

PL / SQL

Les exceptions

Ines BAKLOUTI

ines.baklouti@esprit.tn

Ecole Supérieure Privée d'Ingénierie et de Technologies



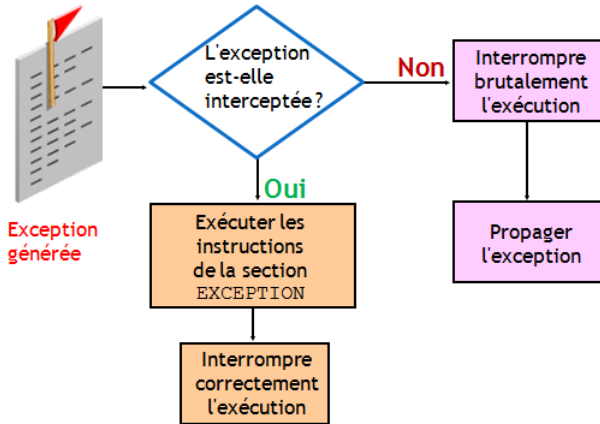
Plan

- 1 Les types d'exceptions
- 2 Les exceptions prédéfinies
- 3 Les exceptions définies par l'utilisateur
 - Redéclaration des exceptions prédéfinies
 - Déclencher des exceptions explicitement
- 4 Propagation des exceptions

Introduction

- Une exception est tout **avertissement** (warning) ou **erreur** détecté lors de la compilation d'un bloc PL/SQL.
- Les exceptions dûes au erreurs de compilation peuvent être levées à cause de défauts syntaxiques, erreurs de codages, et plusieurs autres sources. Vous ne pouvez pas anticiper toutes les exceptions possibles, mais vous pouvez écrire des gestionnaires d'exception qui permettent à votre programme de continuer à fonctionner en leur présence.
- Chaque bloc PL/SQL peut avoir une partie de gestion d'exception, qui peut avoir un ou plusieurs gestionnaire d'exception.

Introduction



Plan

- 1 Les types d'exceptions
- 2 Les exceptions prédéfinies
- 3 Les exceptions définies par l'utilisateur
 - Redéclaration des exceptions prédéfinies
 - Déclencher des exceptions explicitement
- 4 Propagation des exceptions

Les types d'exceptions

- Exception prédéfinie: est une exception définie en interne par PL/SQL et est levée implicitement par le serveur oracle. Par exemple: NO_DATA_FOUND, TOO_MANY_ROWS, etc.
- Exception définie par l'utilisateur: Vous pouvez déclarer vos propres exceptions dans la partie déclarative d'un bloc PL / SQL anonyme, sous-programme (procédure, fonction ou trigger), ou package.

Exception	Possède code erreur	Possède un nom	Levée implicitement	Levée explicitement
Prédéfinie	toujours	toujours	oui	optionnellement
Définie par l'utilisateur	seulement si vous y accordez un code	toujours	non	toujours

- Les fonctions d'interception des exceptions:
 - SQLCODE : renvoie la valeur numérique du code d'erreur
 - SQLERRM : renvoie le message associé au code d'erreur

Plan

- 1 Les types d'exceptions
- 2 Les exceptions prédéfinies
- 3 Les exceptions définies par l'utilisateur
 - Redéclaration des exceptions prédéfinies
 - Déclencher des exceptions explicitement
- 4 Propagation des exceptions

Les exceptions prédéfinies

Syntaxe

DECLARE

/* déclarations */

BEGIN

/* traitements */

EXCEPTION

WHEN exception1 [OR exception2 . . .] THEN
statement1; statement2;

. . .

[WHEN exception3 [OR exception4 . . .] THEN
statement1; statement2;

. . .]

[WHEN OTHERS THEN
statement1; statement2;
. . .]

END;

Les exceptions prédéfinies

- Les exceptions prédéfinies les plus utilisées sont:

Nom de l'exception	Code
NO_DATA_FOUND	+100
TOO_MANY_ROWS	-1422
ZERO_DIVIDE	-1476
DUP_VAL_ON_INDEX	-1
INVALID_CURSOR	-1001
CURSOR_ALREADY_OPEN	-6511
INVALID_NUMBER	-1722
ROWTYPE_MISMATCH	-6504

Les exceptions prédéfinies

Exemple 1

DECLARE

Iname VARCHAR2(15);

BEGIN

SELECT last_name INTO Iname FROM employees

WHERE first_name='John';

DBMS_OUTPUT.PUT_LINE ('Le nom de John est : ' || Iname);

EXCEPTION

WHEN TOO_MANY_ROWS THEN

DBMS_OUTPUT.PUT_LINE ('La requête renvoie plusieurs lignes, utilisez un curseur');

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE ('Aucun employé ne porte le prénom John');

END;

Les exceptions prédéfinies

Exemple 2

```
CREATE OR REPLACE PROCEDURE pr_nom(prenom varchar2) IS
  lname VARCHAR2(15);
BEGIN
  SELECT last_name INTO lname FROM employees
  WHERE first_name=prenom;
  DBMS_OUTPUT.PUT_LINE ('Le nom de '|| prenom||' est : ' ||lname);
EXCEPTION
  WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE ('La requête revoie plusieurs lignes, utilisez un curseur');
  WHEN OTHERS THEN
    ROLLBACK;
    DBMS_OUTPUT.PUT_LINE('code de l'erreur : '||SQLCODE);
    DBMS_OUTPUT.PUT_LINE('message de l'erreur : '||SQLERRM);
END;
```

Plan

- 1 Les types d'exceptions
- 2 Les exceptions prédéfinies
- 3 Les exceptions définies par l'utilisateur**
 - Redéclaration des exceptions prédéfinies
 - Déclencher des exceptions explicitement
- 4 Propagation des exceptions

Redéclaration des exceptions prédéfinies

Exemple

DECLARE

a number:=100;

b number:=0;

division_zero EXCEPTION;

PRAGMA EXCEPTION_INIT(division_zero,-01476);

BEGIN

a:=a/b;

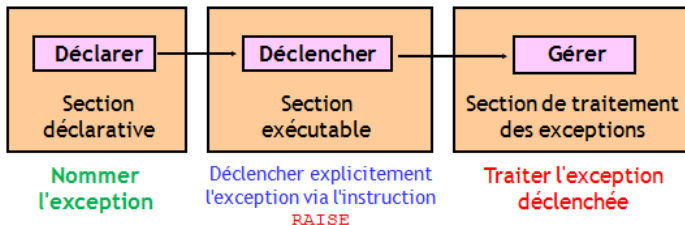
EXCEPTION

WHEN division_zero THEN

DBMS_OUTPUT.PUT_LINE('ERREUR: Division par 0');

END;

Déclencher des exceptions explicitement



Avec le mot clé RAISE

Syntaxe

DECLARE

```
exception_name EXCEPTION;  – déclarer l'exception  
[PRAGMA EXCEPTION_INIT(exception_name,error_code)];
```

...

BEGIN

...

```
RAISE exception_name;  – déclencher l'exception
```

EXCEPTION

```
WHEN exception_name THEN  – gérer l'exception  
– traitement;
```

. . .

```
END;
```

Avec le mot clé RAISE

Exemple

```
CREATE OR REPLACE PROCEDURE pr_check_salary (current_salary number) IS
    salary_too_high EXCEPTION;  -- déclarer l'exception
    PRAGMA EXCEPTION_INIT(salary_too_high,-20100); -- associer optionnellement le code -20100
    à l'exception
    max_salary NUMBER := 10000;
BEGIN
    IF current_salary > max_salary THEN
        RAISE salary_too_high;  -- déclencher l'exception
    END IF;
EXCEPTION
    WHEN salary_too_high THEN  -- gérer l'exception
        DBMS_OUTPUT.PUT_LINE('ERREUR : '||SQLCODE);
        DBMS_OUTPUT.PUT_LINE('MESSAGE : '||SQLERRM);
        DBMS_OUTPUT.PUT_LINE('Salaire '||current_salary||' est très élevé');
        DBMS_OUTPUT.PUT_LINE ('Le salaire maximum est ' || max_salary);
END;
```


Avec la procédure RAISE_APPLICATION_ERROR

Syntaxe

`RAISE_APPLICATION_ERROR (error_number, message);`

Avec:

- `Error_number` représente un entier négatif compris entre -20000 et -20999
 - `message` représente le texte du message d'erreur d'une longueur maximum de 2048 octets
-
- La procédure `RAISE_APPLICATION_ERROR` peut être utilisée à deux endroits :
 - Section exécutable
 - Section de traitement des exceptions

Avec la procédure RAISE_APPLICATION_ERROR

Exemple 1

BEGIN

DELETE FROM employees WHERE manager_id = 500;

IF SQL%NOTFOUND THEN

RAISE_APPLICATION_ERROR(-20200, 'Numéro de manager non valide');

END IF;

END;

Exemple 2

DECLARE

mgr_id NUMBER;

BEGIN

SELECT manager_id INTO mgr_id FROM employees WHERE employee_id=300;

EXCEPTION

WHEN NO_DATA_FOUND THEN

RAISE_APPLICATION_ERROR (-20999, 'Numéro d"employé non valide');

END;

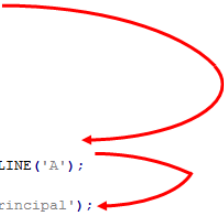
Plan

- 1 Les types d'exceptions
- 2 Les exceptions prédéfinies
- 3 Les exceptions définies par l'utilisateur
 - Redéclaration des exceptions prédéfinies
 - Déclencher des exceptions explicitement
- 4 Propagation des exceptions

Propagation des exceptions

Exemple 1

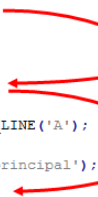
```
DECLARE
x number:=1;
A exception;
B exception;
C exception;
BEGIN
DBMS_OUTPUT.PUT_LINE('Bonjour');
    BEGIN
        DBMS_OUTPUT.PUT_LINE('sous bloc');
        if x=1 then
            raise A;
        elsif x=2 then
            raise B;
        else
            raise C;
        end if;
    EXCEPTION
        when A then
            DBMS_OUTPUT.PUT_LINE('A');
    END;
DBMS_OUTPUT.PUT_LINE('bloc principal');
EXCEPTION
    when B then
        DBMS_OUTPUT.PUT_LINE('B');
END;
```



Propagation des exceptions

Exemple 2

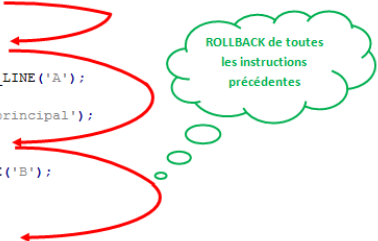
```
DECLARE
x number:=2;
A exception;
B exception;
C exception;
BEGIN
DBMS_OUTPUT.PUT_LINE('Bonjour');
    BEGIN
        DBMS_OUTPUT.PUT_LINE('sous bloc');
        if x=1 then
            raise A;
        elsif x=2 then
            raise B;
        else
            raise C;
        end if;
    EXCEPTION
        when A then
            DBMS_OUTPUT.PUT_LINE('A');
    END;
DBMS_OUTPUT.PUT_LINE('bloc principal');
EXCEPTION
    when B then
        DBMS_OUTPUT.PUT_LINE('B');
END;
```



Propagation des exceptions

Exemple 3

```
DECLARE
x number:=3;
A exception;
B exception;
C exception;
BEGIN
DBMS_OUTPUT.PUT_LINE('Bonjour');
  BEGIN
    DBMS_OUTPUT.PUT_LINE('sous bloc');
    if x=1 then
      raise A;
    elsif x=2 then
      raise B;
    else
      raise C;
    end if;
  EXCEPTION
    when A then
      DBMS_OUTPUT.PUT_LINE('A');
  END;
DBMS_OUTPUT.PUT_LINE('bloc principal');
EXCEPTION
  when B then
    DBMS_OUTPUT.PUT_LINE('B');
END;
```



ROLLBACK de toutes
les instructions
précédentes