Utiliser des types de données composites

Objectifs

A la fin de ce chapitre, vous pourrez :

- créer des enregistrements PL/SQL définis par l'utilisateur
- créer des enregistrements avec l'attribut `%ROWTYPE`
- créer une table `INDEX BY`
- créer une table d'enregistrements `INDEX BY`
- faire la distinction entre les enregistrements, les tables, et les tables d'enregistrement

Types de données composites

- Il existe deux catégories de données composites :
  - enregistrements PL/SQL
  - ensembles PL/SQL
    - table `INDEX BY`
    - table imbriquée
    - `VARRAY`
- Les types de données composites contiennent des composantes internes
- Les types de données composites sont réutilisables

Enregistrements PL/SQL

- Ils doivent être composés d'un ou plusieurs champs de type scalaire, `RECORD` ou `INDEX BY`
- Leur structure est similaire à celle des enregistrements en langage de troisième génération (3GL)
- Ils sont différents des lignes d'une table de base de données
- Ils traitent un ensemble de champs comme une unité logique
- Ils permettent d'extraire une ligne de données à partir d'une table pour la traiter

Créer un enregistrement PL/SQL

Syntaxe :

```
TYPE type_name IS RECORD
     (field_declaration[, field_declaration]…);
identifier    type_name;
```

Où *field_declaration* est :

```
field_name {field_type | variable%TYPE
           | table.column%TYPE | table%ROWTYPE}
           [[NOT NULL] {:= | DEFAULT} expr]
```

Créer un enregistrement PL/SQL

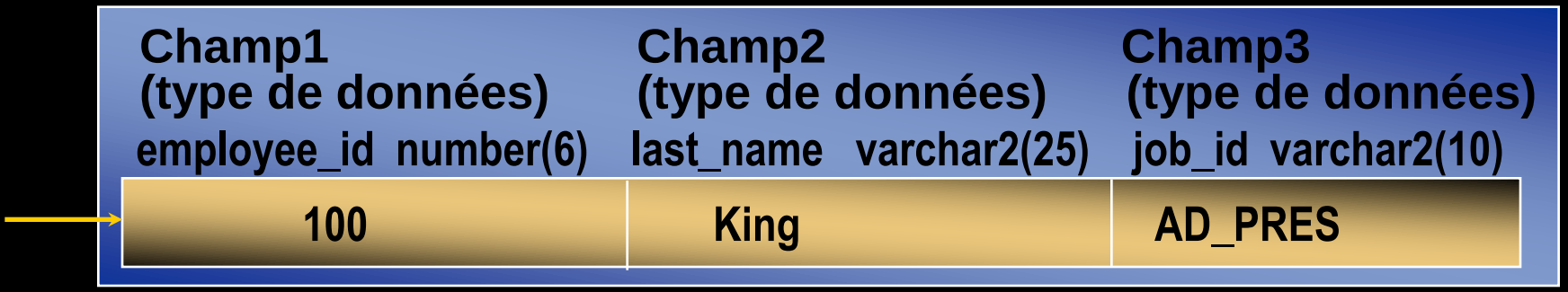
Déclarer des variables pour le stockage du nom, du poste et du salaire d'un nouvel employé.

Exemple :

```
...
  TYPE emp_record_type IS RECORD
    (last_name   VARCHAR2(25),
     job_id      VARCHAR2(10),
     salary      NUMBER(8,2));
  emp_record     emp_record_type;
...
```

Structure d'un enregistrement PL/SQL

Example:



Attribut %ROWTYPE

- Permet de déclarer une variable à partir d'un ensemble de colonnes d'une table ou d'une vue de base de données
- Doit être précédé du nom de la table de base de données
- Les champs de l'enregistrement tirent leurs noms et leurs types de données des colonnes de la table ou de la vue

Avantages liés à l'utilisation de l'attribut %ROWTYPE

- Il n'est pas nécessaire de connaître le nombre et les types de données des colonnes de la table de base de données sous-jacente
- Le nombre et les types des colonnes de la table de base de données sous-jacente peuvent être modifiés à l'exécution
- L'attribut permet d'extraire une ligne avec l'instruction `SELECT *`

Attribut %ROWTYPE

Exemples :

Déclarer une variable pour le stockage des informations relatives à un service de la table DEPARTMENTS

```
dept_record    departments%ROWTYPE;
```

Déclarer une variable pour le stockage des informations relatives à un employé de la table EMPLOYEES

```
emp_record     employees%ROWTYPE;
```

Tables INDEX BY

- Ces tables comportent 2 composants :
  - une clé primaire de type BINARY_INTEGER
  - une colonne de type scalaire ou RECORD
- Leur taille peut augmenter de façon dynamique, car elles ne présentent pas de contrainte

Créer une table INDEX BY

Syntaxe :

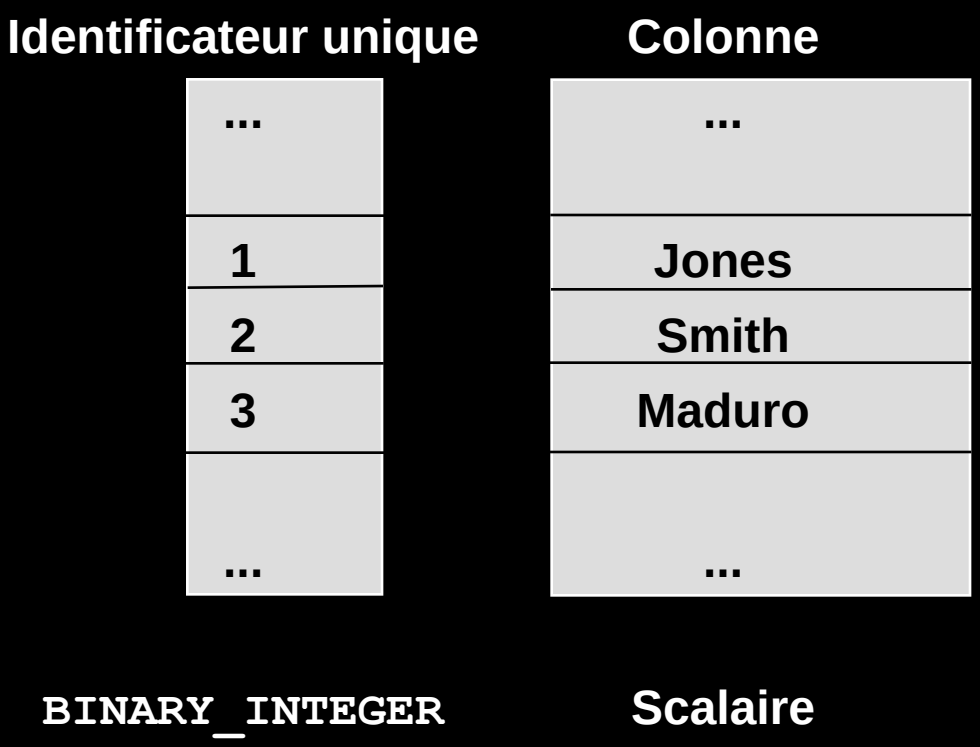
```
TYPE type_name IS TABLE OF
     {column_type | variable%TYPE
     | table.column%TYPE} [NOT NULL]
     | table.%ROWTYPE
     [INDEX BY BINARY_INTEGER];
identifier    type_name;
```

Déclarer une variable INDEX BY pour le stockage des noms

Exemple :

```
...
TYPE ename_table_type IS TABLE OF
                         employees.last_name%TYPE
  INDEX BY BINARY_INTEGER;
ename_table ename_table_type;
...
```

Structure des tables `INDEX BY`



Créer une table INDEX BY

```
DECLARE
  TYPE ename_table_type IS TABLE OF
       employees.last_name%TYPE
       INDEX BY BINARY_INTEGER;
  TYPE hiredate_table_type IS TABLE OF DATE
       INDEX BY BINARY_INTEGER;
  ename_table       ename_table_type;
  hiredate_table    hiredate_table_type;
BEGIN
  ename_table(1)    := 'CAMERON';
  hiredate_table(8) := SYSDATE + 7;
    IF ename_table.EXISTS(1) THEN
      INSERT INTO ...
    ...
END;
/
```

Utiliser les méthodes des tables INDEX BY

Les méthodes suivantes facilitent l'utilisation des tables INDEX BY :

- EXISTS
- COUNT
- FIRST and LAST
- PRIOR
- NEXT
- TRIM
- DELETE

Table d'enregistrements INDEX BY

- Définir une variable TABLE avec un type de données PL/SQL admis
- Déclarer une variable PL/SQL pour le stockage des informations d'un service.

Exemple :

```
DECLARE
  TYPE dept_table_type IS TABLE OF
       departments%ROWTYPE
       INDEX BY BINARY_INTEGER;
  dept_table dept_table_type;
  -- Each element of dept_table is a record
```

Exemple de table d'enregistrements INDEX BY

```
SET SERVEROUTPUT ON
DECLARE
   TYPE emp_table_type is table of
      employees%ROWTYPE INDEX BY BINARY_INTEGER;
   my_emp_table  emp_table_type;
   v_count       NUMBER(3):= 104;
BEGIN
  FOR i IN 100..v_count
  LOOP
        SELECT * INTO my_emp_table(i) FROM employees
        WHERE employee_id = i;
  END LOOP;
  FOR i IN my_emp_table.FIRST..my_emp_table.LAST
  LOOP
     DBMS_OUTPUT.PUT_LINE(my_emp_table(i).last_name);
  END LOOP;
END;
```

Synthèse

Ce chapitre vous a permis d'apprendre à :

- définir et référencer des variables PL/SQL possédant des types de données composites :
  - enregistrements PL/SQL
  - tables `INDEX BY`
  - tables d'enregistrements `INDEX BY`
- définir un enregistrement PL/SQL en utilisant l'attribut `%ROWTYPE`

Présentation de l'exercice 5

Dans cet exercice, vous allez :

- déclarer des tables INDEX BY
- traiter des données en utilisant les tables `INDEX BY`
- déclarer un enregistrement PL/SQL
- traiter des données à l'aide d'un enregistrement PL/SQL