Tehnici de programare cu baze de date

#7 PL/SQL
Excepţii în PL/SQL

Adrian Runceanu
www.runceanu.ro/adrian

2022

Curs 7

Excepţii în PL/SQL

Excepţii în PL/SQL

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

1. Manipularea excepţiilor

- 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 excepţiilor

Ce este o exceptie?

- *O exceptie apare atunci cand este descoperita o eroare in timpul executiei unui program si care perturba functionarea normala a programului.*

Sunt multe cauze posibile ale exceptiilor:

1. Un utilizator face o greseala de ortografie in timp ce tasteaza
2. Un program nu functioneaza corect
3. O pagina web de publicitate nu exista, etc.

1. Manipularea excepţiilor

- Atunci cand codul nu functioneaza cum era de asteptat, PL/SQL produce o exceptie.
- Atunci cand este produsa (provocata) o exceptie, restul codului din partea executabila a blocului PL/SQL nu se mai executa.

1. Manipularea excepţiilor

Ce este un manipulator de exceptii?

- *Un manipulator de exceptii este un cod care defineste actiunile de recuperare care trebuie executate atunci cand se produce o exceptie.*
- In timpul scrierii unui cod, programatorii trebuie sa anticipeze tipurile de erori care pot aparea in timpul executiei codului respectiv.
- Este necesara includerea manipulatorilor de exceptie in cod pentru a aborda aceste erori.
- Intr-un fel, manipulatorii de exceptie permit programatorilor sa-si protejeze codul.

1. Manipularea excepţiilor

De ce tipuri de erori ar trebui sa tina seama programatorii prin folosirea unor subprograme de tratare a exceptiilor (manipulatorii de exceptie) ?

1. Erori de sistem (de exemplu hard-diskul este plin)
2. Erori de date (de exemplu incercarea 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 excepţiilor

De ce este important manipulatorul de exceptii?

- Protejarea de erori a utilizatorilor (erorile frecvente pot fi frustrante pentru utilizatori si/sau pot determina iesirea utilizatorului din aplicatie)
- Protejarea de erori a bazei de date (datele pot fi pierdute sau suprascrise)
- Erorile importante iau mult din resursele sistemului (daca se face o greseala corectarea acesteia poate fi costisitoare – utilizatorii pot solicita frecvent serviciul de asistenta pentru erori)
- Codul este mai usor de citit deoarece rutinele manipulatorului de eroare pot fi scrise in acelasi bloc in care a aparut eroarea.

1. Manipularea excepţiilor

Manipularea exceptiilor cu PL/SQL

- Un bloc intotdeauna se incheie atunci cand PL/SQL produce o exceptie, dar putem specifica un manipulator de exceptie pentru efectuarea actiunilor finale inainte de incheierea blocului.

- Sectiunea exceptiei incepe cu cuvantul cheie ***EXCEPTION***.

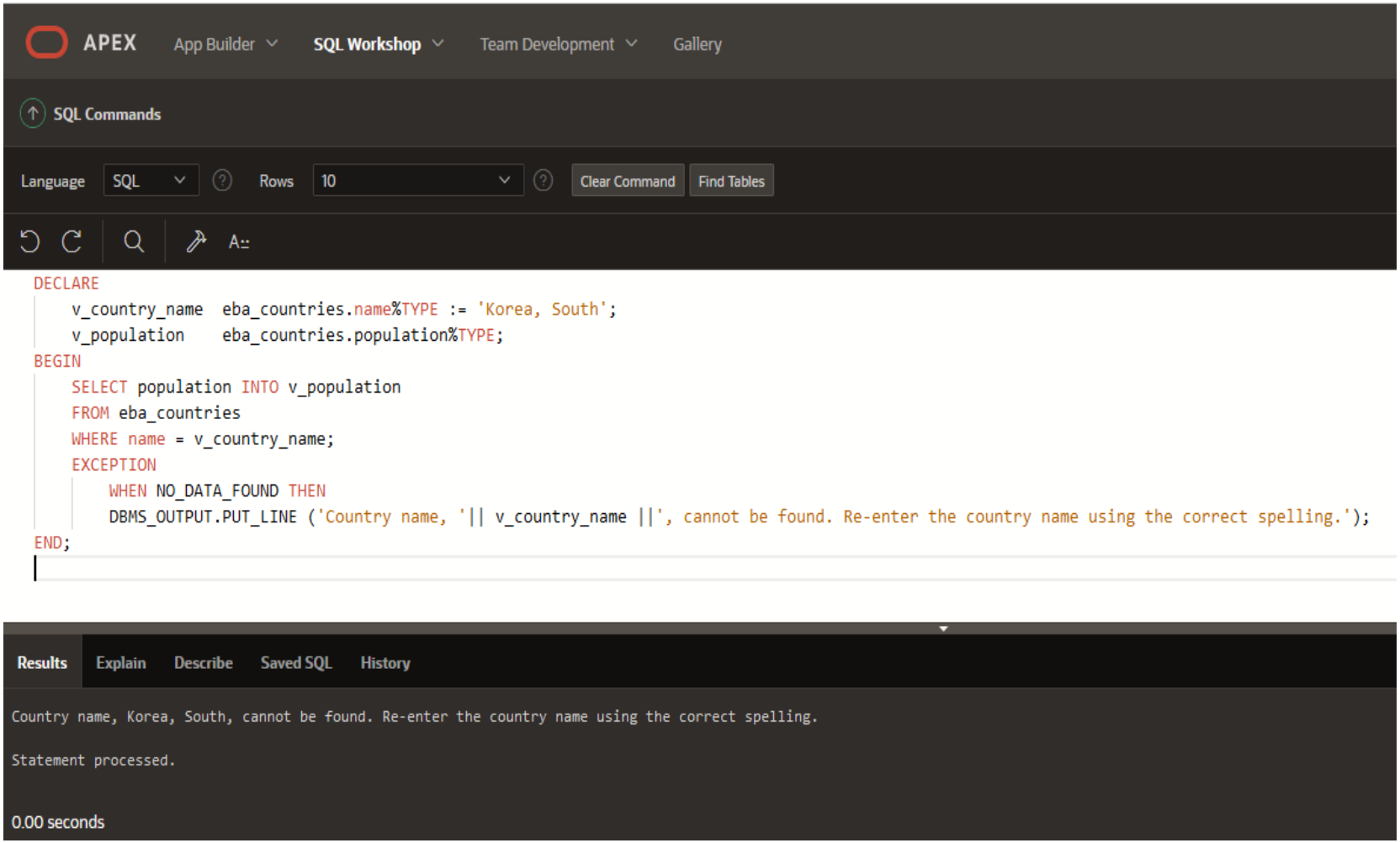
1. Manipularea excepţiilor

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 excepţiilor

Executia codului propus:



1. Manipularea excepţiilor

- Atunci cand este manipulata o exceptie, programul PL/SQL nu se incheie brusc.
- *Atunci cand se produce exceptia, controlul se transfera sectiunii exceptiei si este executat manipulatorul de exceptie din acea sectiune.*
- Blocul PL/SQL se incheie in mod obisnuit si se finalizeaza cu succes.
- Poate aparea o singura exceptie la un moment dat.
- Atunci cand apare o exceptie, PL/SQL prelucreaza un singur manipulator inaintea iesirii 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:

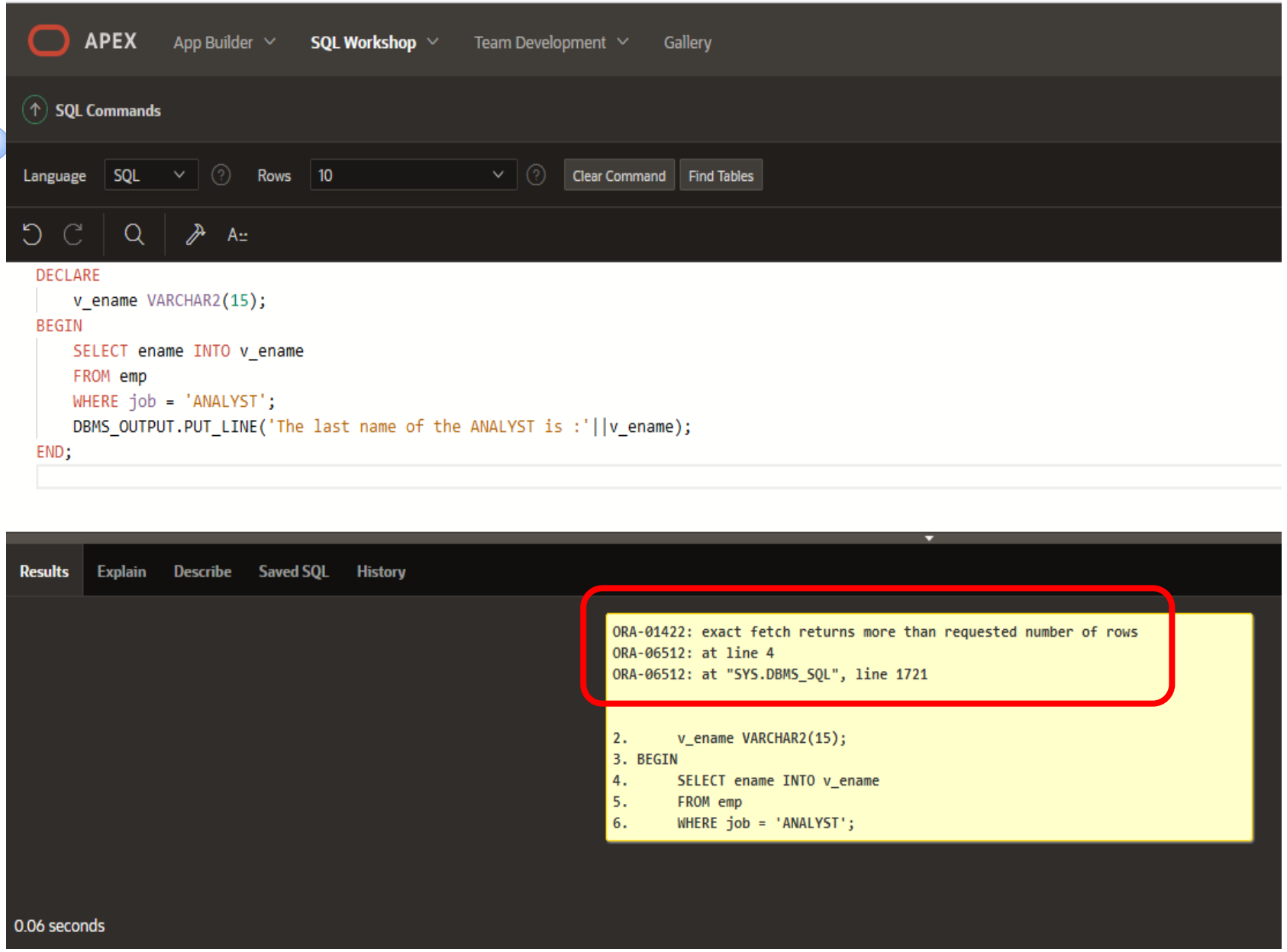
Codul de la punctul A nu se executa deoarece instructiunea SELECT esueaza.

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**.

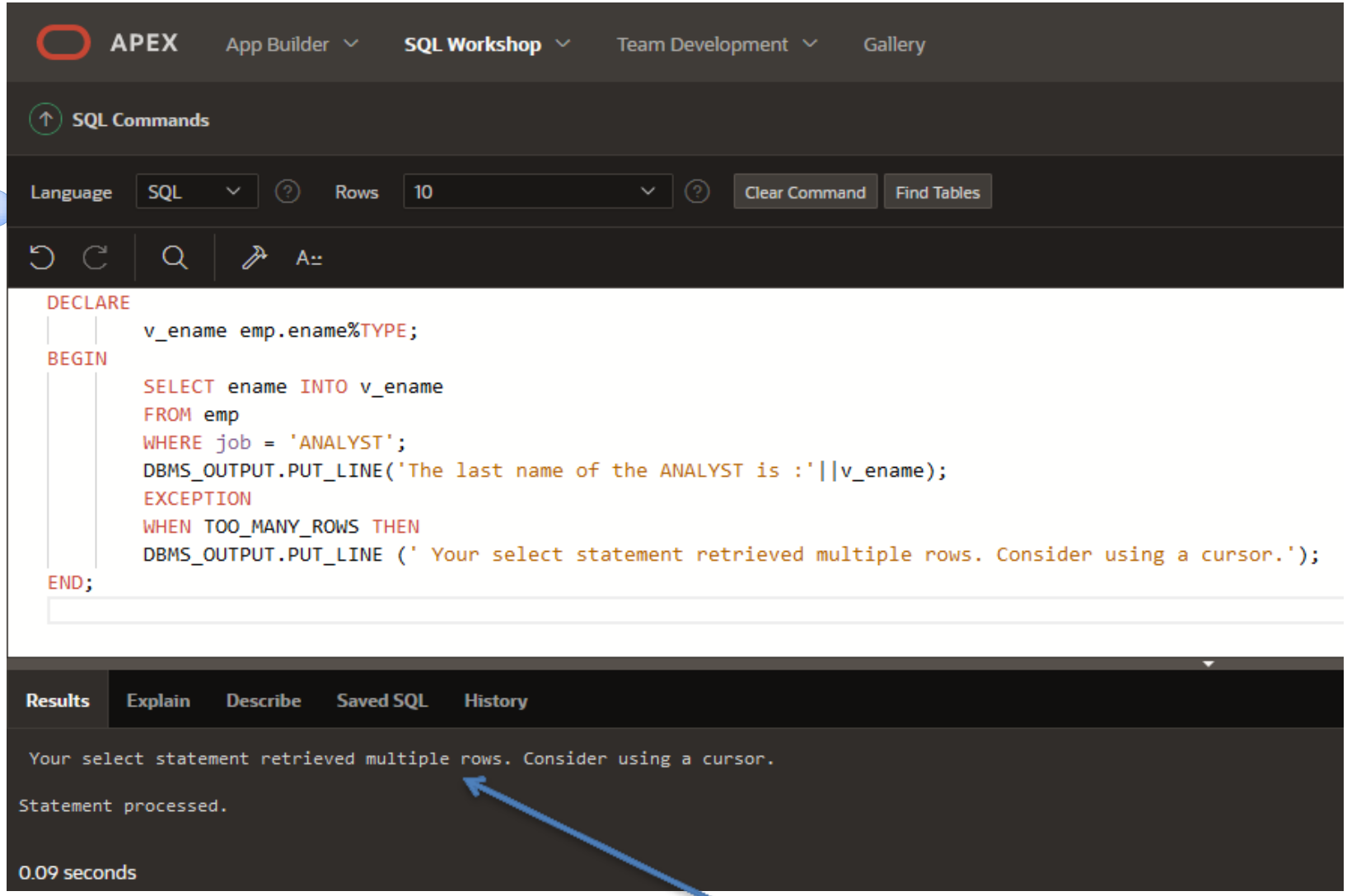
Oricum, se produce o exceptie deoarece exista mai multe date pentru **ANALYST**



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**.



Eroare (exceptie) care intrerupe executia interogarii

Excepţii în PL/SQL

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

2. „Prinderea în capcană” a excepţiilor

Putem manipula sau „prinde in capcana” orice eroare prin includerea unui manipulator corespunzator in sectiunea de manipulare a exceptiilor 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 excepţiilor

- *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 manipulatoare 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 excepţiilor

```
EXCEPTION
WHEN exception1 [OR exception2. . .]
THEN
        Instructiune1;
        Instructiune2;
        …
[WHEN OTHERS THEN
        Instructiune1;
        Instructiune2;
        …]
```

2. „Prinderea în capcană” a excepţiilor

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 excepţiilor

Manipulatorul de exceptii OTHERS

- Sectiunea de manipulare a exceptiilor prinde doar acele exceptii care sunt specificate.
- Orice alte exceptii nu sunt prinse pana nu folosim manipulatorul de exceptii OTHERS.
- Manipulatorul OTHERS prinde toate exceptiile care nu au fost deja prinse.
- Atunci cand este folosit, OTHERS trebuie sa fie ultimul manipulator de exceptie care este definit.

2. „Prinderea în capcană” a excepţiilor

Fie urmatorul exemplu:

1. Daca programul intalneste exceptia NO_DATA_FOUND atunci sunt executate instructiunile din manipulatorul corespunzator.
2. La fel se intampla si daca este intalnita exceptia TOO_MANY_ROWS.
3. Daca sunt intalnite alte exceptii atunci sunt executate instructiunile manipulatorului OTHERS.

2. „Prinderea în capcană” a excepţiilor

```
EXCEPTION
WHEN NO_DATA_FOUND THEN
        Instructiune1;
        …
WHEN TOO_MANY_ROWS THEN
        Instructiune2;
        …
WHEN OTHERS THEN
        Instructiune3;
```

2. „Prinderea în capcană” a excepţiilor

Reguli pentru prinderea exceptiilor

1. Intotdeauna adaugati manipulatoare 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 excepţiilor

Reguli pentru prinderea exceptiilor

3. Folositi manipulatoare 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

Excepţii în PL/SQL

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

3. Prinderea excepţiilor serverului Oracle

- Manipularea erorilor PL/SQL este flexibila si permite programatorilor sa foloseasca:
  - erori definite de serverul Oracle
  - cat si cele definite de programator
- Erorile predefinite sunt erori Oracle obisnuite pentru care PL/SQL are denumiri de exceptii predefinite.
- Erorile care nu sunt predefinite ne determina sa folosim codurile de erori si mesajele **ORA**.
- Sintaxa este diferita pentru fiecare in parte, dar putem prinde ambele tipuri de erori in sectiunea EXCEPTION a blocului PL/SQL.

Tipuri de excepţii

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 excepţiilor serverului Oracle

Manipularea exceptiilor cu PL/SQL

Sunt doua metode pentru a trata o exceptie:

1. Implicit de catre serverul Oracle – Are loc o eroare Oracle si exceptia asociata se produce imediat.

- De exemplu daca are loc eroarea ORA-01403 cand nici un rand nu este preluat din baza de date intr-o instructiune SELECT, atunci PL/SQL produce exceptia NO_DATA_FOUND

3. Prinderea excepţiilor serverului Oracle

2. Explicit de catre programator – depinde de modul de implementare a programului.

- Se pot produce exceptii folosind instructiunea **RAISE** in interiorul blocului.

- Exceptiile produse pot fi atat definite de utilizator cat si predefinite.

3. Prinderea excepţiilor serverului Oracle

Tipuri de erori ale serverului Oracle

Atunci cand au loc erori ale serverului Oracle, automat se produce exceptia asociata, se ignora restul sectiunii executabile a blocului si se cauta un manipulator in sectiunea de exceptie.

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 numar de eroare standard (**ORA-nnnnn**) si un mesaj de eroare, dar nu are un nume predefinit.

- Va puteti declara propriile nume pentru aceste erori astfel incat puteti referi aceste erori in sectiunea de exceptie.

3. Prinderea excepţiilor serverului Oracle

3.1. Capturarea erorilor predefinite ale serverului Oracle

- Numele predefinite sunt referite in rutina de manipulare a exceptiilor

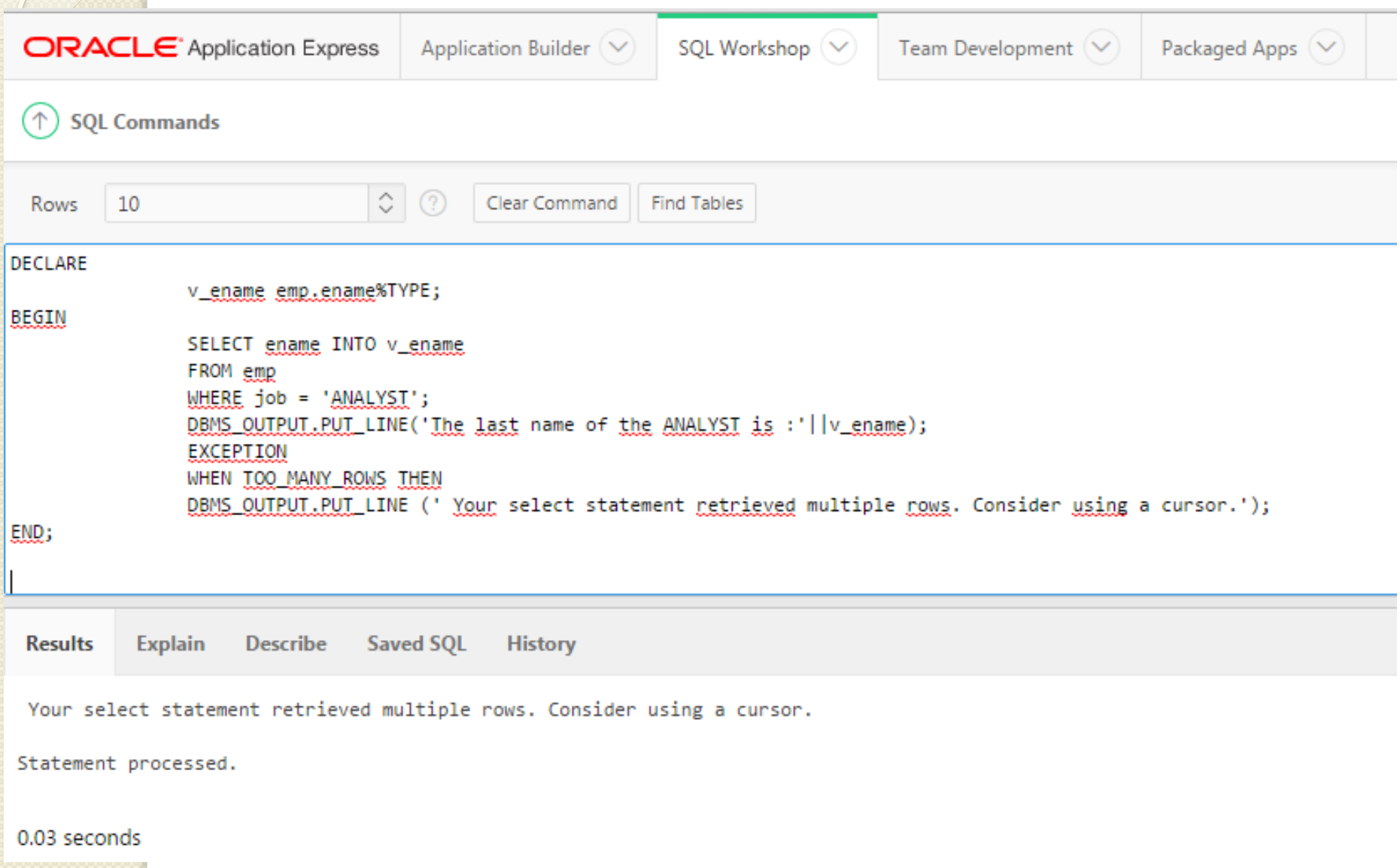
- Exemple de exceptii predefinite:
  1. NO_DATA_FOUND
  2. TOO_MANY_ROWS
  3. INVALID_CURSOR
  4. ZERO_DIVIDE
  5. DUP_VAL_ON_INDEX

3. Prinderea excepţiilor 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;
```



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;
```

ORACLE Application Express | Application Builder | SQL Workshop | Team Development | Packaged Apps

SQL Commands

Rows 10 Clear Command Find Tables

```
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;
```

Results | Explain | Describe | Saved SQL | History

```
Select statement found multiple rows

Statement processed.


0.08 seconds
```

3. Prinderea excepţiilor serverului Oracle

3.2. Capturarea erorilor nepredefinite ale serverului Oracle

- Exceptiile nepredefinite sunt asemanatoare cu cele predefinite, totusi ele nu au nume predefinite in PL/SQL.
- *Sunt erori standard ale serverului Oracle si au numere de eroare ORA.*
- Va creati propriile nume pentru ele in sectiunea DECLARE si asociati aceste nume cu numerele de eroare ORA folosind functia ***PRAGMA EXCEPTION_INIT***.

3. Prinderea excepţiilor 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 excepţii

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 excepţiilor serverului Oracle

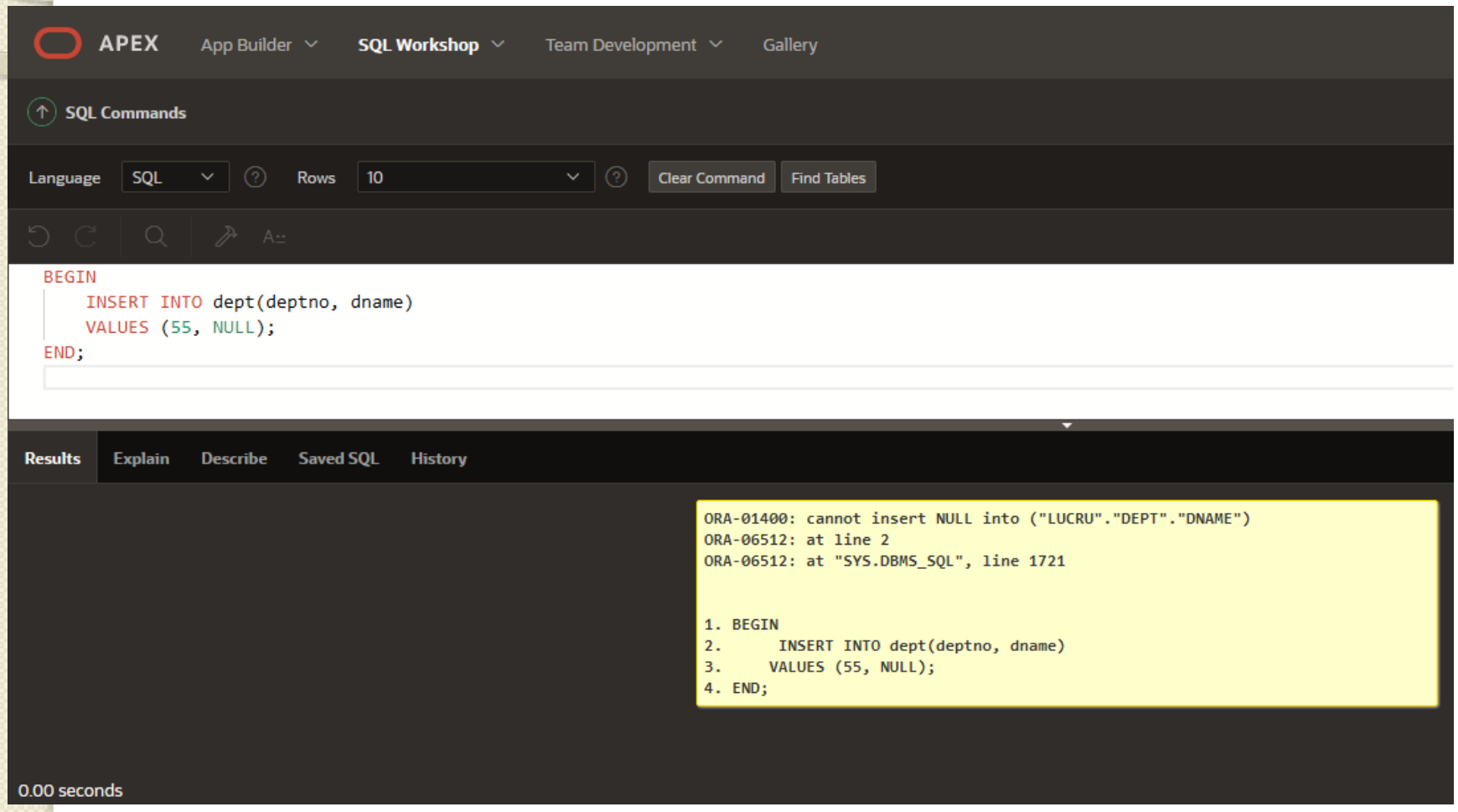
Erori care nu sunt predefinite

Examinati urmatorul exemplu:

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

Se va afisa:

ORA-01400: cannot insert NULL into ("LUCRU". "DEPT"."DNAME")



3. Prinderea excepţiilor serverului Oracle

- Instructiunea ***INSERT*** incearca sa insereze valoarea ***NULL*** pentru coloana ***department_name*** a tabelei ***departments***.
- Totusi, operatia nu are succes deoarece ***department_name*** este o coloana ***NOT NULL***.
- Nu este nici un nume de eroare predefinita pentru incalcarea unei constrangeri ***NOT NULL***.
- Modul de rezolvare a acestei probleme este de a declara propriul nume si a-l asocia cu eroarea ORA-01400.

3. Prinderea excepţiilor 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

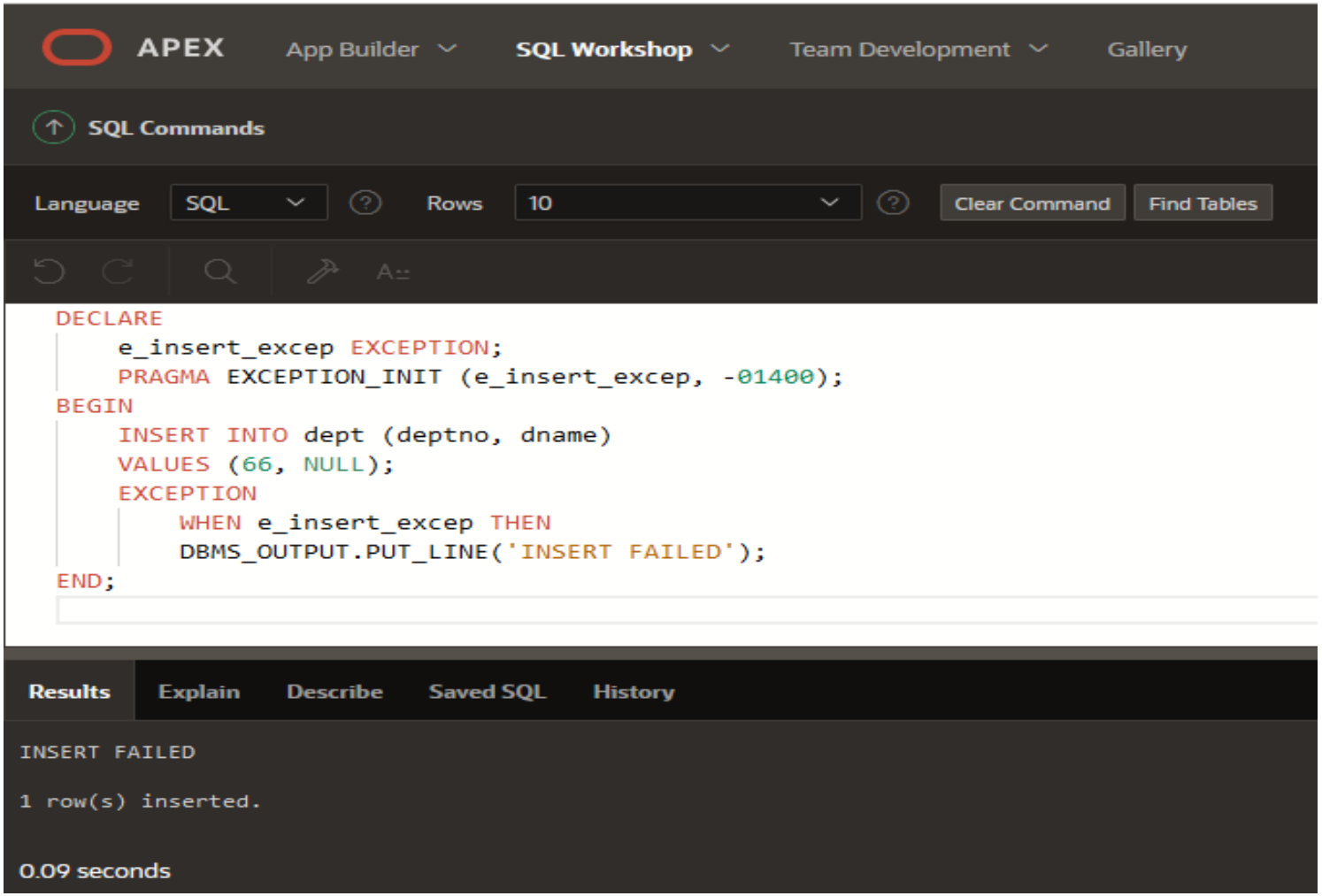
3. Prinderea excepţiilor serverului Oracle

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 excepţiilor serverului Oracle

Executia codului propus:



3. Prinderea excepţiilor serverului Oracle

Functii pentru capturarea exceptiilor

- Atunci cand are loc o exceptie, puteti regasi codul erorii asociate sau un mesaj de eroare prin folosirea a doua functii:

1. ***SQLERRM*** – *returneaza un sir de caractere ce contine mesajul asociat cu numarul erorii*
2. ***SQLCODE*** – *returneaza valoarea numerica pentru codul erorii (O puteti atribui unei variabile de tip NUMBER)*

Pe baza valorilor codului sau pe baza mesajului, puteti decide masurile ulterioare ce trebuie luate.

3. Prinderea excepţiilor 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

- Functiile SQLCODE si SQLERRM nu pot fi folosite direct intr-o instructiune SQL.
- In schimb, trebuie sa atribuiti valorile lor unor variabile locale, apoi folositi acele variabile in instructiuni SQL asa cum va arata urmatorul exemplu:

3. Prinderea excepţiilor 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;
```

Excepţii în PL/SQL

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

4. Excepţii 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. Excepţii 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. Excepţii 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.

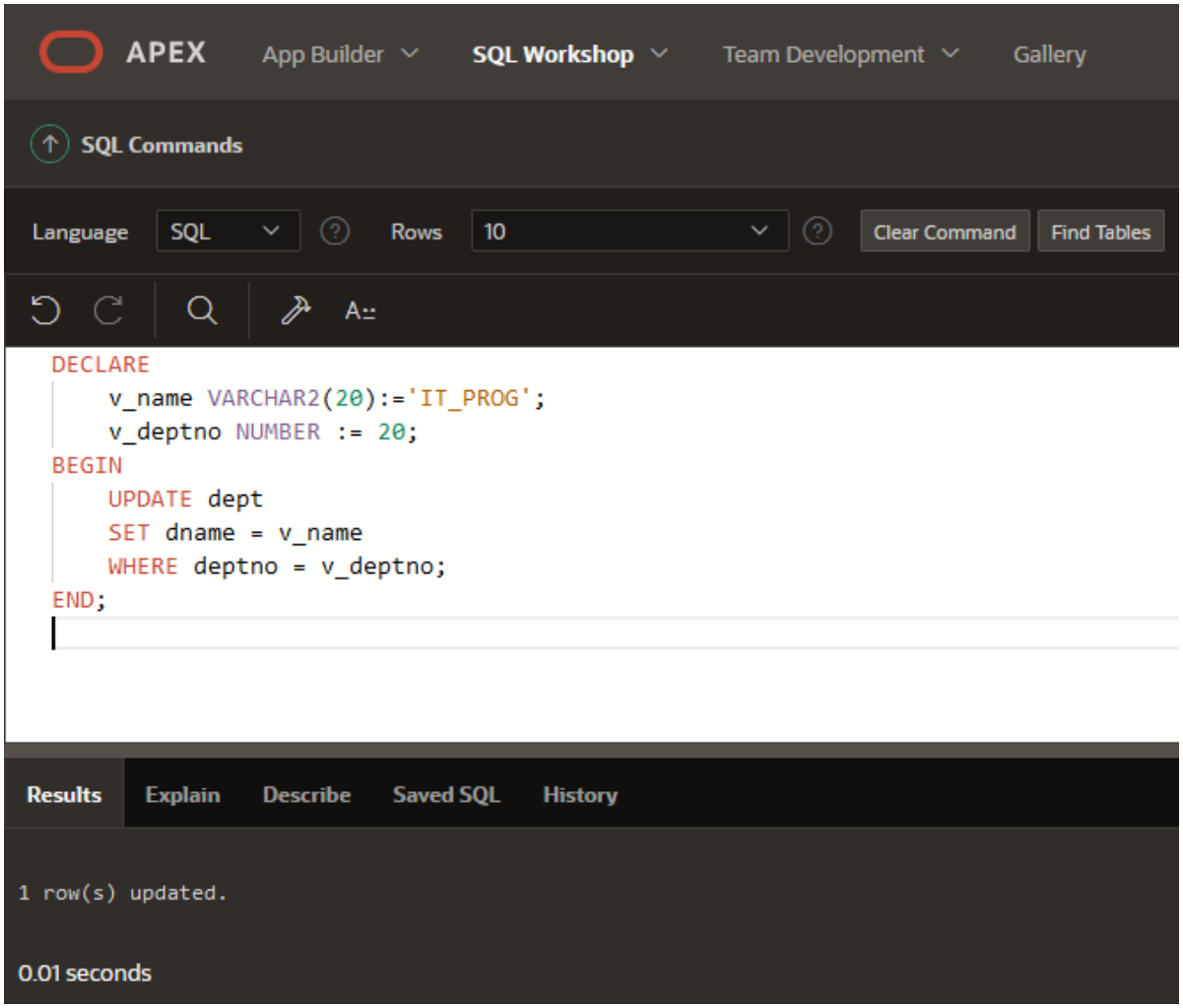
4. Excepţii de prindere definite de utilizator

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;
```

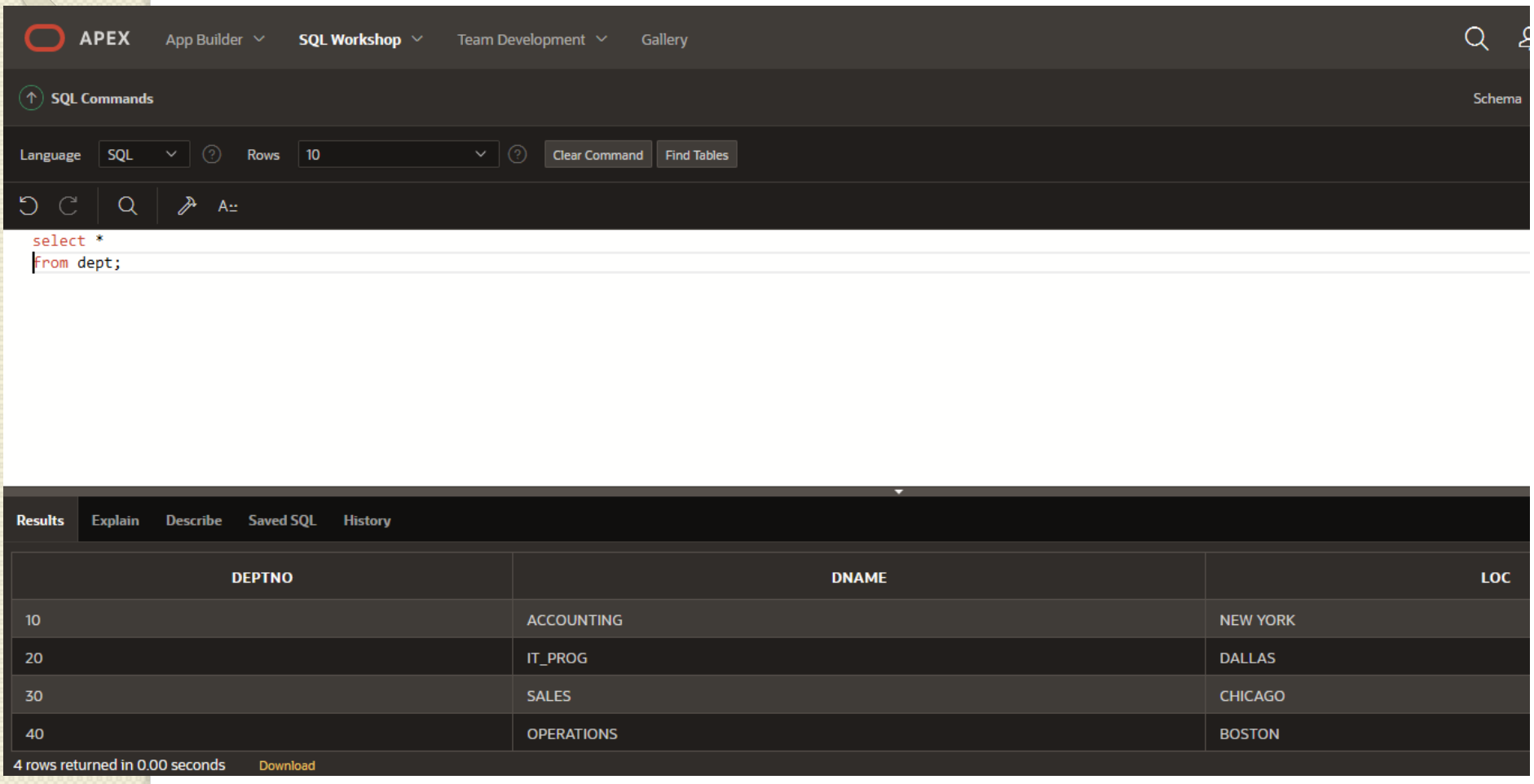
4. Excepţii de prindere definite de utilizator

Executia codului propus:



4. Excepţii de prindere definite de utilizator

Executia codului propus:



4. Excepţii de prindere definite de utilizator

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

4. Excepţii de prindere definite de utilizator

1. Declararea numelui exceptiei definite de utilizator in sectiunea declarativa

e_invalid_department EXCEPTION;

2. Folosirea instructiunii RAISE pentru a produce explicit exceptia in sectiunea executabila.

IF SQL%NOTFOUND THEN RAISE e_invalid_department;

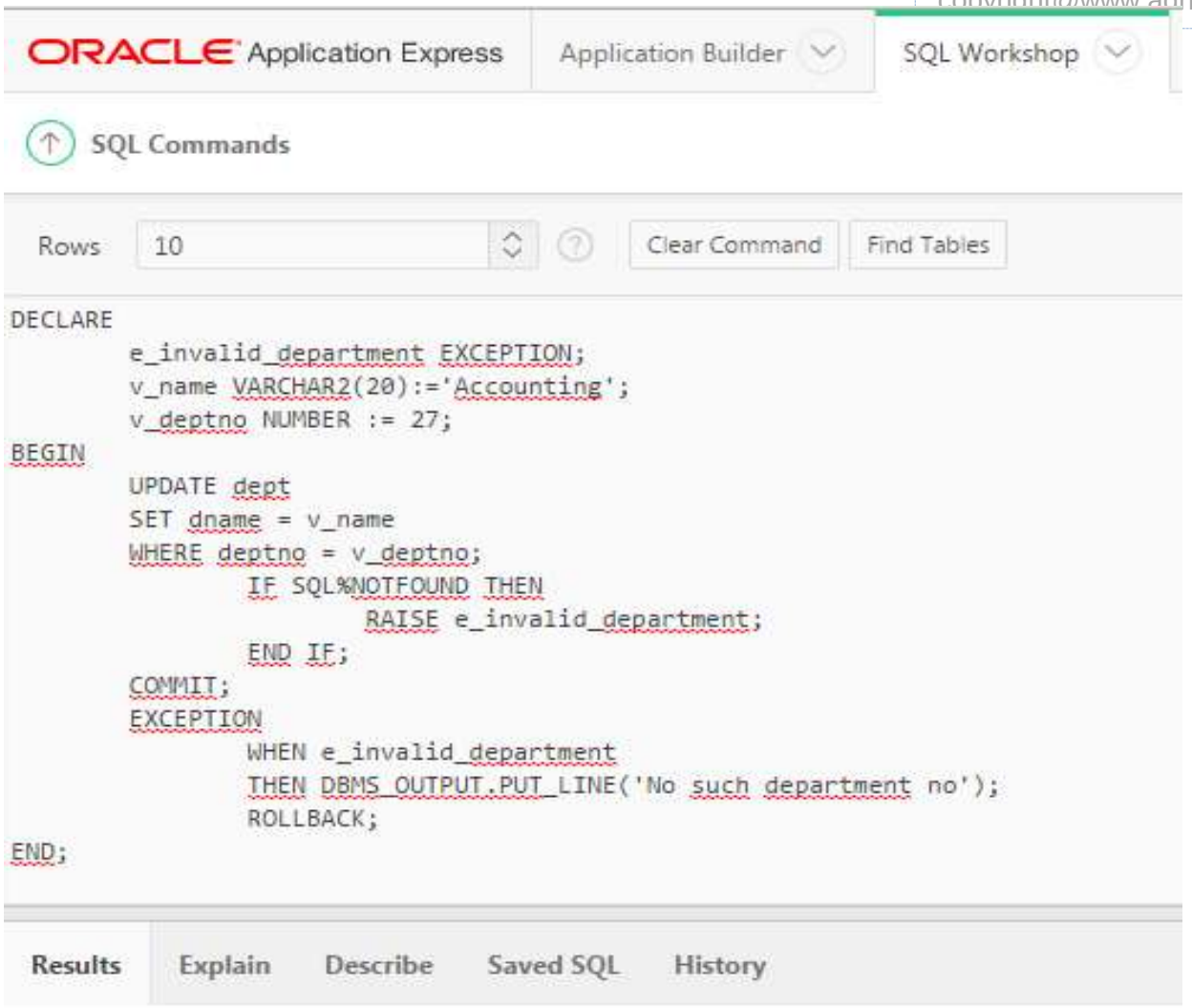
4. Excepţii de prindere definite de utilizator

3. Referirea exceptiei declarate in rutina manipulatorului de exceptie corespunzator

```
EXCEPTION WHEN
    e_invalid_department THEN
DBMS_OUTPUT.PUT_LINE('No such
    department id.');
```

Codul complet

```
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;
```

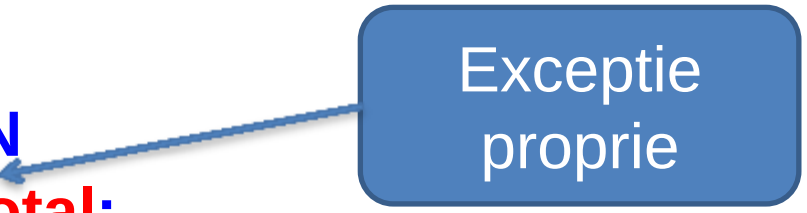


4. Excepţii de prindere definite de utilizator

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. Excepţii 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;
```

4. Excepţii 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. Excepţii de prindere definite de utilizator

Procedura **RAISE_APPLICATION_ERROR** se poate folosi in doua locuri diferite:

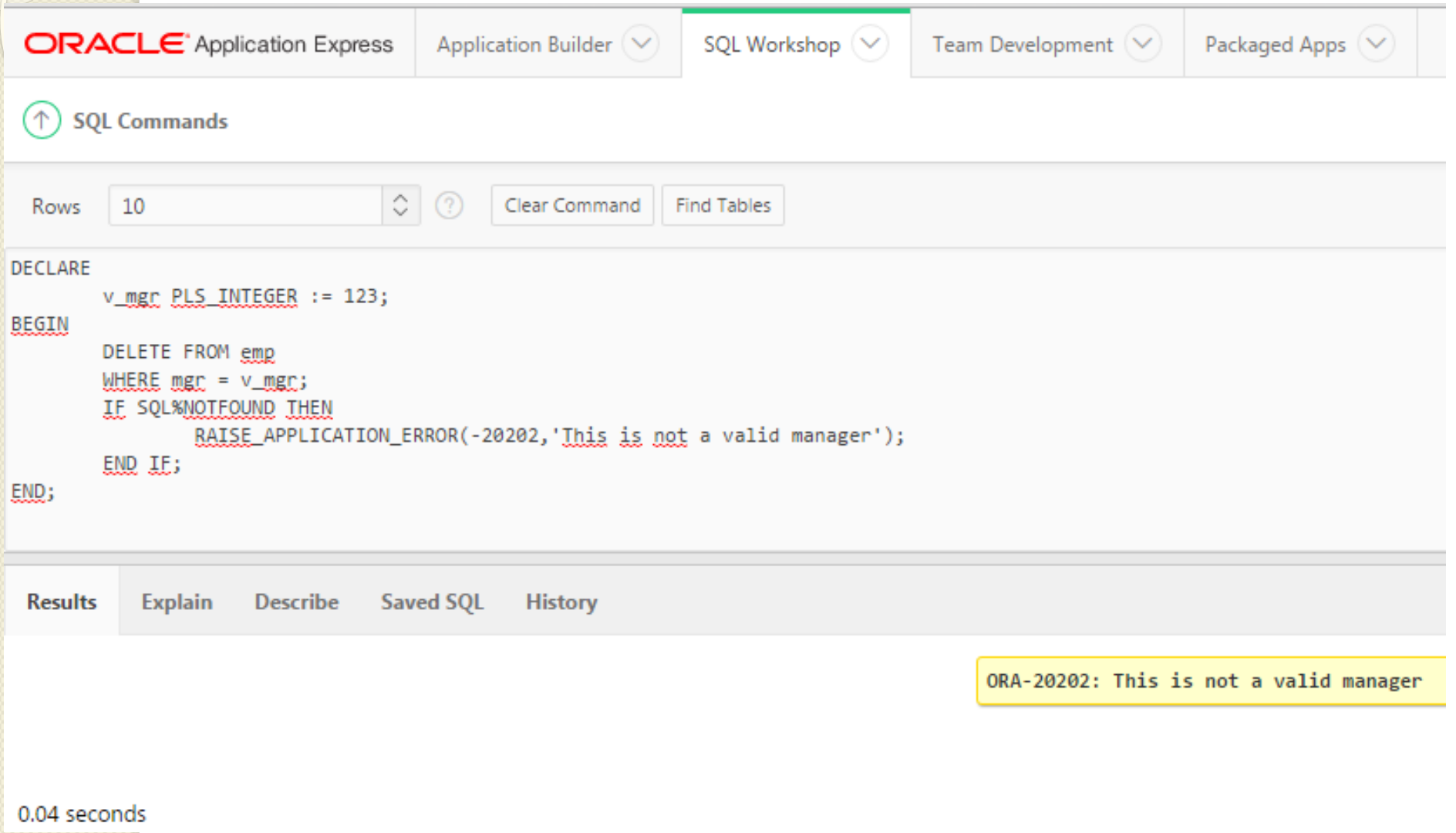
1. **Sectiunea executabila**
2. **Sectiunea pentru exceptii**

4. Excepţii de prindere definite de utilizator

Exemplu - Procedura* RAISE_APPLICATION_ERROR *in sectiunea executabila

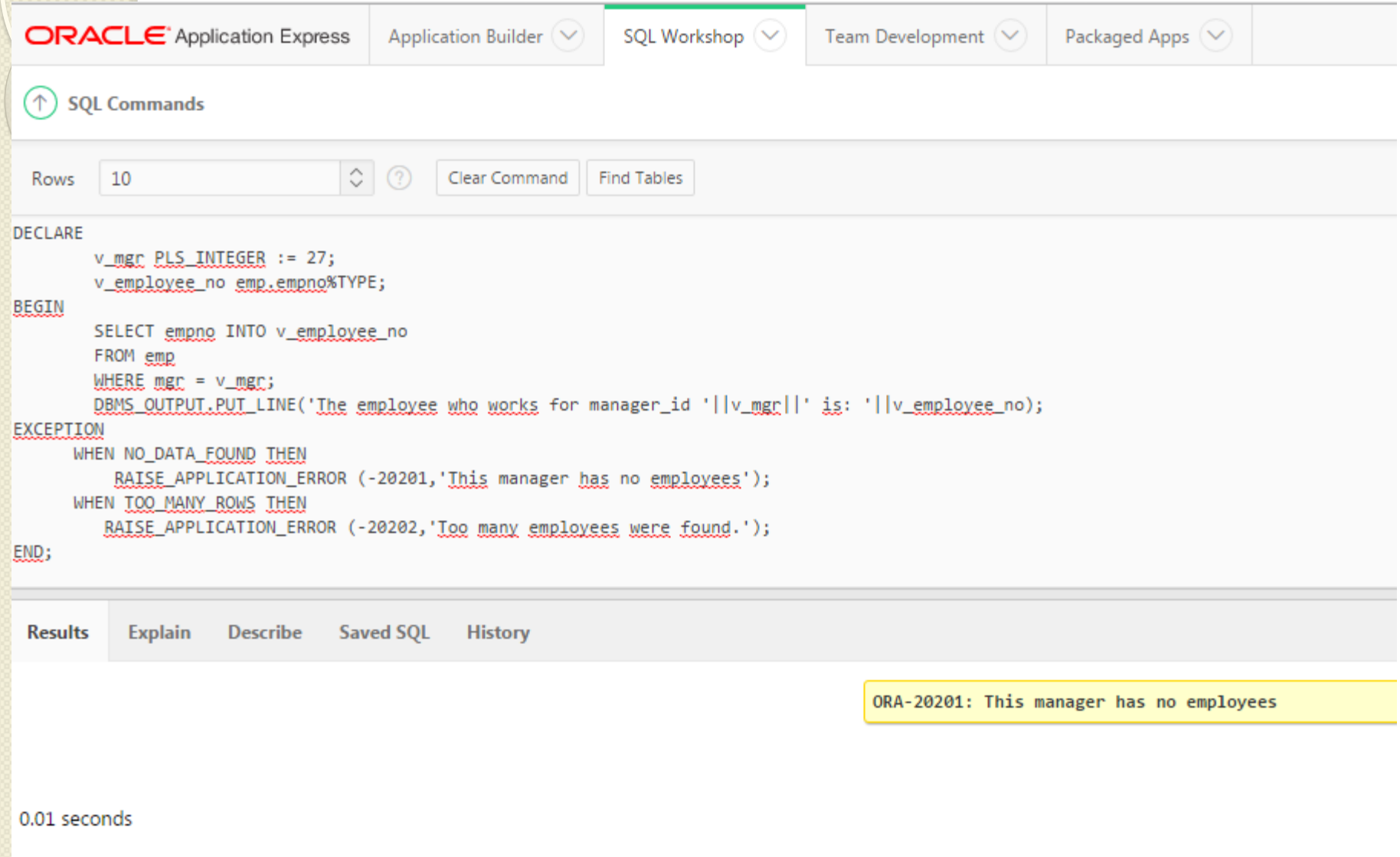
Atunci cand este apelata, procedura afiseaza utilizatorului numarul de eroare si mesajul.

```
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;
```



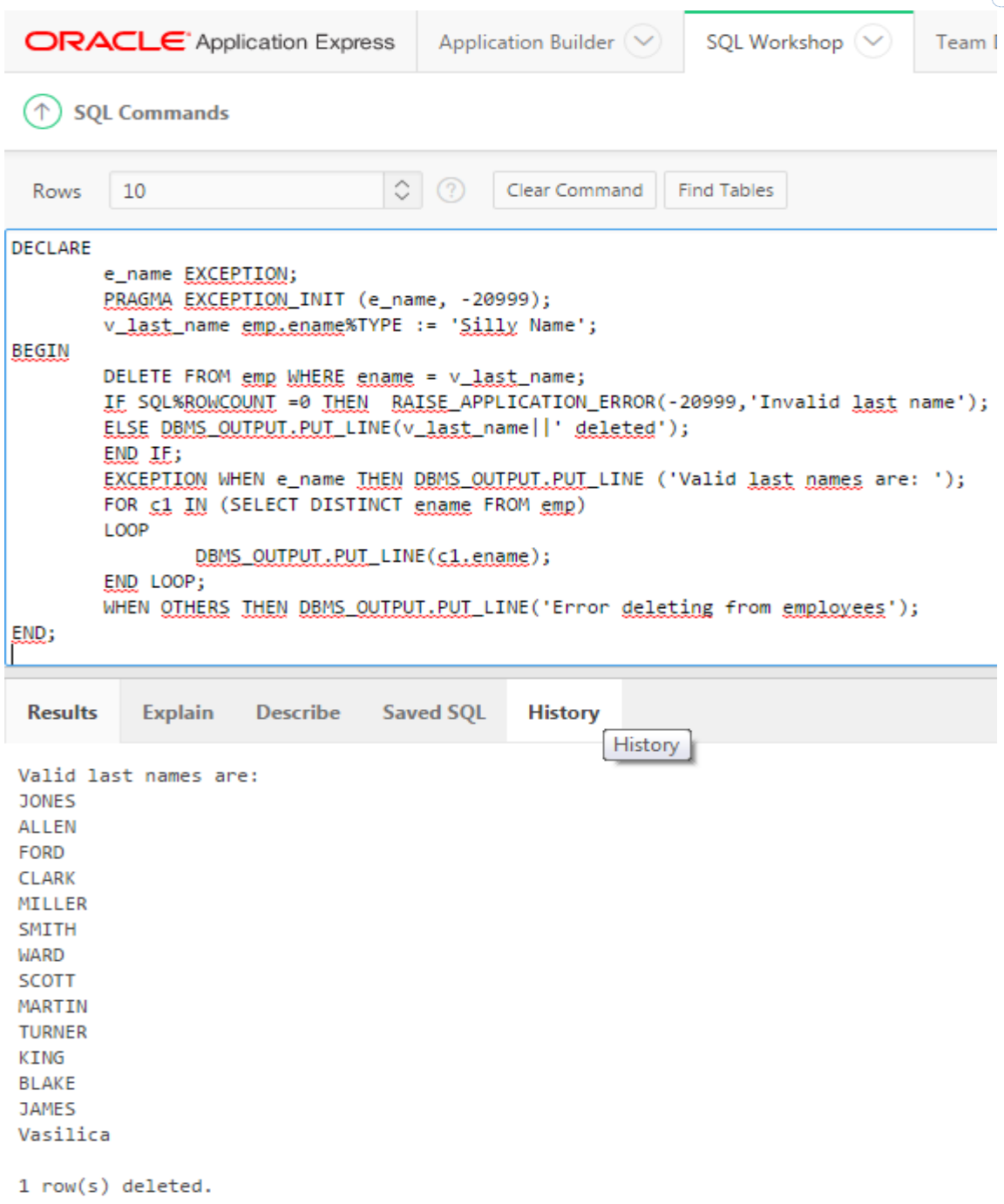
Exemplu - **RAISE_APPLICATION_ERROR *in sectiunea pentru exceptii***

```
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;
```



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;
```



Întrebări?