Interagir avec le serveur Oracle

Objectifs

A la fin de ce chapitre, vous pourrez :

- écrire une instruction SELECT correcte en PL/SQL
- écrire des instructions LMD en PL/SQL
- contrôler des transactions en PL/SQL
- déterminer le résultat de l'exécution d'instructions SQL LMD (Langage de manipulation de données)

Instructions SQL en PL/SQL

- Extraire une ligne de données à partir d'une base de données en utilisant la commande `SELECT`
- Modifier des lignes de la base de données en utilisant des instructions LMD
- Contrôler une transaction avec la commande `COMMIT`, `ROLLBACK` ou `SAVEPOINT`
- Déterminer le résultat de l'exécution d'une instruction LMD avec des attributs de curseur implicite

Instructions SELECT en PL/SQL

Extraire les données de la base de données à l'aide d'une instruction SELECT

Syntaxe :

```
SELECT   select_list
INTO     {variable_name[, variable_name]...
         | record_name}
FROM     table
[WHERE   condition];
```

Instructions SELECT en PL/SQL

- La clause INTO est obligatoire
- Les interrogations doivent renvoyer une et une seule ligne

Exemple :

```
DECLARE
  v_deptno           NUMBER(4);
  v_location_id      NUMBER(4);
BEGIN
  SELECT      department_id, location_id
  INTO        v_deptno, v_location_id
  FROM        departments
  WHERE       department_name = 'Sales';
  ...
END;
/
```

Extraire des données en PL/SQL

Extraire la date d'embauche et le salaire de l'employé indiqué

Exemple :

```
DECLARE
  v_hire_date   employees.hire_date%TYPE;
  v_salary      employees.salary%TYPE;
BEGIN
  SELECT   hire_date, salary
  INTO     v_hire_date, v_salary
  FROM     employees
  WHERE    employee_id = 100;
  ...
END;
/
```

Extraire des données en PL/SQL

Renvoyer la somme des salaires de tous les employés du service indiqué

Exemple :

```
SET SERVEROUTPUT ON
DECLARE
  v_sum_sal   NUMBER(10,2);
  v_deptno     NUMBER NOT NULL := 60;
BEGIN
  SELECT      SUM(salary)  -- group function
  INTO        v_sum_sal
  FROM        employees
  WHERE       department_id = v_deptno;
  DBMS_OUTPUT.PUT_LINE ('The sum salary is ' ||
                        TO_CHAR(v_sum_sal));
END;
/
```

Conventions d'appellation

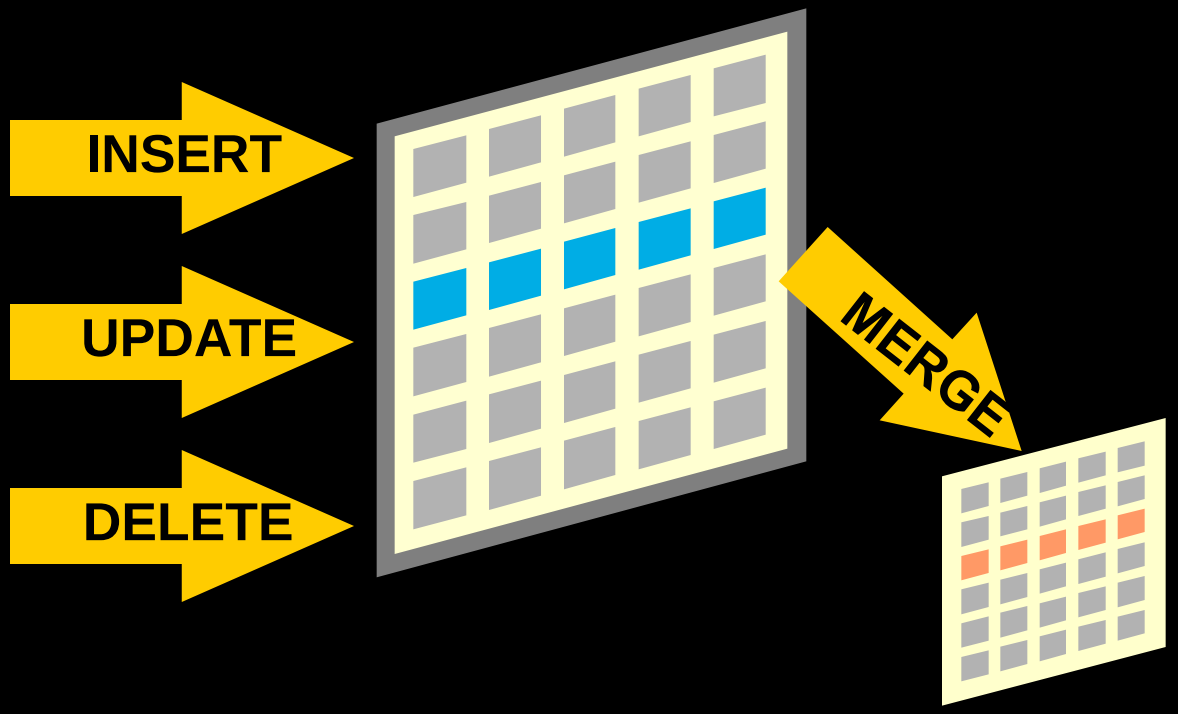
```
DECLARE
  hire_date       employees.hire_date%TYPE;
  sysdate         hire_date%TYPE;
  employee_id     employees.employee_id%TYPE := 176;
BEGIN
  SELECT      hire_date, sysdate
  INTO        hire_date, sysdate
  FROM        employees
  WHERE       employee_id = employee_id;
END;
/
```

```
DECLARE
*
ERROR at line 1:
ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 6
```

Manipuler les données en PL/SQL

Modifier des tables de base de données en utilisant les instructions LMD suivantes :

- `INSERT`
- `UPDATE`
- `DELETE`
- `MERGE`



Insérer des données

Ajouter les informations relatives à un nouvel employé à la table `EMPLOYEES`

Exemple :

```
BEGIN
  INSERT INTO employees
  (employee_id, first_name, last_name, email,
   hire_date, job_id, salary)
  VALUES
   (employees_seq.NEXTVAL, 'Ruth', 'Cores', 'RCORES',
    sysdate, 'AD_ASST', 4000);
END;
/
```

Mettre à jour des données

Augmenter le salaire de tous les employés chargés du contrôle des stocks

Exemple :

```
DECLARE
  v_sal_increase   employees.salary%TYPE := 800;
BEGIN
  UPDATE      employees
  SET         salary = salary + v_sal_increase
  WHERE       job_id = 'ST_CLERK';
END;
/
```

Supprimer des données

Supprimer les lignes appartenant au service 10 à partir de la table `EMPLOYEES`

Exemple :

```
DECLARE
  v_deptno   employees.department_id%TYPE := 10;
BEGIN
  DELETE FROM   employees
  WHERE         department_id = v_deptno;
END;
/
```

Fusionner des lignes

Insérer ou mettre à jour des lignes dans la table `COPY_EMP`, pour correspondre à la table `EMPLOYEES`

```
DECLARE
      v_empno employees.employee_id%TYPE := 100;
BEGIN
MERGE INTO copy_emp c
     USING employees e
     ON (e.employee_id = v_empno)
   WHEN MATCHED THEN
     UPDATE SET
       c.first_name     = e.first_name,
       c.last_name      = e.last_name,
       c.email          = e.email,
       . . .
   WHEN NOT MATCHED THEN
     INSERT VALUES(e.employee_id, e.first_name, e.last_name,
          . . .,e.department_id);
END;
```

Conventions d'appellation

- Utiliser une convention d'appellation pour éviter toute ambiguïté dans la clause `WHERE`
- Les colonnes de base de données et les identificateurs doivent porter des noms différents
- Des erreurs de syntaxe peuvent survenir car PL/SQL recherche en premier lieu une colonne de table dans la base de données
- Les noms des variables locales et les paramètres formels ont priorité sur les noms des tables de la base de données
- Les noms de colonne des tables de la base de données ont priorité sur les noms des variables locales

Curseur SQL

- Un curseur est une zone de travail réservée à SQL
- Il existe deux types de curseur :
  - curseurs implicites
  - curseurs explicites
- Le serveur Oracle utilise des curseurs implicites pour analyser et exécuter les instructions SQL
- Les curseurs explicites sont déclarés de manière explicite par le programmeur

Attributs d'un curseur SQL

Grâce aux attributs d'un curseur SQL, vous pouvez tester le résultat lié à l'exécution d'instructions SQL

`SQL%ROWCOUNT`	Nombre de lignes affectées par la dernière instruction SQL (valeur entière)
`SQL%FOUND`	Attribut booléen qui prend la valeur `TRUE` si la dernière instruction SQL affecte une ou plusieurs lignes
`SQL%NOTFOUND`	Attribut booléen qui prend la valeur `TRUE` si la dernière instruction SQL n'affecte aucune ligne
`SQL%ISOPEN`	Prend toujours la valeur `FALSE` car PL/SQL ferme les curseurs implicites immédiatement après leur exécution

Attributs de curseur SQL

Supprimer les lignes possédant l'ID d'employé indiqué dans la table `EMPLOYEES` et afficher le nombre de lignes supprimées

Exemple :

```
VARIABLE rows_deleted VARCHAR2(30)
DECLARE
  v_employee_id employees.employee_id%TYPE := 176;
BEGIN
  DELETE FROM  employees
  WHERE        employee_id = v_employee_id;
  :rows_deleted := (SQL%ROWCOUNT ||
                        ' row deleted.');
END;
/
PRINT rows_deleted
```

Instructions de gestion des transactions

- Initialiser une transaction avec la première instruction LMD suivant `COMMIT` ou `ROLLBACK`
- Utiliser les instructions SQL `COMMIT` et `ROLLBACK` pour mettre fin explicitement à une transaction

Synthèse

Ce chapitre vous a permis d'apprendre à :

- intégrer du code SQL dans un bloc PL/SQL en utilisant `SELECT`, `INSERT`, `UPDATE`, `DELETE` et `MERGE`
- intégrer des instructions de gestion des transactions dans un bloc PL/SQL en utilisant `COMMIT`, `ROLLBACK` et `SAVEPOINT`

Synthèse

Ce chapitre vous a appris les règles suivantes :

- il existe deux types de curseur : implicites et explicites
- les attributs d'un curseur implicite permettent de vérifier le résultat de l'exécution d'instructions LMD :
  - `SQL%ROWCOUNT`
  - `SQL%FOUND`
  - `SQL%NOTFOUND`
  - `SQL%ISOPEN`
- les curseurs explicites sont définis par le programmeur

Présentation de l'exercice 3

Dans cet exercice, vous allez créer un bloc PL/SQL pour :

- sélectionner des données dans une table
- insérer des données dans une table
- mettre à jour des données dans une table
- supprimer un enregistrement d'une table