INSERT INTO log_table(info)
      VALUES('SELECT 1 nu gaseste date');
  END;
  BEGIN
    SELECT ...
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      INSERT INTO log_table(info)
      VALUES('SELECT 2 nu gaseste date');
  END;
  ...
END;
```

Activarea excepțiilor

Pentru activarea unei excepții există două metode:

- activarea explicită a excepției (definite de utilizator sau predefinite) în interiorul blocului, cu ajutorul comenzii *RAISE*;
- activarea automată a excepției asociate unei erori *Oracle*.

Excepțiile pot fi sesizate în secțiunea executabilă, declarativă sau în cea de tratare a excepțiilor. La aceste niveluri ale programului, o excepție poate fi gestionată în moduri diferite.

Pentru a reinvoca o excepție, după ce a fost tratată în blocul curent, se folosește instrucțiunea *RAISE*, dar fără a fi însoțită de numele excepției. În acest fel, după executarea instrucțiunilor corespunzătoare tratării excepției, aceasta se transmite și blocului „părinte”. Pentru a fi recunoscută ca atare de către blocul „părinte”, excepția trebuie să nu fie definită în blocul curent, ci în blocul „părinte” (sau chiar mai sus în ierarhie), în caz contrar ea putând fi captată de către blocul „părinte” doar la categoria *OTHERS*.

Pentru a executa același set de acțiuni în cazul mai multor excepții nominalizate explicit, în secțiunea de prelucrare a excepțiilor se poate utiliza operatorul *OR*.

Pentru a evita tratarea fiecărei erori în parte, se folosește secțiunea *WHEN OTHERS* care va cuprinde acțiuni pentru fiecare excepție care nu a fost tratată, adică pentru captarea excepțiilor neprevăzute sau necunoscute. Această secțiune trebuie utilizată cu atenție deoarece poate masca erori critice sau poate împiedica aplicația să răspundă în mod corespunzător.

Propagarea excepțiilor

Dacă este declanșată o eroare în secțiunea executabilă și blocul curent are un *handler* pentru tratarea ei, atunci blocul se termină cu succes, iar controlul este dat blocului imediat exterior.

Dacă se produce o excepție care nu este tratată în blocul curent, atunci excepția se propagă spre blocul „părinte“, iar blocul *PL/SQL* curent se termină fără succes. Procesul se repetă până când fie se găsește într-un bloc modalitatea de tratare a erorii, fie se oprește execuția și se semnalează situația apărută (*unhandled exception error*).

Dacă este declanșată o eroare în partea declarativă a blocului, aceasta este propagată către blocul imediat exterior, chiar dacă există un *handler* al acesteia în blocul corespunzător secțiunii declarative.

La fel se întâmplă dacă o eroare este declanșată în secțiunea de tratare a erorilor. La un moment dat, într-o secțiune *EXCEPTION*, poate fi activă numai o singură excepție.

Instrucțiunea *GOTO* nu permite:

- saltul la secțiunea de tratare a unei excepții;
- saltul de la secțiunea de tratare a unei excepții, în blocul curent.

Comanda *GOTO* permite totuși saltul de la secțiunea de tratare a unei excepții la un bloc care include blocul curent.

Exemplu:

Exemplul următor marchează un salt ilegal în blocul curent.

```
DECLARE
  v_var  NUMBER(10,3);
BEGIN
  SELECT dim2/NVL(valoare,0)
  INTO   v_var
  FROM   opera
  WHERE  dim1 > 100;
  <<eticheta>>
  INSERT INTO politaasig(cod_polita, valoare)
  VALUES (7531, v_var);
EXCEPTION
  WHEN ZERO_DIVIDE THEN v_var:=0;
  GOTO <<eticheta>>; --salt ilegal in blocul curent
END;
```

În continuare, vor fi analizate modalitățile de propagare a excepțiilor în cele trei cazuri comentate: excepții sesizate în secțiunea declarativă, în secțiunea executabilă și în secțiunea de tratare a erorilor.

Excepție sesizată în secțiunea executabilă

Excepția este sesizată și tratată în subbloc. După aceea, controlul revine blocului exterior.

```
DECLARE
  A EXCEPTION;
BEGIN
  ...
  BEGIN
    RAISE A; -- exceptia A sesizata in subbloc
  EXCEPTION
    WHEN A THEN ...-- exceptia tratata in subbloc
  ...
  END;
-- aici este reluat controlul
END;
```

Excepția este sesizată în subbloc, dar nu este tratată în acesta și atunci se propagă spre blocul exterior. Regula poate fi aplicată de mai multe ori.

```
DECLARE
  A EXCEPTION;
  B EXCEPTION;
BEGIN
  BEGIN
    RAISE B; --exceptia B sesizata in subbloc
  EXCEPTION
    WHEN A THEN ...
    --exceptia B nu este tratata in subbloc
  END;
EXCEPTION
  WHEN B THEN ...
  /* exceptia B s-a propagat spre blocul exterior unde a
  fost tratata, apoi controlul trece in exteriorul blocului */
END;
```

Excepție sesizată în secțiunea declarativă

Dacă în secțiunea declarativă este generată o excepție, atunci aceasta se propagă către blocul exterior, unde are loc tratarea acesteia. Chiar dacă există un *handler* pentru excepție în blocul curent, acesta nu este executat.

Exemplu:

Să se realizeze un program prin care să se exemplifice propagarea erorilor apărute în secțiunea declarativă a unui bloc *PL/SQL*.

Programul calculează numărul creatorilor de opere de artă care au lucrări expuse în muzeu.

```
BEGIN
  DECLARE
    nr_artisti  NUMBER(3) := 'XYZ';
  BEGIN
    SELECT  COUNT (DISTINCT cod_autor)
    INTO    nr_artisti
    FROM    opera;
  EXCEPTION
    WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('Eroare bloc intern:' || SQLERRM);
  END;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Eroare bloc extern:' || SQLERRM );
END;
```

Deoarece la inițializarea variabilei *nr_artisti* apare o neconcordanță între tipul declarat și cel asignat, este generată eroarea internă *VALUE_ERROR*. Cum eroarea a apărut în partea declarativă a blocului intern, deși acesta conține un *handler OTHERS* care ar fi putut capta eroarea, *handler*-ul nu este executat, eroarea fiind propagată către blocul extern unde este tratată în *handler*-ul *OTHERS* asociat. Aceasta se poate remarca deoarece la execuție se obține mesajul: „*Eroare bloc extern: ORA-06502: PL/SQL: numeric or value error*“.

Excepție sesizată în secțiunea *EXCEPTION*

Dacă excepția este sesizată în secțiunea *EXCEPTION*, ea se propagă imediat spre blocul exterior.

```
BEGIN
  DECLARE
    A  EXCEPTION;
    B  EXCEPTION;
  BEGIN
    RAISE A; --sesizare exceptie A
  EXCEPTION
    WHEN A THEN
      RAISE B; --sesizare exceptie B
    WHEN B THEN ...
      /* exceptia este propagata spre blocul exterior
      cu toate ca exista aici un handler pentru ea */
  END;
EXCEPTION
  WHEN B THEN ...
    --exceptia B este tratata in blocul exterior
END;
```

Informații despre erori

Pentru a obține textul corespunzător erorilor la compilare, poate fi utilizată vizualizarea *USER_ERRORS* din dicționarul datelor. Pentru informații adiționale referitoare la erori pot fi consultate vizualizările *ALL_ERRORS* sau *DBA_ERRORS*.

Vizualizarea *USER_ERRORS* are câmpurile:

NAME (numele obiectului),

TYPE (tipul obiectului),

SEQUENCE (numărul secvenței),

LINE (numărul liniei din codul sursă în care a apărut eroarea),

POSITION (poziția în linie unde a apărut eroarea),

TEXT (mesajul asociat erorii).

Exemplu:

Să se afișeze erorile de compilare din procedura *alfa*.

```
SELECT LINE, POSITION, TEXT
FROM   USER_ERRORS
WHERE  NAME = 'ALFA';
```

LINE specifică numărul liniei în care apare eroarea, dar acesta nu corespunde liniei efective din fișierul text (se referă la codul sursă depus în *USER_SOURCE*).