Morgado-Samagaio Jonathan

PABD / TP10 Oracle

Table des matières

1. Introduction

Dans ce TP, nous allons voir comment utiliser les exceptions dans des blocs PL/SQL avec l'utilisation de PRAGMA

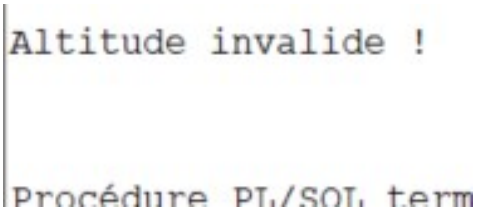
2. Bloc PL /SQL d'ajout d'une montagne

Nous reprenons le bloc d'ajout d'une montagne en lui ajoutant PRAGMA. Voici le code :

```
SET SERVEROUTPUT ON;
ACCEPT pNom PROMPT 'Saisir un nom de montagne : ';
ACCEPT pAlt PROMPT 'Saisir l''altitude de la montag
ACCEPT pCha PROMPT 'Saisir la chaîne de montagnes :

DECLARE
    vNom Montagne.Nom%TYPE := '&pNom';
    vAlt Montagne.Altitude%TYPE := &pAlt;
    vCha Montagne.Chaine%TYPE := '&pCha';
    n NUMBER;
    erreur_montagne EXCEPTION;
    erreur_altitude EXCEPTION;
    PRAGMA EXCEPTION_INIT(erreur_altitude, -2290);
BEGIN
    --Vérification de l'existence de la montagne
    SELECT COUNT(*) INTO n FROM Montagne WHERE nom
    IF(n > 0) THEN
        RAISE erreur_montagne;
    END IF;
    --Insérer la nouvelle montagne
    INSERT INTO Montagne (nm, nom, altitude, chaine
    VALUES(seq_montagne.NEXTVAL, vNom, vAlt, vCha);
    DBMS_OUTPUT.PUT_LINE('Montagne '||vNom||' ajout
    --Valider (fin de transaction)
    COMMIT;
EXCEPTION
    WHEN erreur_montagne THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Montagne existante !
    WHEN erreur_altitude THEN
        ROLLBACK;
```

Nous pouvons voir l'association d'erreur_altitude et -2290. En testant avec une altitude négative, on obtient :



3) On va maintenant modifier le bloc pour utiliser une erreur système a la place de erreur. Voici le code

```
SET SERVEROUTPUT ON;

ACCEPT pNom PROMPT 'Saisir un nom de montagne : ';
ACCEPT pAlt PROMPT 'Saisir l''altitude de la montagn
ACCEPT pCha PROMPT 'Saisir la chaîne de montagnes :

DECLARE
   vNom Montagne.Nom%TYPE      := '&pNom';
   vAlt Montagne.Altitude%TYPE := &pAlt;
   vCha Montagne.Chaine%TYPE   := '&pCha';
   n NUMBER;
   altError EXCEPTION;
   PRAGMA EXCEPTION_INIT(altError, -2290);
BEGIN
    -- Insérer la nouvelle montagne
    INSERT INTO Montagne (nm, nom, altitude, chaine)
    VALUES (seq_montagne.NEXTVAL, vNom, vAlt, vCha);
    DBMS_OUTPUT.PUT_LINE('Montagne '||vNom||' ajouté
    -- Valider (fin de transaction)
    COMMIT;
EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Montagne existante !'
    WHEN altError THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Altitude invalide !')
```

3. Bloc PL /SQL d'ajout d'une montagne à un pays

Nous allons modifier le bloc d'ajout de montagne à un pays en ajoutant PRAGMA. Voici le code :

```
SET SERVEROUTPUT ON;

ACCEPT pNom  PROMPT 'Saisir un nom de montagne : ';
ACCEPT pAlt  PROMPT 'Saisir l''altitude de la montagne : ';
ACCEPT pCha  PROMPT 'Saisir la chaîne de montagnes : ';
ACCEPT pNomp PROMPT 'Saisir le nom du pays : ';

DECLARE
   vNom  Montagne.Nom%TYPE      := '&pNom';
   vAlt  Montagne.Altitude%TYPE :=  &pAlt;
   vCha  Montagne.Chaine%TYPE   := '&pCha';
   vNomp Pays.Nom%TYPE          := '&pNomp';
   n NUMBER;
   altError EXCEPTION;
   PRAGMA EXCEPTION_INIT(altError, -2290);
   nomPError EXCEPTION;
   PRAGMA EXCEPTION_INIT(nomPError, -2291);
BEGIN
   -- Insérer la nouvelle montagne
   INSERT INTO Montagne (nm, nom, altitude, chaine)
   VALUES (seq_montagne.NEXTVAL, vNom, vAlt, vCha);
   DBMS_OUTPUT.PUT_LINE('Montagne '||vNom||' enregistrée.');
   -- Insérer la nouvelle localisation
   INSERT INTO Localiser (nm, nomp)
   VALUES (seq_montagne.CURRVAL, vNomp);
   DBMS_OUTPUT.PUT_LINE('Localisation de '||vNom||' dans '||vNomP||
   -- Valider (fin de transaction)
   COMMIT;
EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Montagne existante !');
    WHEN altError THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Altitude invalide !');
```

4. Bloc PL/SQL d'ajout d'un pays

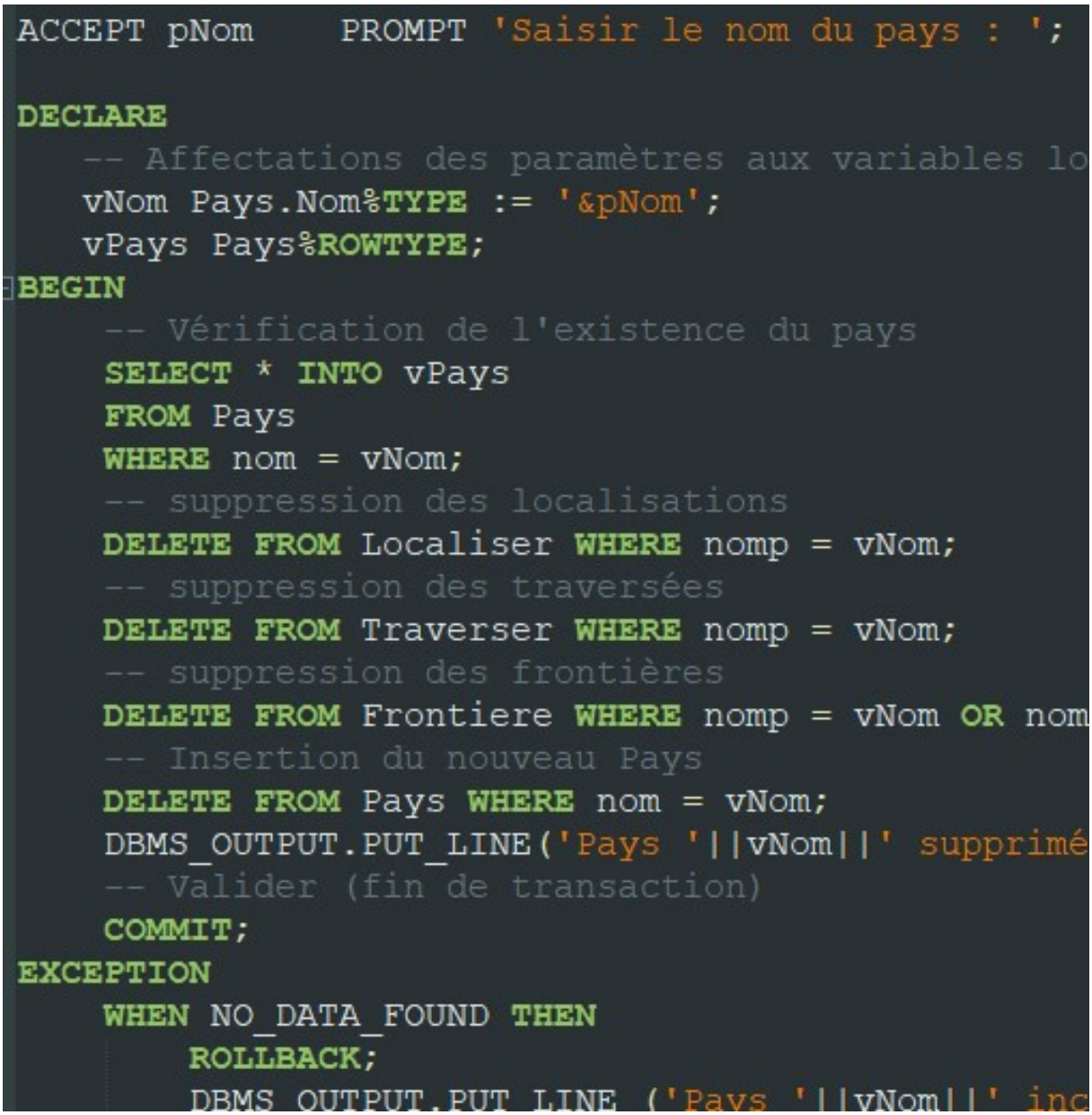
Nous reprenons le bloc d'ajout d'un pays en ajoutant PRAGMA. Voici le code :

```
ACCEPT pNom    PROMPT 'Saisir le nom du pays : ';
ACCEPT pCap    PROMPT 'Saisir la capitale du pays : ';
ACCEPT pSup    PROMPT 'Saisir la superficie du pays : ';
ACCEPT pPop    PROMPT 'Saisir la population du pays : ';
ACCEPT pIs3    PROMPT 'Saisir le code ISO3 du pays : ';
ACCEPT pIs2    PROMPT 'Saisir le code ISO2 du pays : ';
ACCEPT pCon    PROMPT 'Saisir le contnent du pays : ';

DECLARE
   -- Affectations des paramètres aux variables locales
   vNom Pays.Nom%TYPE         := '&pNom';
   vCap Pays.Capitale%TYPE    := '&pCap';
   vSup Pays.Superficie%TYPE  := &pSup;
   vPop Pays.Population%TYPE  := &pPop;
   vIs3 Pays.Iso3%TYPE        := '&pIs3';
   vIs2 Pays.Iso2%TYPE        := '&pIs2';
   vCon Pays.NomC%TYPE        := '&pCon';
   n NUMBER;
   ckError EXCEPTION;
   PRAGMA EXCEPTION_INIT(ckError, -2290);
   fkError EXCEPTION;
   PRAGMA EXCEPTION_INIT(fkError, -2291);
BEGIN
    -- Insertion du nouveau Pays
    INSERT INTO Pays (nom, capitale, superficie, population, iso3,
    VALUES (vNom, vCap, vSup, vPop, vIs3, vIs2, vCon);
    DBMS_OUTPUT.PUT_LINE('Pays '||vNom||' ajouté.');
    -- Valider (fin de transaction)
    COMMIT;
EXCEPTION
    WHEN fkError THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Continent'||vCon||' inconnu !');
    WHEN ckError THEN
        ROLLBACK;
        IF (SQLERRM LIKE '%CK_PAYS_SUPERFICIE%') THEN
            DBMS_OUTPUT.PUT_LINE ('Superficie de '||vNom||' négati
        ELSIF (SQLERRM LIKE '%CK_PAYS_POPULATION%') THEN
            DBMS_OUTPUT.PUT_LINE ('Population de '||vNom||' négati
        END IF;
    WHEN DUP_VAL_ON_INDEX THEN
        ROLLBACK;
        IF (SQLERRM LIKE '%PK_PAYS%') THEN
            DBMS_OUTPUT.PUT_LINE ('Pays '||vNom||' existant !');
        ELSIF (SQLERRM LIKE '%UK_PAYS_ISO3%') THEN
```

5. Bloc PL/SQL de suppression d'un pays

Nous reprenons le de suppression d'un pays. Voici le code :



```
ACCEPT pNom    PROMPT 'Saisir le nom du pays : ';

DECLARE
   -- Affectations des paramètres aux variables lo
   vNom Pays.Nom%TYPE := '&pNom';
   vPays Pays%ROWTYPE;
BEGIN
    -- Vérification de l'existence du pays
    SELECT * INTO vPays
    FROM Pays
    WHERE nom = vNom;
    -- suppression des localisations
    DELETE FROM Localiser WHERE nomp = vNom;
    -- suppression des traversées
    DELETE FROM Traverser WHERE nomp = vNom;
    -- suppression des frontières
    DELETE FROM Frontiere WHERE nomp = vNom OR nom
    -- Insertion du nouveau Pays
    DELETE FROM Pays WHERE nom = vNom;
    DBMS_OUTPUT.PUT_LINE('Pays '||vNom||' supprimé
    -- Valider (fin de transaction)
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE ('Pays '||vNom||' inc
```

6. Conclusion

Dans ce TP, nous avons pu voir comment utiliser les exceptions systèmes avec PRAGMA qui associe à un numéro d'erreur une variable que l'on créer.