

Tehnici de programare cu baze de date

#7

PL/SQL

Excepții în PL/SQL

<https://www.runceanu.ro/adrian/activitate-didactica/>

Curs 7

Excepții în PL/SQL

Excepții în PL/SQL

- 1. Manipularea excepțiilor**
- 2. „Prinderea în capcană” a excepțiilor**
- 3. Prinderea excepțiilor serverului Oracle**
- 4. Excepții de prindere definite de utilizator**

1. Manipularea excepțiilor

- Am studiat pana acum blocurile PL/SQL care au parte declarativa si parte executabila.
- Orice cod SQL si PL/SQL care trebuie executat se scrie intr-un bloc executabil.
- Pana acum am presupus ca, codul functioneaza corect.
- Dar codul poate produce erori neprevazute la un moment dat.
- O sa studiem cum sa ne descurcam cu astfel de erori in blocurile PL/SQL.

1. Manipularea excepțiilor

Ce este o excepție?

- *O excepție apare atunci când este descoperită o eroare în timpul execuției unui program și care perturbă funcționarea normală a programului.*

Sunt multe cauze posibile ale excepțiilor:

1. Un utilizator face o greșeală de ortografie în timp ce tastează
2. Un program nu funcționează corect
3. O pagină web de publicitate nu există, etc.

1. Manipularea excepțiilor

- Atunci când codul nu funcționează cum era de așteptat, **PL/SQL** produce o excepție.
- Atunci când este produsă (provocată) o excepție, restul codului din partea executabilă a blocului **PL/SQL** nu se mai execută.

1. Manipularea excepțiilor

Ce este un manipulator de excepții?

- *Un manipulator de excepții este un cod care definește acțiunile de recuperare care trebuie executate atunci când se produce o excepție.*
- In timpul scrierii unui cod, programatorii trebuie să anticipeze tipurile de erori care pot apărea în timpul execuției codului respectiv.
- Este necesară includerea manipulatorilor de excepție în cod pentru a aborda aceste erori.
- Într-un fel, manipulatorii de excepție permit programatorilor să-și protejeze codul.

1. Manipularea excepțiilor

De ce tipuri de erori ar trebui să țină seama programatorii prin folosirea unor subprograme de tratare a excepțiilor (manipulatorii de excepție) ?

1. **Erori de sistem** (de exemplu hard-diskul este plin)
2. **Erori de date** (de exemplu încercarea de a duplica valoarea unei chei primare)
3. **Erori de utilizatori** (de exemplu erori la introducerea datelor de intrare)
4. **Multe alte posibilitati!**

1. Manipularea excepțiilor

De ce este important manipulatorul de excepții?

- Protejarea de erori a utilizatorilor (erorile frecvente pot fi frustrante pentru utilizatori și/sau pot determina ieșirea utilizatorului din aplicație)
- Protejarea de erori a bazei de date (datele pot fi pierdute sau suprascrise)
- Erorile importante iau mult din resursele sistemului (dacă se face o greșeală corectarea acesteia poate fi costisitoare – utilizatorii pot solicita frecvent serviciul de asistență pentru erori)
- Codul este mai ușor de citit deoarece rutinele manipulatorului de eroare pot fi scrise în același bloc în care a apărut eroarea.

1. Manipularea excepțiilor

Manipularea excepțiilor cu PL/SQL

- Un bloc întotdeauna se încheie atunci când **PL/SQL** produce o excepție, dar putem specifica un manipulator de excepție pentru efectuarea acțiunilor finale înainte de încheierea blocului.
- Secțiunea excepției începe cu cuvântul cheie **EXCEPTION**.

1. Manipularea excepțiilor

Exemplu:

DECLARE

**v_country_name eba_countries.name%TYPE := 'Korea,
South';**

v_population eba_countries.population%TYPE;

BEGIN

SELECT population INTO v_population

FROM eba_countries

WHERE name = v_country_name;

EXCEPTION

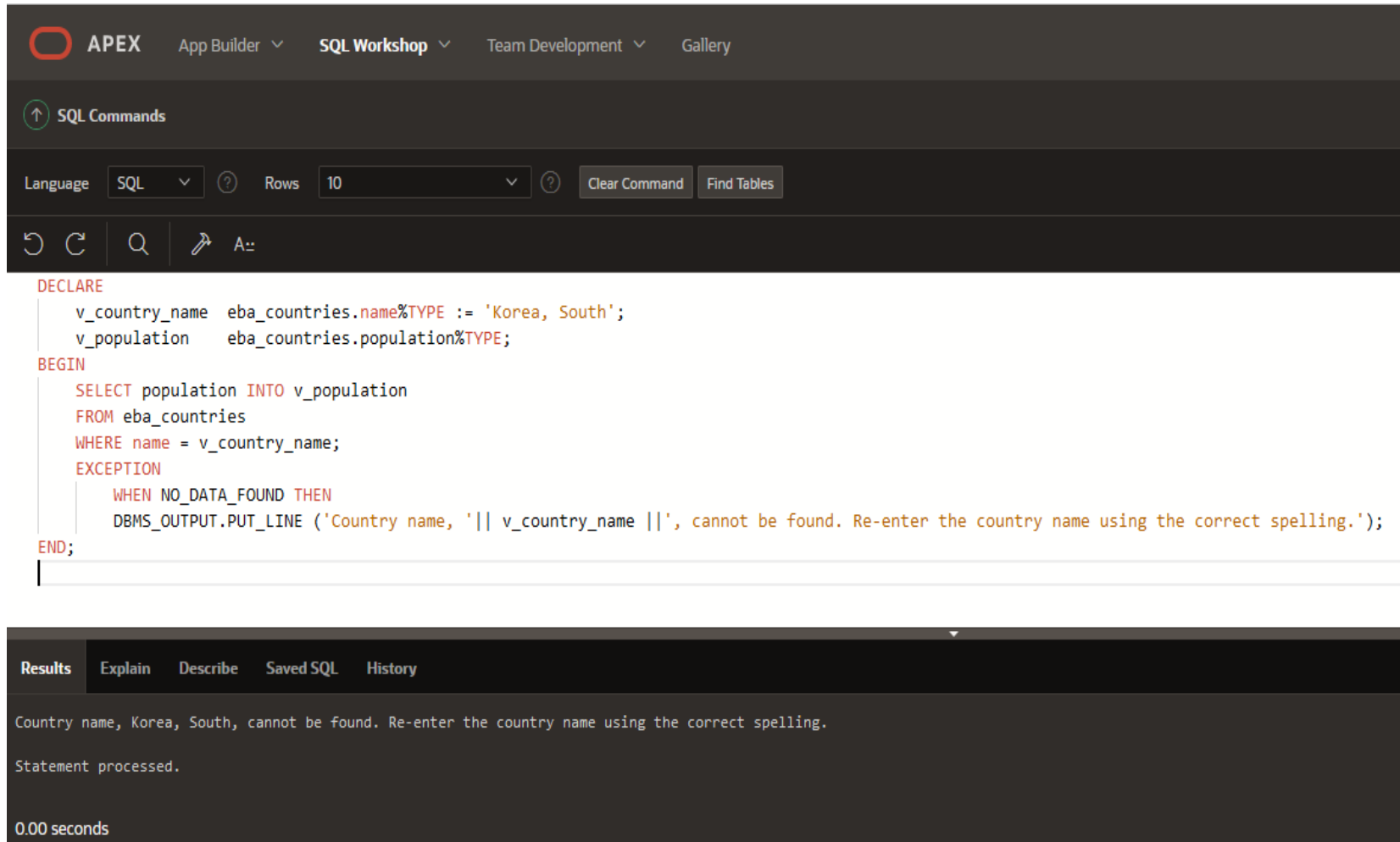
WHEN NO_DATA_FOUND THEN

**DBMS_OUTPUT.PUT_LINE ('Country name, '||
v_country_name ||', cannot be found. Re-enter the
country name using the correct spelling.');**

END;

1. Manipularea excepțiilor

Execuția codului propus:



The screenshot displays the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a 'Language' dropdown set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command' and 'Find Tables'. The SQL editor contains the following code:

```
DECLARE
  v_country_name eba_countries.name%TYPE := 'Korea, South';
  v_population   eba_countries.population%TYPE;
BEGIN
  SELECT population INTO v_population
  FROM eba_countries
  WHERE name = v_country_name;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE ('Country name, ' || v_country_name || ', cannot be found. Re-enter the country name using the correct spelling.');
```

The 'Results' tab is selected, showing the output of the query:

```
Country name, Korea, South, cannot be found. Re-enter the country name using the correct spelling.

Statement processed.

0.00 seconds
```

1. Manipularea excepțiilor

- Atunci când este manipulată o excepție, programul **PL/SQL** nu se încheie brusc.
- *Atunci când se produce excepția, controlul se transferă secțiunii excepției și este executat manipulatorul de excepție din acea secțiune.*
- Blocul **PL/SQL** se încheie în mod obișnuit și se finalizează cu succes.
- Poate apărea o singură excepție la un moment dat.
- Atunci când apare o excepție, **PL/SQL** prelucrează un singur manipulator înainte de ieșirea din bloc.

Exemple:

1)

DECLARE

**v_country_name eba_countries.name%TYPE :=
'Korea, South';**

v_population eba_countries.population%TYPE;

BEGIN

SELECT population INTO v_population

FROM eba_countries

WHERE name = v_country_name;

DBMS_OUTPUT.PUT_LINE(v_population); -- Punct A

EXCEPTION

WHEN NO_DATA_FOUND THEN

**DBMS_OUTPUT.PUT_LINE ('Country name, ||
v_country_name ||', cannot be found. Re-enter the
country name using the correct spelling.);**

END;

Observatie:

Codul de la punctul A nu se executa deoarece instructiunea SELECT esueaza.

```
1 DECLARE
2   v_country_name eba_countries.name%TYPE := 'Korea, South';
3   v_population   eba_countries.population%TYPE;
4 BEGIN
5   SELECT population INTO v_population
6   FROM eba_countries
7   WHERE name = v_country_name;
8   DBMS_OUTPUT.PUT_LINE(v_population);    -- Punct A
9   EXCEPTION
10  WHEN NO_DATA_FOUND THEN
11    DBMS_OUTPUT.PUT_LINE ('Country name, ' || v_country_name || ', cannot be found. Re-enter the country name using the correct spelling. ');
12 END;
13
```

Results Explain Describe Saved SQL History

Country name, Korea, South, cannot be found. Re-enter the country name using the correct spelling.

Statement processed.

2)

DECLARE

v_ename VARCHAR2(15);

BEGIN

SELECT ename INTO v_ename

FROM emp

WHERE job = 'ANALYST';

**DBMS_OUTPUT.PUT_LINE('The last name
of the ANALYST is :'||v_ename);**

END;

Instructiunea **SELECT** din bloc gaseste ***ename*** pentru **ANALYST**.

Oricum, se produce o exceptie deoarece exista mai multe date pentru **ANALYST**

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language SQL Rows 10 Clear Command Find Tables

SQL Editor toolbar: Undo, Refresh, Search, Run, A-Z

```
DECLARE
  v_ename VARCHAR2(15);
BEGIN
  SELECT ename INTO v_ename
  FROM emp
  WHERE job = 'ANALYST';
  DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST is :'||v_ename);
END;
```

Results Explain Describe Saved SQL History

ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 4
ORA-06512: at "SYS.DBMS_SQL", line 1721

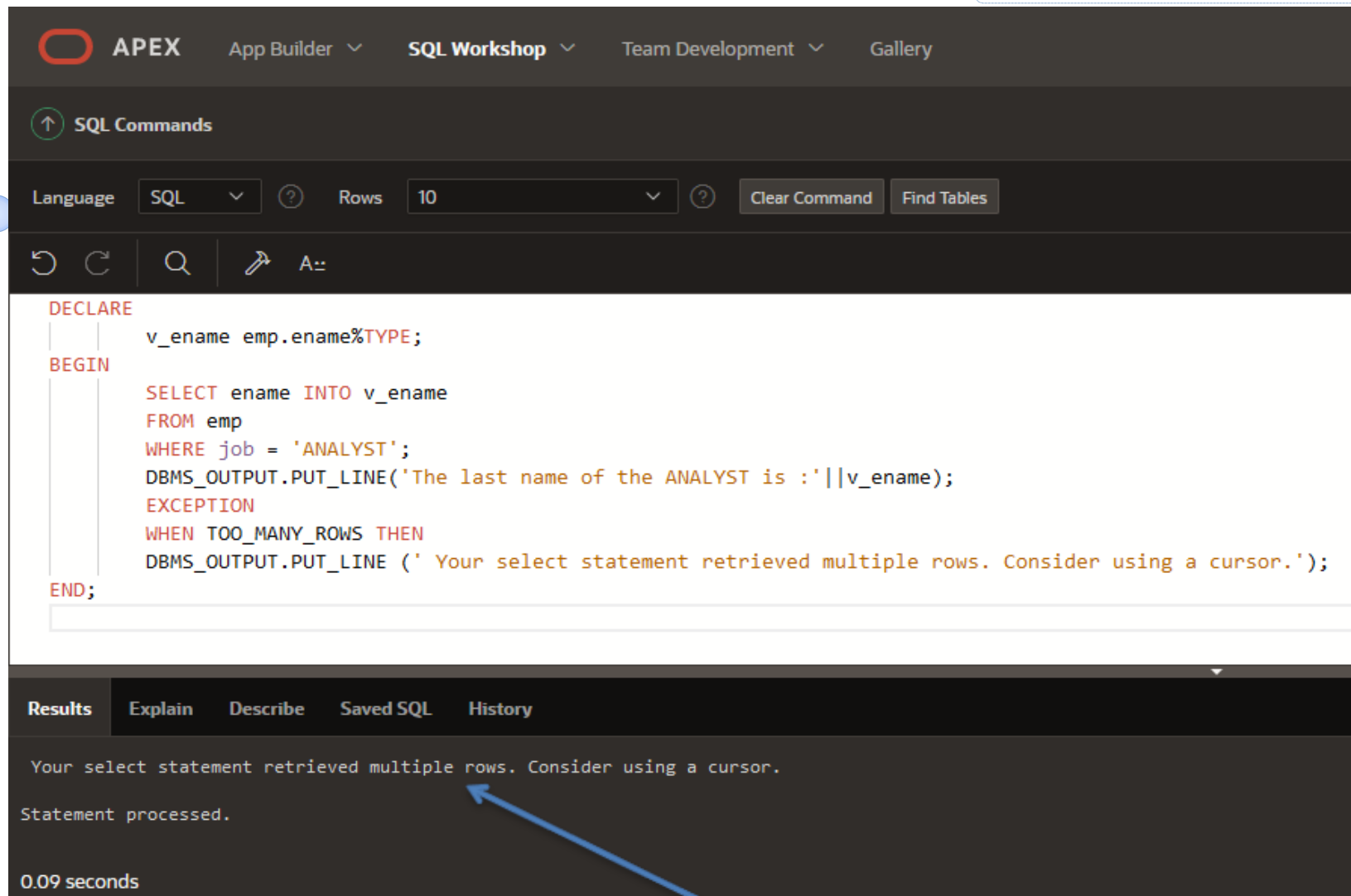
```
2.      v_ename VARCHAR2(15);
3. BEGIN
4.      SELECT ename INTO v_ename
5.      FROM emp
6.      WHERE job = 'ANALYST';
```

0.06 seconds

3)

DECLARE**v_ename emp.ename%TYPE;****BEGIN****SELECT ename INTO v_ename****FROM emp****WHERE job = 'ANALYST';****DBMS_OUTPUT.PUT_LINE('The last name of the
ANALYST is :'||v_ename);****EXCEPTION****WHEN TOO_MANY_ROWS THEN****DBMS_OUTPUT.PUT_LINE (' Your select
statement retrieved multiple rows. Consider using a
cursor.');****END;**

Acest cod contine un manipulator pentru o eroare predefinita a serverului **Oracle** numita **TOO_MANY_ROWS**.



The screenshot displays the APEX SQL Workshop interface. At the top, the navigation bar includes the APEX logo, 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is active, showing a 'Language' dropdown set to 'SQL', a 'Rows' dropdown set to '10', and buttons for 'Clear Command' and 'Find Tables'. The main area contains a PL/SQL script:

```
DECLARE
    v_ename emp.ename%TYPE;
BEGIN
    SELECT ename INTO v_ename
    FROM emp
    WHERE job = 'ANALYST';
    DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST is :'||v_ename);
    EXCEPTION
    WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE (' Your select statement retrieved multiple rows. Consider using a cursor. ');
END;
```

The 'Results' tab is selected, showing the following output:

```
Your select statement retrieved multiple rows. Consider using a cursor.
Statement processed.
0.09 seconds
```

A blue arrow points from a callout box to the error message in the results.

Eroare (exceptie)
care intrerupe
executia interogarii

Excepții în PL/SQL

1. Manipularea excepțiilor
2. „Prinderea în capcană” a excepțiilor
3. Prinderea excepțiilor serverului Oracle
4. Excepții de prindere definite de utilizator

2. „Prinderea în capcană” a excepțiilor

Putem manipula sau „prinde în capcana” orice eroare prin includerea unui manipulator corespunzător în secțiunea de manipulare a excepțiilor a blocului **PL/SQL**.

Sintaxa este:

EXCEPTION

WHEN exception1 [OR exception2. . .]

THEN

Instructiune1;

Instructiune2; ...

[WHEN exception3 [OR exception4. . .]

THEN

Instructiune1;

Instructiune2; ...]

[WHEN OTHERS THEN

Instructiune1;

Instructiune2; ...]

2. „Prinderea în capcană” a excepțiilor

- *Fiecare manipulator este format dintr-o clauza WHEN care specifica numele unei exceptii, urmata de o secventa de instructiuni care se executa cand se produce exceptia.*
- Putem include oricate manipuloare in sectiunea **EXCEPTION** pentru a ne ocupa de exceptiile specifice.
- De asemenea, nu putem avea mai multi manipulatori pentru aceeasi exceptie.

2. „Prinderea în capcană” a excepțiilor

EXCEPTION

WHEN exception1 [OR exception2. . .]

THEN

Instructiune1;

Instructiune2;

...

[WHEN OTHERS THEN

Instructiune1;

Instructiune2;

...]

2. „Prinderea în capcană” a excepțiilor

In cadrul sintaxei:

- **exception** – este o *denumire standard a unei exceptii predefinite* sau *numele unei exceptii definite de utilizator declarata in partea declarativa*
- **instructiune** – reprezinta una sau mai multe instructiuni **PL/SQL** sau **SQL**
- **OTHERS** – este o clauza optionala de manipulare a exceptiilor ce preia orice exceptie ce nu a fost manipulata explicit

2. „Prinderea în capcană” a excepțiilor

Manipulatorul de excepții OTHERS

- Secțiunea de manipulare a excepțiilor prinde doar acele excepții care sunt specificate.
- Orice alte excepții nu sunt prinse până nu folosim manipulatorul de excepții **OTHERS**.
- Manipulatorul **OTHERS** prinde toate excepțiile care nu au fost deja prinse.
- Atunci când este folosit, **OTHERS** trebuie să fie ultimul manipulator de excepție care este definit.

2. „Prinderea în capcană” a excepțiilor

Fie urmatorul exemplu:

1. Dacă programul întâlnește excepția **NO_DATA_FOUND** atunci sunt executate instrucțiunile din manipulatorul corespunzător.
2. La fel se întâmplă și dacă este întâlnită excepția **TOO_MANY_ROWS**.
3. Dacă sunt întâlnite alte excepții atunci sunt executate instrucțiunile manipulatorului **OTHERS**.

2. „Prinderea în capcană” a excepțiilor

EXCEPTION

WHEN NO_DATA_FOUND THEN
 Instructiune1;

...

WHEN TOO_MANY_ROWS THEN
 Instructiune2;

...

WHEN OTHERS THEN
 Instructiune3;

2. „Prinderea în capcană” a excepțiilor

Reguli pentru prinderea excepțiilor

1. Intotdeauna adaugati manipuloare de exceptii atunci cand exista posibilitatea de aparitie a unei erori.
2. Erorile apar mai ales:
 - a) la calcule
 - b) la manipularea sirurilor de caractere
 - c) si la operatiile asupra bazelor de date

2. „Prinderea în capcană” a excepțiilor

Reguli pentru prinderea excepțiilor

3. Folositi manipuloare de exceptie cu nume in locul folosirii manipulatorului **OTHERS**.
4. Invatati denumirile si cauzele exceptiilor predefinite
5. Testati codul cu diferite combinatii de date gresite pentru a vedea potentialele erori
6. Scrieti informatiile care apar la depanare in manipulatorul vostru de exceptie

Excepții în PL/SQL

1. Manipularea excepțiilor
2. „Prinderea în capcană” a excepțiilor
3. Prinderea excepțiilor serverului
Oracle
4. Excepții de prindere definite de utilizator

3. Prinderea excepțiilor serverului Oracle

- Manipularea erorilor **PL/SQL** este flexibilă și permite programatorilor să folosească:
 - erori definite de serverul **Oracle**
 - cât și cele definite de programator
- Erorile predefinite sunt erori **Oracle** obișnuite pentru care **PL/SQL** are denumiri de excepții predefinite.
- Erorile care nu sunt predefinite ne determină să folosim codurile de erori și mesajele **ORA**.
- Sintaxa este diferită pentru fiecare în parte, dar putem prinde ambele tipuri de erori în secțiunea **EXCEPTION** a blocului **PL/SQL**.

Tipuri de excepții

Exception	Description	Instructions for Handling
Predefined Oracle server error	One of approximately 20 errors that occur most often in PL/SQL code	You need not declare these exceptions. They are predefined by the Oracle server and are raised implicitly (automatically).
Non-predefined Oracle server error	Any other standard Oracle server error	Declare within the declarative section and allow the Oracle Server to raise them implicitly (automatically).
User-defined error	A condition that the PL/SQL programmer decides is abnormal	Declare within the declarative section, and raise explicitly.

3. Prinderea excepțiilor serverului Oracle

Manipularea excepțiilor cu PL/SQL

Sunt două metode pentru a trata o excepție:

1. Implicit de către serverul **Oracle** – Are loc o eroare **Oracle** și excepția asociată se produce imediat.

- De exemplu dacă are loc eroarea **ORA-01403** când nici un rând nu este preluat din baza de date într-o instrucțiune **SELECT**, atunci **PL/SQL** produce excepția **NO_DATA_FOUND**

3. Prinderea excepțiilor serverului Oracle

2. *Explicit* de către programator – depinde de modul de implementare a programului.

- Se pot produce excepții folosind instrucțiunea **RAISE** în interiorul blocului.
- Excepțiile produse pot fi atât definite de utilizator cât și predefinite.

3. Prinderea excepțiilor serverului Oracle

Tipuri de erori ale serverului Oracle

Atunci când au loc erori ale serverului **Oracle**, automat se produce excepția asociată, se ignoră restul secțiunii executabile a blocului și se caută un manipulator în secțiunea de excepție.

Sunt **doua tipuri de erori ale serverului Oracle**:

- 1. Erori predefinite ale serverului Oracle* – fiecare dintre aceste erori are un nume predefinit
- 2. Erori ale serverului Oracle care nu sunt predefinite* – fiecare dintre aceste erori are un număr de eroare standard (**ORA-nnnnn**) și un mesaj de eroare, dar nu are un nume predefinit.

- Va puteți declara propriile nume pentru aceste erori astfel încât puteți referi aceste erori în secțiunea de excepție.

3. Prinderea excepțiilor serverului Oracle

3.1. Capturarea erorilor predefinite ale serverului Oracle

- Numele predefinite sunt referite în rutina de manipulare a excepțiilor
- Exemple de excepții predefinite:
 1. **NO_DATA_FOUND**
 2. **TOO_MANY_ROWS**
 3. **INVALID_CURSOR**
 4. **ZERO_DIVIDE**
 5. **DUP_VAL_ON_INDEX**

3. Prinderea excepțiilor serverului Oracle

Exemplu1 – urmatorul exemplu foloseste eroarea predefinita **TOO_MANY_ROWS**.

Observati ca nu este declarata in sectiunea **DECLARATION**

DECLARE

```
v_ename VARCHAR2(15);
```

BEGIN

```
SELECT ename INTO v_ename
```

```
FROM emp
```

```
WHERE job = 'ANALYST';
```

```
DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST  
is : '||v_ename);
```

EXCEPTION

```
WHEN TOO_MANY_ROWS THEN
```

```
DBMS_OUTPUT.PUT_LINE (' Your select statement  
retrieved multiple rows. Consider using a cursor.');
```

END;

```
1 DECLARE
2   v_ename VARCHAR2(15);
3 BEGIN
4   SELECT ename INTO v_ename
5   FROM emp
6   WHERE job = 'ANALYST';
7   DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST is : '||v_ename);
8   EXCEPTION
9     WHEN TOO_MANY_ROWS THEN
10      DBMS_OUTPUT.PUT_LINE (' Your select statement retrieved multiple rows. Consider using a cursor. ');
11 END;
12
```

Results Explain Describe Saved SQL History

Your select statement retrieved multiple rows. Consider using a cursor.

Statement processed.

Exemplu2 – acest exemplu trateaza exceptiile **TOO_MANY_ROWS** si **NO_DATA_FOUND** si are un manipulator **OTHERS** in cazul oricaror altor erori.

° **DECLARE**

v_ename VARCHAR2(15);

BEGIN

SELECT ename INTO v_ename

FROM emp

WHERE job = 'ANALYST';

**DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST is :
'||v_ename);**

EXCEPTION

WHEN TOO_MANY_ROWS THEN

**DBMS_OUTPUT.PUT_LINE ('Select statement found multiple
rows');**

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE ('Select statement found no rows');

WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE ('Another type of error occurred');

END;

```
Language SQL ? Rows 10 ? Clear Command Find Tables
↶ ↷ 🔍 📌 A::
1 DECLARE
2   v_ename VARCHAR2(15);
3 BEGIN
4   SELECT ename INTO v_ename
5   FROM emp
6   WHERE job = 'ANALYST';
7   DBMS_OUTPUT.PUT_LINE('The last name of the ANALYST is : '||v_ename);
8   EXCEPTION
9     WHEN TOO_MANY_ROWS THEN
10    DBMS_OUTPUT.PUT_LINE ('Select statement found multiple rows');
11    WHEN NO_DATA_FOUND THEN
12    DBMS_OUTPUT.PUT_LINE ('Select statement found no rows');
13    WHEN OTHERS THEN
14    DBMS_OUTPUT.PUT_LINE ('Another type of error occurred');
15 END;
16
```

Results Explain Describe Saved SQL History

Select statement found multiple rows

Statement processed.

3. Prinderea excepțiilor serverului Oracle

3.2. Capturarea erorilor nepredefinite ale serverului Oracle

- Excepțiile nepredefinite sunt asemănătoare cu cele predefinite, totuși ele nu au nume predefinite în **PL/SQL**.
- *Sunt erori standard ale serverului Oracle și au numere de eroare **ORA**.*
- Va creați propriile nume pentru ele în secțiunea **DECLARE** și asociați aceste nume cu numerele de eroare **ORA** folosind funcția ***PRAGMA EXCEPTION_INIT***.

3. Prinderea excepțiilor serverului Oracle

3.2. Capturarea erorilor nepredefinite ale serverului Oracle (continuare)

- Puteti captura o eroare a serverului **Oracle** care nu este predefinita declarand-o mai intai.
- Exceptia declarata se produce implicit.
- In **PL/SQL PRAGMA EXCEPTION_INIT** spune compilatorului sa asocieze un nume de exceptie cu un numar de eroare **Oracle**.
- Va permite sa referiti orice exceptie a serverului **Oracle** prin nume si sa scrieti un manipulator specific pentru ea.

Tipuri de excepții

Exception	Description	Instructions for Handling
Predefined Oracle server error	One of approximately 20 errors that occur most often in PL/SQL code	You need not declare these exceptions. They are predefined by the Oracle server and are raised implicitly (automatically).
Non-predefined Oracle server error	Any other standard Oracle server error	Declare within the declarative section and allow the Oracle Server to raise them implicitly (automatically).
User-defined error	A condition that the PL/SQL programmer decides is abnormal	Declare within the declarative section, and raise explicitly.

3. Prinderea excepțiilor serverului Oracle

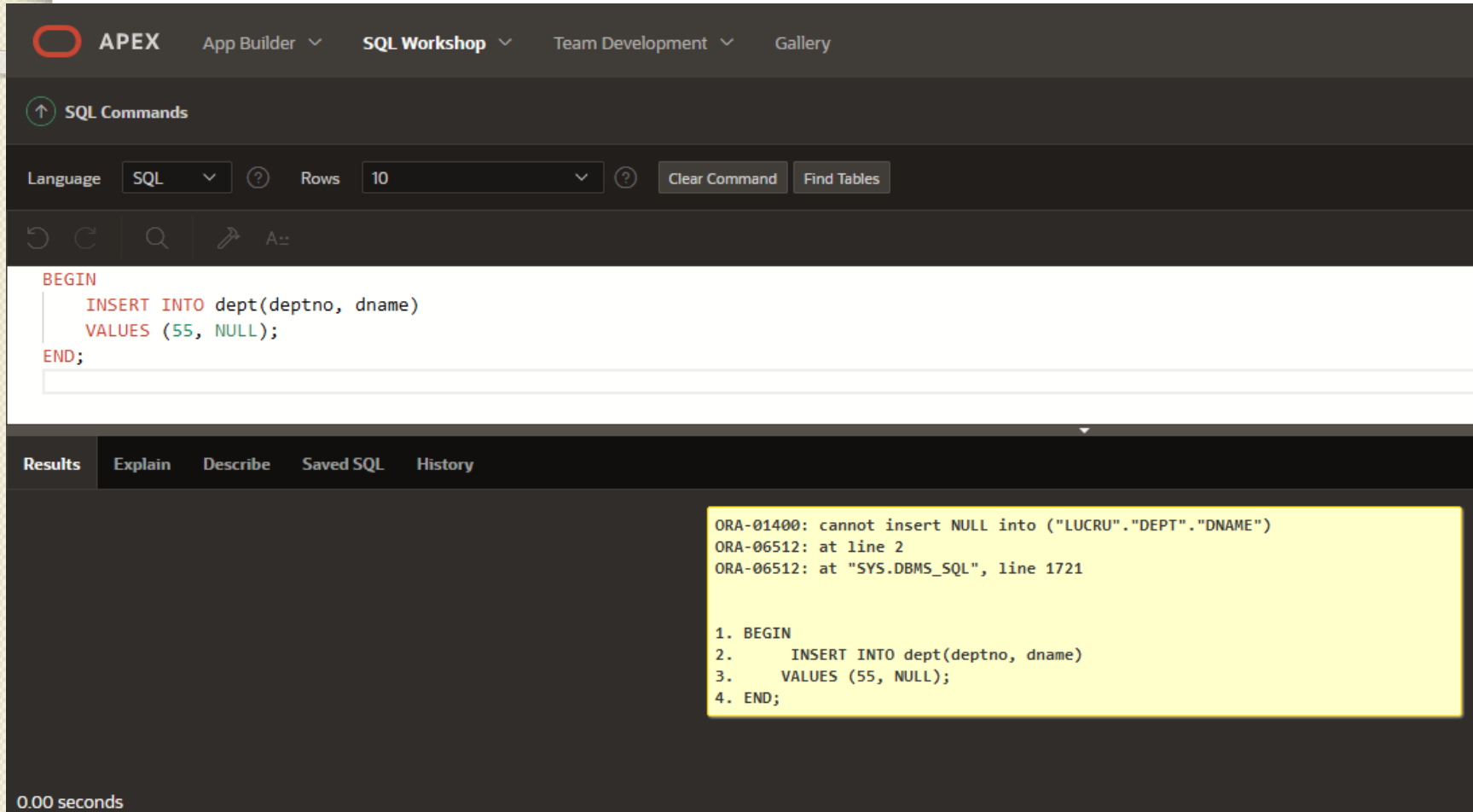
Erori care nu sunt predefinite

Examinați următorul exemplu:

```
BEGIN  
  INSERT INTO dept (deptno, dname)  
  VALUES (80, NULL);  
END;
```

Se va afișa:

```
ORA-01400: cannot insert NULL into  
(“LUCRU”. “DEPT”. “DNAME”)
```



The screenshot shows the Oracle APEX SQL Workshop interface. At the top, the navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is active, showing 'Language' set to 'SQL' and 'Rows' set to '10'. The SQL command entered is:

```
BEGIN
  INSERT INTO dept(deptno, dname)
  VALUES (55, NULL);
END;
```

The 'Results' tab is selected, displaying the following error messages:

```
ORA-01400: cannot insert NULL into ("LUCRU"."DEPT"."DNAME")
ORA-06512: at line 2
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

Below the error messages, the execution plan is shown:

```
1. BEGIN
2.     INSERT INTO dept(deptno, dname)
3.     VALUES (55, NULL);
4. END;
```

At the bottom left, the execution time is displayed as '0.00 seconds'.

3. Prinderea excepțiilor serverului Oracle

- Instrucțiunea **INSERT** încearcă să insereze valoarea **NULL** pentru coloana **department_name** a tabelii **departments**.
- Totuși, operația nu are succes deoarece **department_name** este o coloană **NOT NULL**.
- Nu este nici un nume de eroare predefinită pentru încălcarea unei constrangeri **NOT NULL**.
- Modul de rezolvare a acestei probleme este de a declara propriul nume și a-l asocia cu eroarea ORA-01400.

3. Prinderea excepțiilor serverului Oracle

Pasii necesari:

1. Declararea numelui exceptiei in sectiunea declarativa
2. Asocierea exceptiei declarate cu numarul erorii standard a serverului Oracle folosind functia **PRAGMA EXCEPTION_INIT**
3. Referirea numelui exceptiei declarate in cadrul rutinei corespunzatoare manipulatorului

3. Prinderea excepțiilor serverului Oracle

Exemplu:

DECLARE

e_insert_excep EXCEPTION;

**PRAGMA EXCEPTION_INIT (e_insert_excep, -
01400);**

BEGIN

INSERT INTO dept (deptno, dname)

VALUES (66, NULL);

EXCEPTION

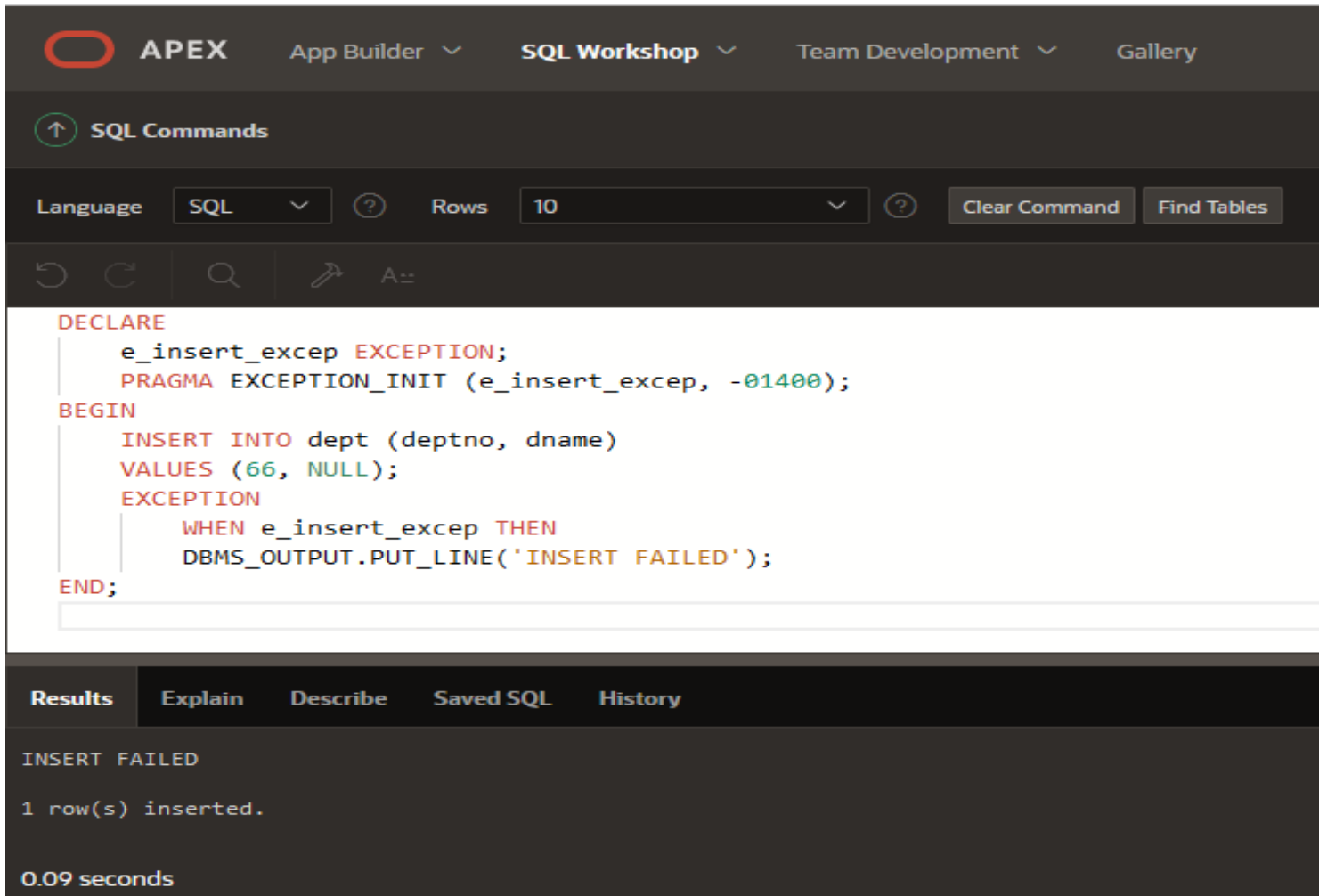
WHEN e_insert_excep THEN

DBMS_OUTPUT.PUT_LINE('INSERT FAILED');

END;

3. Prinderea excepțiilor serverului Oracle

Execuția codului propus:



The screenshot displays the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a 'Language' dropdown set to 'SQL' and 'Rows' set to '10'. The main area contains the following PL/SQL code:

```
DECLARE
    e_insert_excep EXCEPTION;
    PRAGMA EXCEPTION_INIT (e_insert_excep, -01400);
BEGIN
    INSERT INTO dept (deptno, dname)
    VALUES (66, NULL);
    EXCEPTION
        WHEN e_insert_excep THEN
            DBMS_OUTPUT.PUT_LINE('INSERT FAILED');
END;
```

The 'Results' tab is selected, showing the output of the execution:

```
INSERT FAILED
1 row(s) inserted.
0.09 seconds
```

3. Prinderea excepțiilor serverului Oracle

Functii pentru capturarea excepțiilor

- Atunci când are loc o excepție, puteți regăsi codul erorii asociate sau un mesaj de eroare prin folosirea a două funcții:
 1. **SQLERRM** – *returnează un șir de caractere ce conține mesajul asociat cu numărul erorii*
 2. **SQLCODE** – *returnează valoarea numerică pentru codul erorii (O puteți atribui unei variabile de tip NUMBER)*

Pe baza valorilor codului sau pe baza mesajului, puteți decide măsurile ulterioare ce trebuie luate.

3. Prinderea excepțiilor serverului Oracle

SQLCODE Value	Description
0	No exception encountered
1	User defined exception
+100	NO_DATA_FOUND exception
Negative number	Another Oracle Server error number

- Funcțiile **SQLCODE** și **SQLERRM** nu pot fi folosite direct într-o instrucțiune **SQL**.
- În schimb, trebuie să atribuiți valorile lor unor variabile locale, apoi folosiți acele variabile în instrucțiuni **SQL** așa cum va arăta următorul exemplu:

3. Prinderea excepțiilor serverului Oracle

DECLARE

v_error_code NUMBER;

v_error_message VARCHAR2(255);

BEGIN

...

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

v_error_code := SQLCODE ;

v_error_message := SQLERRM ;

**INSERT INTO error_log(e_user, e_date, error_code,
error_message)**

**VALUES(USER, SYSDATE, v_error_code,
v_error_message);**

END;

Excepții în PL/SQL

- 1. Manipularea excepțiilor**
- 2. „Prinderea în capcană” a excepțiilor**
- 3. Prinderea excepțiilor serverului Oracle**
- 4. Excepții de prindere definite de utilizator**

4. Excepții de prindere definite de utilizator

- Aceste erori nu sunt gasite automat de catre serverul **Oracle**, dar sunt definite de programator si sunt specifice codului programatorului.
- Un exemplu de eroare definita de programator este **INVALID_MANAGER_ID**.
- Puteti defini atat un cod de eroare cat si un mesaj de eroare pentru erorile definite de utilizator.

4. Excepții de prindere definite de utilizator

Exception	Description	Instructions for Handling
Predefined Oracle server error	One of approximately 20 errors that occur most often in PL/SQL code	You need not declare these exceptions. They are predefined by the Oracle server and are raised implicitly.
Non-predefined Oracle server error	Any other standard Oracle server error	Declare within the declarative section and allow the Oracle server to raise them implicitly.
User-defined error	A condition that the developer determines is abnormal	Declare within the declarative section, and raise explicitly.

4. Excepții de prindere definite de utilizator

- **PL/SQL** ne permite sa definim propriile exceptii in functie de cerintele aplicatiei.
- De exemplu puteti dori sa creati o exceptie definita de utilizator atunci cand este nevoie sa abordati conditii de eroare pentru datele de intrare.

4. Excepții de prindere definite de utilizator

De exemplu, sa presupunem ca programul solicita utilizatorului un numar si nume de departament astfel incat sa poata actualiza numele departamentului.

DECLARE

v_name VARCHAR2(20):='IT_PROG';

v_deptno NUMBER := 27;

BEGIN

UPDATE dept

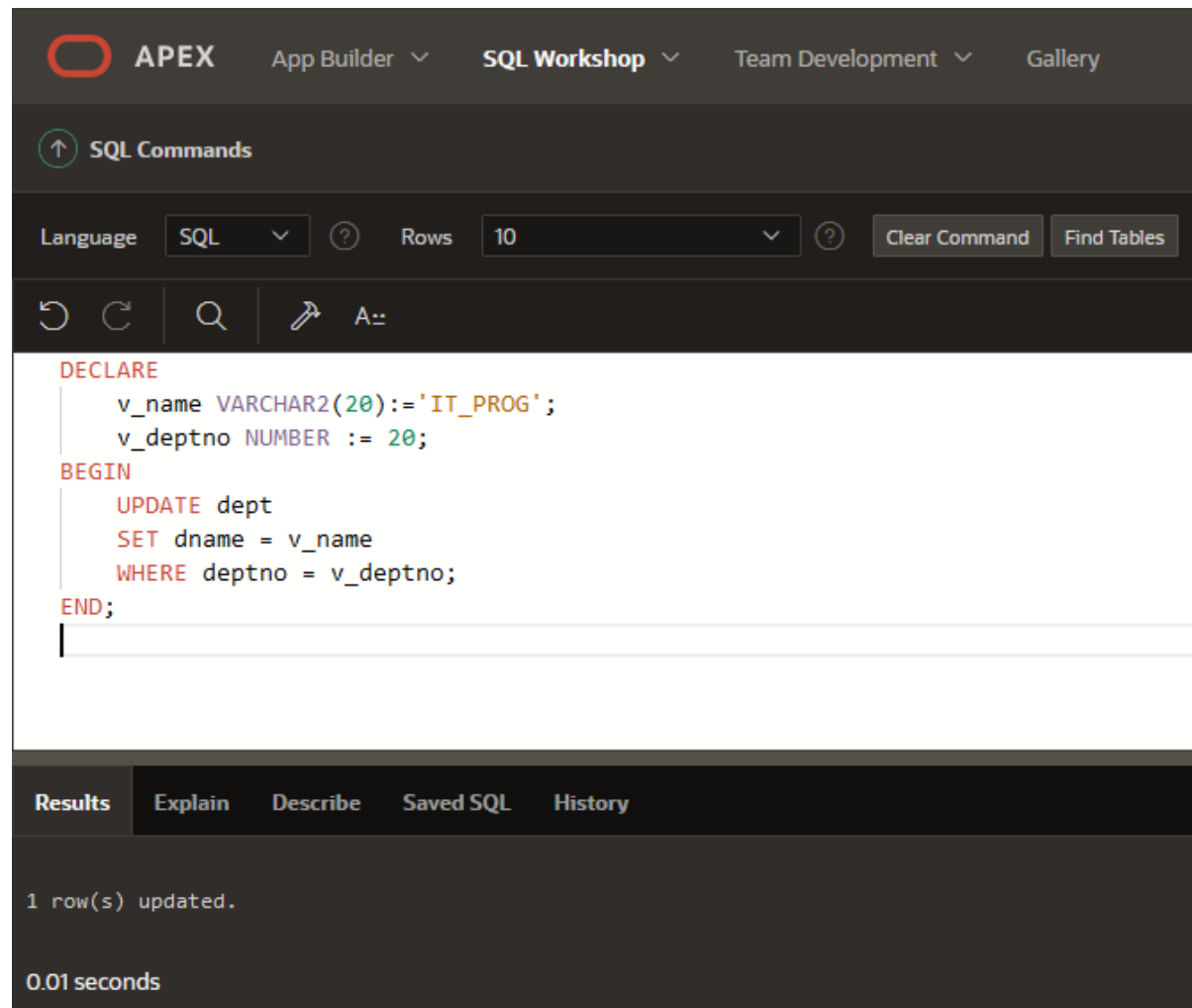
SET dname = v_name

WHERE deptno = v_deptno;

END;

4. Excepții de prindere definite de utilizator

Execuția codului propus:



The screenshot displays the Oracle APEX SQL Workshop interface. At the top, the navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is visible, with a dropdown menu set to 'SQL' and 'Rows' set to '10'. The main area contains the following SQL code:

```
DECLARE
  v_name VARCHAR2(20):='IT_PROG';
  v_deptno NUMBER := 20;
BEGIN
  UPDATE dept
  SET dname = v_name
  WHERE deptno = v_deptno;
END;
```

At the bottom, the 'Results' tab is active, showing the execution output: '1 row(s) updated.' and '0.01 seconds'.

4. Excepții de prindere definite de utilizator

Execuția codului propus:

The screenshot displays the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The main area shows the 'SQL Commands' editor with the following settings: Language set to 'SQL', Rows set to '10', and buttons for 'Clear Command' and 'Find Tables'. The SQL command entered is:

```
select *  
from dept;
```

Below the editor, the 'Results' tab is active, showing a table with the following data:

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	IT_PROG	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

At the bottom of the results area, it indicates '4 rows returned in 0.00 seconds' and provides a 'Download' link.

4. Excepții de prindere definite de utilizator

- Ce se întâmplă atunci când utilizatorul introduce un department incorect?
- Codul de mai sus nu produce o eroare **Oracle**.
- Este nevoie să definim o eroare utilizator.
- Acest lucru se face prin:

4. Excepții de prindere definite de utilizator

1. Declararea numelui excepției definite de utilizator în secțiunea declarativă

```
e_invalid_department EXCEPTION;
```

2. Folosirea instrucțiunii RAISE pentru a produce explicit excepția în secțiunea executabilă.

```
IF SQL%NOTFOUND THEN RAISE  
e_invalid_department;
```

4. Excepții de prindere definite de utilizator

3. *Referirea excepției declarate în rutina manipulatorului de excepție corespunzător*

```
EXCEPTION WHEN
```

```
    e_invalid_department THEN
```

```
    DBMS_OUTPUT.PUT_LINE('No such  
    department id.');
```

DECLARE

```
e_invalid_department EXCEPTION;  
v_name VARCHAR2(20):='Accounting';  
v_deptno NUMBER := 27;
```

BEGIN

```
UPDATE dept  
SET dname = v_name  
WHERE deptno = v_deptno;  
IF SQL%NOTFOUND THEN  
    RAISE e_invalid_department;  
END IF;
```

```
COMMIT;
```

```
EXCEPTION
```

```
    WHEN e_invalid_department  
    THEN DBMS_OUTPUT.PUT_LINE('No  
such department no');  
    ROLLBACK;
```

```
END;
```

```
Language SQL ? Rows 10 ? Clear Command
↶ ↷ 🔍 ↗ A::
1 DECLARE
2     e_invalid_department EXCEPTION;
3     v_name VARCHAR2(20):='Accounting';
4     v_deptno NUMBER := 27;
5 BEGIN
6     UPDATE dept
7     SET dname = v_name
8     WHERE deptno = v_deptno;
9     IF SQL%NOTFOUND THEN
10        RAISE e_invalid_department;
11    END IF;
12    COMMIT;
13    EXCEPTION
14        WHEN e_invalid_department
15        THEN DBMS_OUTPUT.PUT_LINE('No such department no');
16        ROLLBACK;
17 END;
```

Results Explain Describe Saved SQL History

No such department no

1 row(s) updated.

0.02 seconds


4. Excepții de prindere definite de utilizator

Instructiunea RAISE

Se poate folosi instructiunea **RAISE** pentru a ridica o exceptie denumita. Putem ridica:

- 1. O exceptie proprie (care este o exceptie definita de utilizator)***

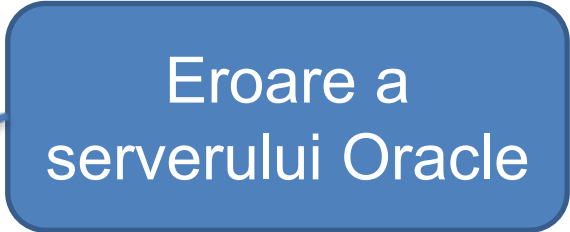
```
IF v_grand_total=0 THEN
    RAISE e_invalid_total;
ELSE
    DBMS_OUTPUT.PUT_LINE(v_num_students/v_gr
and_total);
END IF;
```



4. Excepții de prindere definite de utilizator

2. O eroare a serverului Oracle

```
IF v_grand_total=0 THEN
    RAISE ZERO_DIVIDE;
ELSE
    DBMS_OUTPUT.PUT_LINE(v_num_students/v_gr
and_total);
END IF;
```



Eroare a serverului Oracle

4. Excepții de prindere definite de utilizator

Procedura RAISE_APPLICATION_ERROR

- Puteti folosi procedura **RAISE_APPLICATION_ERROR** pentru a *returna mesaje de eroare definite de utilizator din subprogramele stocate.*
- Principalul avantaj al folosirii acestei proceduri in locul instructiunii **RAISE** este faptul ca procedura permite sa asociati exceptiei propriul numar de eroare si mesajul semnificativ.
- Numerele de eroare trebuie sa se regaseasca in intervalul [-20999,-20000].

Sintaxa

```
RAISE_APPLICATION_ERROR  
error_number,(message[, {TRUE |  
FALSE}]);
```

- **error_number** – este un numar specificat de utilizator pentru exceptie
- **message** – este un mesaj specificat de utilizator pentru exceptie. Este un sir de caractere lung pana la 2048 bytes.

- **TRUE / FALSE** – este un parametru de tip boolean optional (daca este **TRUE** atunci eroarea este plasata pe stiva erorilor anterioare, daca este **FALSE**, iar aceasta este valoarea implicita, eroarea inlocuieste toate erorile anterioare)
- Domeniul de valori [-20999,-20000] este rezervat de **Oracle** pentru folosirea lor de catre programator si nu este niciodata folosit pentru erorile predefinite ale serverului **Oracle**.

4. Excepții de prindere definite de utilizator

Procedura **RAISE_APPLICATION_ERROR** se poate folosi în două locuri diferite:

1. **Sectiunea executabila**
2. **Sectiunea pentru exceptii**

4. Excepții de prindere definite de utilizator

Exemplu - Procedura RAISE_APPLICATION_ERROR in sectiunea executabila

Atunci cand este apelata, procedura afiseaza utilizatorului numarul de eroare si mesajul.

```
DECLARE
  v_mgr PLS_INTEGER := 123;
BEGIN
  DELETE FROM emp
  WHERE mgr = v_mgr;
  IF SQL%NOTFOUND THEN
    RAISE_APPLICATION_ERROR(-20202,'This is not a
    valid manager');
  END IF;
END;
```

Language SQL ? Rows 10 Clear Command Find Tables    A::

```
1 DECLARE
2   v_mgr PLS_INTEGER := 123;
3 BEGIN
4   DELETE FROM emp
5   WHERE mgr = v_mgr;
6   IF SQL%NOTFOUND THEN
7     RAISE_APPLICATION_ERROR(-20202,'This is not a valid manager');
8   END IF;
9 END;
10
```

Results Explain Describe Saved SQL History

```
ORA-20202: This is not a valid manager
ORA-06512: at line 7
```

```
5.     WHERE mgr = v_mgr;
6.     IF SQL%NOTFOUND THEN
7.         RAISE_APPLICATION_ERROR(-20202,'This is not a valid manager');
8.     END IF;
9. END;
```

Exemplu - RAISE_APPLICATION_ERROR in sectiunea pentru exceptii

◦ **DECLARE**

v_mgr PLS_INTEGER := 27;

v_employee_no emp.empno%TYPE;

BEGIN

SELECT empno INTO v_employee_no

FROM emp

WHERE mgr = v_mgr;

**DBMS_OUTPUT.PUT_LINE('The employee who works for
manager_id '||v_mgr||' is: '||v_employee_no);**

EXCEPTION

WHEN NO_DATA_FOUND THEN

**RAISE_APPLICATION_ERROR (-20201,'This manager has
no employees');**

WHEN TOO_MANY_ROWS THEN

**RAISE_APPLICATION_ERROR (-20202,'Too many
employees were found.');**

END;

Language SQL ? Rows 10 ? Clear Command Find Tables

↶ ↷ 🔍 ↵ A::

```
1 DECLARE
2     v_mgr PLS_INTEGER := 27;
3     v_employee_no emp.empno%TYPE;
4 BEGIN
5     SELECT empno INTO v_employee_no
6     FROM emp
7     WHERE mgr = v_mgr;
8     DBMS_OUTPUT.PUT_LINE('The employee who works for manager_id '||v_mgr||' is: '||v_employee_no);
9 EXCEPTION
10    WHEN NO_DATA_FOUND THEN
11        RAISE_APPLICATION_ERROR (-20201,'This manager has no employees');
12    WHEN TOO_MANY_ROWS THEN
13        RAISE_APPLICATION_ERROR (-20202,'Too many employees were found.');
```

Results Explain Describe Saved SQL History

```
ORA-20201: This manager has no employees
ORA-06512: at line 11
```

```
1. DECLARE
2.     v_mgr PLS_INTEGER := 27;
3.     v_employee_no emp.empno%TYPE;
```

Exemplu - RAISE_APPLICATION_ERROR cu o exceptie definita de utilizator

DECLARE

e_name EXCEPTION;

PRAGMA EXCEPTION_INIT (e_name, -20999);

v_last_name emp.ename%TYPE := 'Silly Name';

BEGIN

DELETE FROM emp WHERE ename = v_last_name;

IF SQL%ROWCOUNT = 0 THEN RAISE_APPLICATION_ERROR(-20999,'Invalid last name');

ELSE DBMS_OUTPUT.PUT_LINE(v_last_name||' deleted');

END IF;

EXCEPTION WHEN e_name THEN DBMS_OUTPUT.PUT_LINE ('Valid last names are: ');

FOR c1 IN (SELECT DISTINCT ename FROM emp)

LOOP

DBMS_OUTPUT.PUT_LINE(c1.ename);

END LOOP;

WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Error deleting from employees');

END;

Language SQL

Rows 10

Clear Command

Find Tables

 A::

```
1 DECLARE
2     e_name EXCEPTION;
3     PRAGMA EXCEPTION_INIT (e_name, -20999);
4     v_last_name emp.ename%TYPE := 'Silly Name';
5 BEGIN
6     DELETE FROM emp WHERE ename = v_last_name;
7     IF SQL%ROWCOUNT =0 THEN RAISE_APPLICATION_ERROR(-20999,'Invalid last name');
8     ELSE DBMS_OUTPUT.PUT_LINE(v_last_name||' deleted');
9     END IF;
10    EXCEPTION WHEN e_name THEN DBMS_OUTPUT.PUT_LINE ('Valid last names are: ');
11    FOR c1 IN (SELECT DISTINCT ename FROM emp)
12    LOOP
13        DBMS_OUTPUT.PUT_LINE(c1.ename);
14    END LOOP;
15    WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Error deleting from employees');
16 END;
```

Results

Explain

Describe

Saved SQL

History

Valid last names are:

JONES
WARD
ADAMS
KING
MARTIN
CLARK
ALLEN
TURNER
SCOTT
SMITH
BLAKE
FORD
JAMES
MILLER

1 row(s) deleted.

0.02 seconds



Întrebări?