Cours 3 : Les exceptions PL/SQL



Auteur : Riadh ZAAFRANI

2ème année SI – 2ème eservices
Mars 2010

Présentation des exceptions

- Le mécanisme de gestion d'erreurs dans **PL/SQL**, est appelé gestionnaire des exceptions.
- Le traitement des **exceptions PL/SQL** permet au développeur de planifier sa gestion et d'abandonner le traitement en présence d'une erreur ou de le continuer si l'erreur n'est pas suffisamment importante.

Présentation des exceptions

- Comme on le sait, un bloc PL/SQL est divisé de quatre parties : l'en-tête, la section de déclaration, la section d'exécution et la section d'exception.
- Lorsqu'une exception est déclanchée dans la section d'exécution d'un bloc **PL/SQL**, la section d'exception « **EXCEPTION** » prend le contrôle.
- **PL/SQL** vérifie si, parmi les différents gestionnaires d'exception, l'un traite cette exception spécifique.

Présentation des exceptions

- La syntaxe d'une section d'exception est la suivante :
- **EXCEPTION**
  WHEN NOM_EXCEPTION [OR NOM_EXCEPTION …]
  THEN
  INSTRUCTIONS ;
  …
  [WHEN OTHERS THEN
  INSTRUCTIONS ;]
  END;
- Une section d'exception unique peut contenir plusieurs gestionnaires d'exception. Les gestionnaires d'exceptions ont une structure comparable à celle de l'ordre conditionnel **CASE**.

Présentation des exceptions

```
SQL> begin
  2    declare
  3      v_nom_produit   Produit.nomproduit%TYPE;
  4    begin
  5      SELECT nomproduit INTO v_nom_produit FROM produit
  6        WHERE code_catégorie = 3;
  7        dbms_output.put_line('Vous ne verrez pas cette ligne !!!');
  8    end;
  9    dbms_output.put_line('Suite de traitements.');
 10  end;
 11  /
Declare
* ERREUR à la ligne 1 :
ORA-01403: Aucune donnée trouvée
ORA-06512: à ligne 5
```

Présentation des exceptions

- Dans cette première requête, il n'y a pas de gestionnaire d'exception.
- Quand l'erreur survient (il n'y a pas de produit de la catégorie 3), le programme est arrêté, affichant un message d'erreur.

Présentation des exceptions

```
SQL> begin
  2    declare
  3      v_nom_produit   Produit.nomproduit%TYPE;
  4    begin
  5      SELECT  nomproduit INTO v_nom_produit FROM produit
  6        WHERE code_catégorie  = 3;
  7        dbms_output.put_line('Vous ne verrez pas cette ligne !!!');
  8    exception
  9     When  NO_DATA_FOUND  Then
 10       dbms_output.put_line('Aucune  catégorie n''a été trouvée.');
 11     end;
 12     dbms_output.put_line('Suite  de  traitements.');
 13  end;
 14  /
```

Aucune catégorie n'a été trouvée.
Suite de traitements.

Présentation des exceptions

- Cette deuxième requête assure le traitement d'une exception **NO_DATA_FOUND**. Après le traitement de cette exception on ne revient pas dans le bloc ayant généré l'exception, on quitte complètement le bloc mais le programme continue normalement.
- Notez que la clause WHEN traite des exceptions nommées. Si aucun gestionnaire ne correspond à l'exception déclenchée, les ordres associés à la clause **WHEN OTHERS** sont exécutés si elle est présente.

Les exceptions prédéfinies

- Toutes les erreurs possèdent un **numéro d'identification unique**. Mais elles ne peuvent être interceptées dans un bloc **PL/SQL** que si un nom est associé au numéro de l'erreur Oracle. Les exceptions prédéfinies sont déclarées dans le package **STANDARD** de PL/SQL. La liste des exceptions prédéfinies est :
- **ACCESS_INTO_NULL :** On a tenté d'affecter une valeur à un objet non initialisé. **ORA-6530 SQLCODE = -6530**
- **CASE_NOT_FOUND :** Il n'y a pas de choix WHEN correspondant dans une instruction CASE et l'option ELSE n'a pas été définie. **ORA-6592 SQLCODE= -6592**

Les exceptions prédéfinies

- **COLECTION_IS_NULL :** On a tenté d'utiliser des méthodes d'une collection, autre que EXISTS, ou essayé d'affecter une valeur à un élément pour une collection non initialisée. **ORA-6531 SQLCODE = -6531**
- **CURSOR_ALREADY_OPEN :** On a tenté d'ouvrir un curseur qui l'était déjà. Il faut fermer un curseur avant de l'ouvrir ou de le rouvrir. **ORA-6511 SQLCODE = -6511**
- **DUP_VAL_ON_INDEX :** Un ordre **INSERT** ou **UPDATE** a tenté d'insérer un doublon dans une colonne ou un groupe de colonnes soumis à un index unique. **ORA-6592 SQLCODE = -6592**

Les exceptions prédéfinies

- **INVALID_CURSOR :** On a référencé un curseur invalide. Cela arrive lorsque l'on FETCH ou l'on ferme, un curseur avant de l'ouvrir **ORA-01001 SQLCODE = -1001**
- **INVALID_NUMBER :** PL/SQL exécute un ordre qui ne parvient pas à convertir une chaîne de caractères en nombre. **ORA-01722 SQLCODE = -1722**
- **LOGIN_DENIED :** Un programme tente de se connecter à Oracle avec une combinaison login/mot de passe invalide. **ORA-01017 SQLCODE= -1017**

Les exceptions prédéfinies

- **NO_DATA_FOUND :** Cette exception est déclenchée dans 3 cas.
  - Lorsqu'on exécute un ordre **SELECT INTO** qui ne ramène aucun enregistrement.
  - Lorsqu'on référence une ligne non définie d'une table PL/SQL
  - Lorsqu'on tente de lire après la fin d'un fichier avec la package UTL_FILE.
- **ORA-01403 SQLCODE= +100**
- **NOT_LOGGED_ON :** Un programme a tenté d'exécuter un appel à la base, en général un ordre LMD, avant d'être connecté. **ORA-01012 SQLCODE= -1012**

Les exceptions prédéfinies

- **PROGRAM_ERROR :** Erreur interne de PL/SQL. **ORA-06501 SQLCODE=-1012**
- **RAWTYPE_MISMATCH :** On a tenté d'affecter une variable enregistrement incompatible avec l'enregistrement retourné par la commande FETCH. **ORA-06504 SQLCODE = -6504**
- **STORAGE_ERROR :** Le programme a épuisé la mémoire disponible, ou la mémoire est corrompue. **ORA-06500 SQLCODE = -6500**
- **TIMEOUT_ON_RESOURCE :** Le délai maximum d'attente d'une ressource par Oracle a expiré. **ORA-00051 SQLCODE = -51**

Les exceptions prédéfinies

- **TOO_MANY_ROWS :** Un ordre SELECT INTO a ramené plus d'une ligne. **ORA-01422 SQLCODE=-1422**
- **VALUE_ERROR :** Lorsque PL/SQL rencontre, en dehors d'un ordre LMD, une erreur de conversion, de troncature ou de bornes sur des données numériques ou alphanumériques. **ORA-06502 SQLCODE = -6502**
- **ZERO_DIVIDE :** Un programme a tenté une division par zéro. **ORA-01476 SQLCODE = -1476**

Les exceptions anonymes

- Pour les exceptions anonymes (n'ayant pas des noms prédéfinis), on utilisera la clause **WHEN OTHERS** pour les traiter.
- Il est toutefois souhaitable de pouvoir déterminer au sein du gestionnaire d'exceptions la nature de l'erreur survenue.
- Oracle fournit les fonctions **SQLCODE** et **SQLERRM**, qui renvoient respectivement le code et le message d'erreur correspondant à l'exception.

Les exceptions anonymes

- **SQL>** begin
  2 DELETE FROM employés Where NumEmployé = 1;
  3 exception
  4 When OTHERS Then
  5 dbms_output.put_line('SQLCODE =' || SQLCODE);
  6 dbms_output.put_line('SQLERRM =' || SQLERRM);
  7 end;
  8 /
- **SQLCODE =-2292**
  SQLERRM =ORA-02292: violation de contrainte (ZAAFRANI.COMMANDES_NUMEMPLOYÉ_FK) d'intégrité - enregistrement fils existant

Les exceptions anonymes

- On préférera, dans de nombreux cas, traiter ces erreurs de manière spécifique afin de mieux les documenter.
- Pour ce faire, on affecte **un nom** particulier à l'erreur que le programme est susceptible de rencontrer, puis on écrit un gestionnaire d'exceptions dédié à cette exception nommée.
- Pour associer un nom à un code d'erreur interne, on se servira d'**une pragma**, une instruction spéciale du compilateur, qui est traitée lors de la compilation plutôt que durant l'exécution.

Les exceptions anonymes

- L'instruction **PRAGMA EXCEPTION_INIT** demande au compilateur d'associer une exception utilisateur à un code d'erreur Oracle spécifique.
- Une fois l'erreur associée à un nom, il est possible de la déclencher à volonté et d'écrire un gestionnaire d'exceptions qui la traitera.
- L'instruction **PRAGMA EXCEPTION_INIT** doit apparaître dans la section de déclaration d'un bloc, après la déclaration du nom d'exception qui est utilisé dans l'instruction.

Les exceptions anonymes

- La syntaxe de déclaration d'une exception anonyme :
- **DECLARE**
  NOM_EXCEPTION EXCEPTION;
  PRAGMA EXCEPTION_INIT(NOM_EXCEPTION, CODE_ERREUR);
  BEGIN
  …
  EXCEPTION
  WHEN NOM_EXCEPTION THEN
  INSTRUCTIONS;
  END;
- **CODE_ERREUR** : C'est le code d'erreur Oracle, y compris le signe moins si le code d'erreur est négatif, ce qui est en général le cas.

Les exceptions anonymes

- **SQL>** Declare
  2 DELETE_CASCADE_ENFANT EXCEPTION;
  3 PRAGMA EXCEPTION_INIT(DELETE_CASCADE_ENFANT, -2292);
  4 v_NumEmployé employés.NumEmployé%TYPE;
  5 begin
  6 v_NumEmployé := &Numéro_Employé;
  7 DELETE FROM employés Where NumEmployé = v_NumEmployé;
  8 exception
  9 When DELETE_CASCADE_ENFANT Then
  10 dbms_output.put_line('exception : DELETE_CASCADE_ENFANT');
  11 DELETE FROM commandes Where NumEmployé = v_NumEmployé;
  12 UPDATE employés SET RendCompte = NULL Where RendCompte = v_NumEmployé;
  13 DELETE FROM employés Where NumEmployé = v_NumEmployé;
  14 ROLLBACK;
  15 end;
  16 /
- **Entrez une valeur pour numéro_employé : 1**
  exception : DELETE_CASCADE_ENFANT

Les exceptions anonymes

- L'exemple précédent montre la déclaration d'une exception associée à l'erreur **ORA-2292**.
- Cette erreur survient lorsque l'on tente d'effacer un enregistrement qui est référencé comme clé étrangère.
- Dans le gestionnaire d'exceptions, tous les enregistrements correspondants dans la table commandes sont effacés et les valeurs du champ rend compte des enregistrements correspondants sont mises à NULL.

Les exceptions utilisateur

- Les exceptions **prédéfinies** par **PL/SQL** se rapportent aux **erreurs internes ou système**.
- Les problèmes rencontrés par un utilisateur dans une application sont pour la plupart spécifiques à cette application. Un programme peut nécessiter la gestion d'erreurs telle que « solde négatif dans un compte ».
- Ces erreurs constituent des exceptions aux traitements normaux , et vos programmes doivent les gérer.

Les exceptions utilisateur

- Ces exceptions sont gérées de la même manière que les exceptions internes, comme dans la syntaxe suivante :
- **DECLARE**
  NOM_EXCEPTION EXCEPTION;
  BEGIN
  …
  … RAISE NOM_EXCEPTION;
  …
- **EXCEPTION**
  WHEN NOM_EXCEPTION THEN
  INSTRUCTIONS;
  END;
- **RAISE :** permet de lancer une exception utilisateur.

Exemple d'exception utilisateur

- Ci-dessous un exemple d'exception utilisateur :
- Le bloc **PL/SQL** commence par la définition d'une exception **UPDATE_EMPLOYES**.
- Si le salaire saisi est inférieur au salaire actuel, la modification de l'employé n'est pas effectuée et l'exception est lancée.

Les exceptions utilisateur

```
SQL>Declare
  2    UPDATE_EMPLOYES  EXCEPTION;
  3    v_NumEmployé employés.NumEmployé%TYPE;
  4    v_salaire employés.salaire%TYPE;
  5  begin
  6    v_NumEmployé := &Numéro_Employé;
  7    v_salaire := &salaire;
  8    for emp in ( SELECT salaire From employés
  9              WHERE  NumEmployé = v_NumEmployé)
 loop
 10        if v_salaire < emp.salaire then
 11             dbms_output.put_line('Le salaire actuel est
'|| emp.salaire)
 12             RAISE UPDATE_EMPLOYES;
 13       end if;
 14    end loop;
```

Les exceptions utilisateur

```
 15    UPDATE  employés SET salaire = v_salaire
 16    WHERE  NumEmployé = v_NumEmployé;
 17  exception
 18    When UPDATE_EMPLOYES  Then
 19       dbms_output.put_line('Exception utilisateur :
UPDATE_EMPLOYES');
 20  end;
 21  /
```

Entrez une valeur pour numéro_employé : 10
Entrez une valeur pour salaire : 1200
Le salaire actuel est 1210
Exception utilisateur : UPDATE_EMPLOYES