B2.3

PL/SQL - CURSOR

BTS SIO - NGO

Définition

- *On définit en PL/SQL un **curseur** comme un espace mémoire qui contient le résultat d'une requête SQL.*
- *PL/SQL utilise des curseurs pour tous les accès à des informations de la base de données. On peut donc dire que toutes les requêtes sont associées à un curseur.*

RAPPEL BLOC PL/SQL

- //etudiant (etu_id, etu_nom)
- //Affichage du nom de l'étudiant 5, afficher une exception si le bloc PL/SQL ne retourne plus d'une ligne

```
DECLARE
Etunom char(50);
BEGIN
select etu_nom into Etunom from etudiant where etu_id=5
DBMS_OUTPUT.PUT_LINE('nom etudiant:'|| Etunom) ;
EXCEPTION
WHEN TOO_MANY_ROW THEN
DBMS_OUTPUT.PUT_LINE('trop de ligne') ;
END ;
/
```

Problématique du BLOC PL/SQL

- une bloc PL/SQL ne doit retourner qu'un seul enregistrement.

- Comment faire pour traiter des requêtes SELECT qui renvoient plusieurs enregistrements ?

Les curseurs (cursors)

1. CURSEUR

- Pour traiter une requête SELECT qui retourne plusieurs lignes, l'utilisateur doit définir un curseur qui lui permet d'extraire la totalité des lignes sélectionnées.
- On parle des curseurs explicites (ou programmables)
- Le traitement du SELECT se fera ligne par ligne

- L'utilisation d'un curseur explicite nécessite 4 étapes :
  - 1. Déclaration du curseur, 2. Ouverture du curseur,
  - 3. Traitement des lignes, 4. Fermeture du curseur

1.1. DÉCLARATION CURSOR

- La déclaration du curseur permet de stocker l'ordre (la requête) SELECT dans le curseur.
- Un curseur est déclaré dans la partie DECLARE d'un bloc PL/SQL.

- La syntaxe est la suivante :
- CURSOR nomcurseur IS la_requête_SELECT ;
- Exemple:
- DECLARE
- CURSOR Nom IS SELECT emp_nom FROM employe

1.2. OUVERTURE / FERMETURE

- L’ouverture et la fermeture du curseur est effectuée dans la partie exécution du bloc PL/SQL.

- BEGIN …
- OPEN nomcurseur;
- …
- CLOSE nomcurseur;
- END;

- NB1 : Il faut bien faire attention à n’ouvrir que les curseurs qui sont déjà déclarés, Il faut faire attention à ne fermer que les curseurs ouverts.

1.3. TRAITEMENT

- Un des avantages des curseurs est de pouvoir traiter (un par un) plusieurs enregistrements renvoyés par une requêtes SQL.
- L'extraction de lignes est réalisée avec l'instruction FETCH. … INTO.

- La syntaxe est la suivante :
- FETCH nomcurseur INTO nomvariable1, nomvariable2…

- Contrairement à un bloc PL/SQL, un curseur peut contenir plus d'une variable

1.4. DECLARATION CURSOR

- DECLARE
- CURSOR nomcurseur IS requête sql;
- Variable 1
- Variable 2 …
- BEGIN
- OPEN nomcurseur ;
- LOOP FETCH nomcurseur INTO variable 1, variable 2…;
- *Instruction*
- *END LOOP*
- CLOSE nomcurseur ;
- END;
- /

1.4. EXEMPLE

```
DECLARE
CURSOR emp IS SELECT emp_id, emp_nom FROM employe;
NO employe.emp_id%TYPE;
NOM employe.emp_nom %TYPE;
BEGIN
OPEN emp;
LOOP FETCH emp INTO NO, NOM;
dbms_output.put_line('Numéro : ' || NO || 'Nom : ' || NOM);
EXIT WHEN emp%NOTFOUND;
END LOOP;
CLOSE emp;
END;
/
```

Attributs des curseurs

▪*Chaque curseur possède des attributs*

▪*Ces attributs permettent de connaître un certain nombre d'informations qui ont été renvoyées après l'instruction du LMD SQL et qui peuvent être utiles au programmeur.*

2.1. Liste des ATTRIBUTS

Variable	Attribut	Description
%FOUND	Cur%FOUND	Booléen valant TRUE si la dernière instruction LMD affecte au moins un enregistrement.
%NOTFOUND	Cur%NOTFOUND	Booléen valant TRUE si la dernière instruction LMD n'affecte aucun enregistrement.
%ROWCOUNT	Cur%ROWCOUNT	Nombre de lignes affectées par la dernière instruction LMD.
%ISOPEN	Cur% ISOPEN	Cet attribut est de type booléen : la valeur de l'attribut est TRUE si le curseur est ouvert et FALSE si le curseur est fermé

21

2.2. Syntaxe attributs

- %ISOPEN
- IF NOT moncurseur%ISOPEN THEN OPEN moncurseur; END IF;

- %FOUND
- OPEN moncurseur; LOOP FETCH moncurseur INTO variable1; EXIT WHEN NOT moncurseur%FOUND; END LOOP; CLOSE moncurseur;

- %NOTFOUND
- OPEN moncurseur; LOOP FETCH moncurseur INTO variable1; EXIT WHEN moncurseur%NOTFOUND; END LOOP; CLOSE moncurseur;

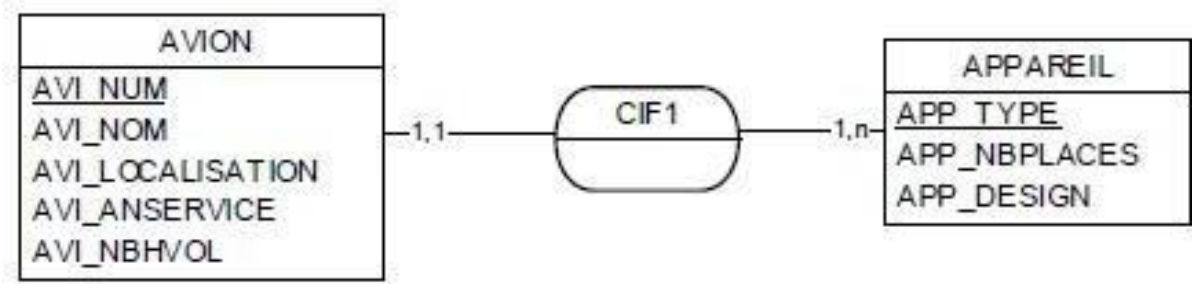
- %ROWCOUNT
- OPEN moncurseur; LOOP FETCH moncurseur INTO variable1; EXIT WHEN NOT moncurseur%FOUND; END LOOP; DBMS_OUTPUT.PUT_LINE(moncurseur%ROWCOUNT); CLOSE moncurseur;

EXERCICES

Avec le compte aero/mpaero
DOCUMENT + COPIE D'ECRAN + dépot NAS

A) CURSOR

- 1) Afficher le nom de tous les appareils en vérifiant auparavant si le curseur est ouvert
- 2) Afficher le nom et le numéro tous les avions de type boeing si le curseur trouve au moins un enregistrement
- 3) Afficher le nom des avions qui ont plus de 200 places
- 4) Afficher le nombre moyen de place des appareils, sortir du curseur si on ne retourne plus de résultat



BOUCLE FOR

▪*L’utilisation de la boucle FOR dans les curseurs permet de fournir au programmeur une structure simple et efficace. Dans ce cas on parle de curseur semi-automatique.*

▪*Si on utilise la boucle FOR dans les curseurs: On va éviter les directives OPEN, FETCH et CLOSE*

La boucle FOR s'arrête automatiquement après l’extraction de la dernière ligne du curseur,
la variable de réception du curseur est automatiquement déclarée (%ROWTYPE du curseur)

3.1. SYNTAXE FOR

```
DECLARE
CURSOR emp IS SELECT * FROM e_emp;
BEGIN
FOR var_employe IN emp LOOP
dbms_output.put_line('NO: ' || var_employe.emp_id || 'NOM: ' ||
var_employe.emp_nom);
END LOOP;
END;
/
```

EXERCICES

DOCUMENT + COPIE D’ECRAN + dépot NAS

B) FOR

1. Afficher tous les avions localisés en France
2. Afficher les avions localisé en Angleterre et les avions localisé en Allemagne
3. Afficher le nom, la localisation des avions de type Airbus A320, on sort si on la requête retourne plus de ligne
4. Ecrire un curseur qui permet d'afficher le nom des avions qui ont un nombre d'heure de vol supérieur à la moyenne des heure de vol des avions
5. Supprimer les avions ayant plus de 30000h de vol

FOR UPDATE

▪*Si on cherche à verrouiller les lignes d'une table interrogée par un curseur pour les mettre à jour, sans que d'autres utilisateurs ne les modifie en même temps, il faut utiliser la clause FOR UPDATE*

▪*Elle est utilisée lors de la déclaration du curseur et verrouille les lignes concernées lorsque le curseur est ouvert, Les verrous sont libérés à la fin de la transaction*

4. FOR UPDATE exemple 1

- DECLARE
- CURSOR moncurseur IS *SELECT champ1 FROM table* FOR UPDATE OF Champs à modifier NOWAIT;
- BEGIN FOR variable IN moncurseur LOOP
- *UPDATE table SET champ = champ + 1 WHERE*
- CURRENT OF moncurseur ;
- END LOOP;
- COMMIT;
- END;

- Avec NOWAIT les lignes sont verrouillés jusqu'à ce que le curseur est fermé ou un commit/rollback est exécuté.

4. FOR UPDATE exemple 2

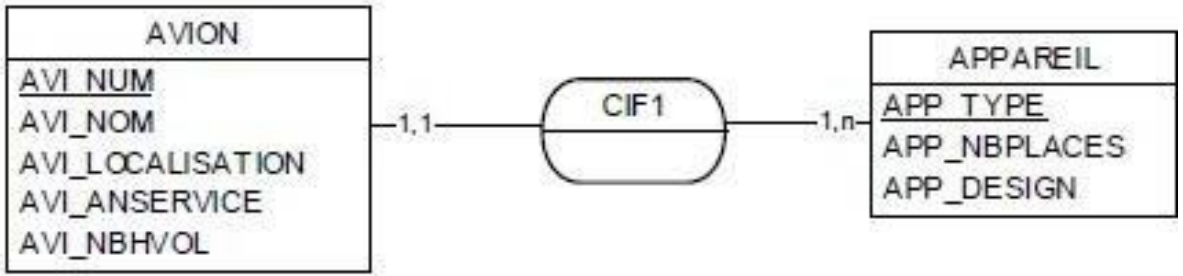
- DECLARE
- CURSOR c_emp IS SELECT * FROM employe FOR UPDATE OF emp_salaire NOWAIT;
- BEGIN FOR emp IN c_emp LOOP
- IF emp.SERVICE_NO=1 THEN update employe set emp_salaire =1000 WHERE CURRENT OF c_emp;
- ELSIF emp.SERVICE_NO=2 THEN update employe set emp_salaire =2000 WHERE CURRENT OF c_emp;
- ELSE update employe set emp_salaire =3000 WHERE CURRENT OF c_emp;
- END IF; END LOOP; COMMIT;
- END;
- /

EXERCICES

DOCUMENT + COPIE D’ECRAN + dépot NAS

C)

- 6) Modifier le design de tous les appareils airbus 300 en bleu
- 7) Modifier le nom de l'avion de type boeing 777 en votre prénom
- 8) Ajouter les heures de vol de tous les boeing 787 de 10h




Design

Merci

Des questions ?

Les cours sont disponibles sur le NAS