Stockage de traitements en bases de données

PL/SQL

Plan du cours

- **Traitements des exceptions**
- **Transaction autonome**
- **Retour sur les triggers**

Les Exceptions

Un bloc PL/SQL est délimité par les mots-clés *BEGIN* et *END*.

- On peut découper un bloc en deux sous blocs :
  - le premier qui est la partie exécutée s'il n'y a pas d'erreur
  - le deuxième pour le traitement des erreurs.
- Ces deux blocs sont séparés par le mot-clé EXCEPTION.

Les Exceptions

La syntaxe d'un bloc est

```
BEGIN
....
EXCEPTION
    WHEN nomErreur1 THEN
    ....
    WHEN nomErreur2 THEN
    ...
    WHEN OTHERS THEN
    ...
END;
```

Les Exceptions

Des eexceptions prétraitées par Oracle

- DUP_VAL_ON_INDEX
- NO_DATA_FOUND
- TOO_MANY_ROWS
- ZERO_DIVIDE
- CURSOR_ALREADY_OPEN

DUP_VAL_ON_INDEX

- Cette exception est levée quand il y a une erreur d'unicité
- Un grand classique

```
BEGIN
    INSERT INTO …..
EXCEPTION
WHEN DUP_VAL_ON_INDEX THEN
    UPDATE …
END;
```

Autres Exceptions

- TOO_MANY_ROWS est levée quand un ordre SELECT en mode direct renvoie plus d'une ligne
- NO_DATA_FOUND est levée quand l'exécution d'un SELECT renvoie 0 lignes.
- CURSOR_ALREADY_OPEN est levée quand on essaie d'ouvrir un curseur déjà ouvert.

Exceptions Utilisateurs

- Le programmeur peut créer ses propres exceptions.

- Pour ce faire
  1. la déclarer : EXCEPTION nomErreur;
  2. la provoquer : RAISE nomErreur;
  3. la traiter : WHEN nomErreur THEN …

Exemple Exception Utilisateur

```
BEGIN
….
IF note > 20 OR note < 0 THEN raise noteErronee;
END IF;
…
EXCEPTION
        WHEN noteErronee THEN
                dbms_output.put_line('Note incorrecte');
END;
```

Imbrication d'Exception

```
ok : BOOLEAN;
BEGIN
        ok := false;
        WHILE ok LOOP
            BEGIN
            -- saisie de la note
            ok := true;
            IF note > 20 OR note < 0 THEN raise noteErronee;
            END IF;
            …
            EXCEPTION
                    WHEN noteErronee THEN
                            ok := false;
            END;
        END LOOP;
        …
EXCEPTION
        …
END;
```

Les autres exceptions d'Oracle

- Toutes les erreurs Oracle ne sont pas définies.
- Pour en définir une, on doit :
  1. La déclarer : EXCEPTION nomException
  2. L'associer : PRAGMA EXCEPTION_INIT(nomErreur, codeErreur);
     (pour le code de l'erreur voir une documentation Oracle)
- C'est le SGBD Oracle qui lève cette exception

La procédure raise_application_error

- C’est une procédure prédéfinie qui permet de sortir de toute procédure, fonction ou trigger
- Elle reçoit deux paramètres
  1. un code d’erreur inférieur à -20000
  2. un message d’erreur
- Si elle se produit dans un trigger, elle provoque l’annulation de l’ordre de MAJ SQL qui a provoqué le trigger.
- Elle affiche le code et le message dans la console

Transaction indépendante

- Une procédure ou une fonction peut s'exécuter dans une transaction indépendante de la transaction courante.

- Pour déclarer une telle procédure
  CREATE OR REPLACE [PROCEDURE|FUNCTION] child_block IS
  PRAGMA AUTONOMOUS_TRANSACTION
  ….

Transaction indépendante

- Dans un trigger, la table en mutation n'est pas accessible. On utilise une fonction ou une procédure avec transaction indépendante pour y accéder.
- Pour mémoriser des erreurs sur des tentatives de maj de la BDD, on doit utiliser une transaction indépendante pour insérer les erreurs dans une table prévue à cet effet.
  Cela permet d'annuler la tentative de MAJ tout en mettant à jour la table d'erreur.