Chapitre 6: les types structurés et les enregistrements

Introduction

Contrairement aux tableaux qui sont des structures de données dont tous les éléments sont de même type, les enregistrements sont des structures de données dont **les éléments peuvent être de type différent** et qui se rapportent à la **même entité** (au sens de Merise)
Les éléments qui composent un enregistrement sont appelés **champs**.

Avant de déclarer une variable enregistrement, il faut avoir au préalable définit son type, c'est à dire le nom et le type des champs qui le compose. Le type d'un enregistrement est appelé **type structuré**. (Les enregistrements sont parfois appelé structures, en analogie avec le langage C)

Préalable: déclaration d'un type structuré

Jusqu'à présent, nous n'avons utilisé que des types primitifs (caractères, entiers, réels, chaînes) et des tableaux de types primitifs. Mais nous pouvons créer nos propres types puis déclarer des variables ou des tableaux d'éléments de ce type.

Pour créer des enregistrements, il faut déclarer un nouveau type, basé sur d'autres types existants, qu'on appelle type structuré Après avoir défini un type structuré, on peut l'utiliser comme un type normal en déclarant une ou plusieurs variables de ce type. Les variables de type structuré sont appelées enregistrements.

La déclaration des types structurés se fait dans une section spéciale des algorithmes appelée Type, qui précède la section des variables (et succède à la section des constantes).

Syntaxes :

(notation inspirée du Pascal)

```
Type
   nom_type = enregistrement
                  | nom_champ1: type_champ1
                  |    …
                  | nom_champn: type_champn
              finenreg
```

(notation inspirée du C)

```
ou  Type
       Structure nom_type
          | nom_champ1: type_champ1
          | …
          | nom_champN: type_champN
       FinStruct
```

Exemple:

```
Type
   tpersonne = enregistrement
                  | nom : chaîne
                  | prénom : chaine
                  | âge : entier
              finenreg
```

```
ou  Type
       Structure tpersonne
          | nom : chaîne
          | prénom : chaîne
          | âge : entier
       FinStruct
```

I. Déclaration d'un enregistrement à partir d'un type structuré

Une fois qu'on a défini un type structuré, on peut déclarer des variables enregistrements exactement de la même façon que l'on déclare des variables d'un type primitif.

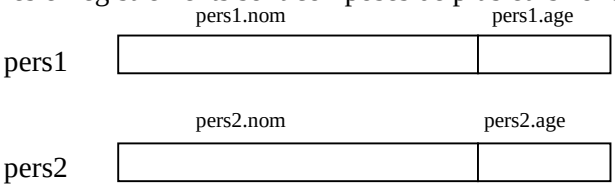
Syntaxe

Var
nom_var : *nom_type*

Exemple:

Var
pers1, pers2, pers3 : tpersonne

Représentation:
les enregistrements sont composés de plusieurs zones de données, correspondant aux champs



Exemple:
Soit l'entité suivante:

PRODUIT
code
lib
paht
pvht
txtva

légende:

code: code alphanumérique du produit
lib: libéllé
paht: prix d'achat hors taxes
pvht: prix de vente hors taxes
txtva: taux de TVA applicable

Voici comment déclarer deux occurrences (variables enregsitrements) du type structuré correspondant à cette entité.

// Il faut d'abord définir le type structuré correspondant:

Type
produit = **enregistrement**
code: chaîne
lib: chaîne
paht: réel
pvht: réel
txtva: réel
finenreg

ou

Type
Structure produit
code: chaîne
lib: chaîne
paht: réel
pvht: réel
txtva: réel
FinStruct

// Ensuite il est possible de déclarer deux variables de ce type
Var
prod1, prod2 : produit

II. Manipulation d'un enregistrement

La manipulation d'un enregistrement se fait au travers de ses champs. Comme pour les tableaux, il n'est pas possible de manipuler un enregistrement globalement, sauf pour affecter un enregistrement à un autre de même type. Par exemple, pour afficher un enregistrement il faut afficher tous ses champs uns par uns.

A. Accès aux champs d'un enregistrement

Alors que les éléments d'un tableau sont accessibles au travers de leur indice, **les champs d'un enregistrement sont accessibles à travers leur nom, grâce à l'opérateur '.'**

nom_enregistrement . nom_champ
représente la valeur mémorisée dans le champ de l'enregistrement

Par exemple, pour accéder à l'âge de la variable pers2, on utilise l'expression:
pers2.âge

💣Attention : le nom d'un champ est TOUJOURS précédé du nom de l'enregistrement auquel il appartient. On ne peut pas trouver un nom de champ tout seul, sans indication de l'enregistrement.

Les champs d'un enregistrement, tout comme les éléments d'un tableau, sont des variables à qui on peut faire subir les mêmes opérations (affectation, saisie, affichage,…).

Exemple 1:
Programme de saisie des données concernant les personnes pers1 et pers2, puis affichage de la différence d'âge entre ces deux personnes

```
Programme Exemple

Type
   Structure tpersonne
        nom : chaîne
        prénom : chaîne
        âge : entier
  FinStruct

Var
   pers1, pers2 : tpersonne

Début
    Aff "Entrez le nom puis l'age de la personne 1"
    Saisir pers1.nom, pers1.age    // il est impossible d'écrire Saisir pers1
    Aff "Entrez le nom puis l'âge de la personne 2"
    Saisir pers2.nom, pers2.age
    Aff  "La différence d'âge entre ", pers1.nom, " et ", pers2.nom, " est de "
    Si pers1.age > pers2.age
    |  Alors  Aff pers1.age – pers2.age, " ans "
    |  Sinon Aff pers2.age – pers1.age, " ans "
    FinSi
Fin
```

B. Passage d'un enregistrement en paramètre d'un sous-programme

Il est possible de **passer tout un enregistrement en paramètre** d'une fonction ou d'une procédure (on n'est pas obligé de passer tous les champs uns à uns, ce qui permet de diminuer le nombre de paramètres à passer), exactement comme pour les tableaux.

Exemple :
Voilà une fonction qui renvoie la différence d'age entre deux personnes

```
Fonction différence (p1, p2 : tpersonne)
Début
   Si pers1.age > pers2.age
     Alors  Retourne  ( pers1.age – pers2.age )
     Sinon  Retourne  ( pers2.age – pers1.age )
  FinSi
FinFonct
```

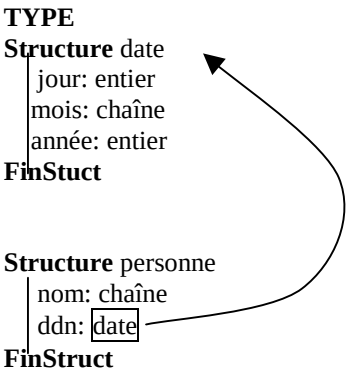
Exemple 2 :
Voilà une procédure qui permet de modifier le prix de vente hors taxes d'un produit passé en paramètre. Cette procédure commence par afficher le libellé et l'ancien prix de vente hors taxes du produit puis saisit le nouveau prix de vente entré par l'utilisateur.

```
Procédure majpv (E/S x: produit)
Début
    Aff "produit: ", x.lib
    Aff "prix de vente hors taxe actuel: ", x.pvht
    Aff "Entrez le nouveau prix de vente: "
    Saisir x.pvht
    Aff "le nouveau prix de vente est: ", x.pvht
FinProc
```

B. L'imbrication d'enregistrements

Supposons que dans le type personne, nous ne voulions plus l'âge de la personne, mais sa date de naissance. Une date est composée de trois variables (jour, mois, année) indissociables. Une date correspond donc à une entité du monde réel qu'on doit représenter par un type enregistrement à 3 champs.
Si on déclare le type date au préalable, on peut l'utiliser dans la déclaration du type personne pour le type de la date de naissance.

Un type structuré peut être utilisé comme type pour des champs d'un autre type structuré



Pour accéder à l'année de naissance d'une personne, il faut utiliser deux fois l'opérateur '.'
pers1.ddn.année

Il faut lire une telle variable **de droite à gauche** : l'année de la date de naissance de la personne 1.

Exemple Complet

Un produit (cf. ex précédents) est livré par un seul fournisseur. Un fournisseur est caractérisé par son code, sa raison sociale et son numéro de téléphone.

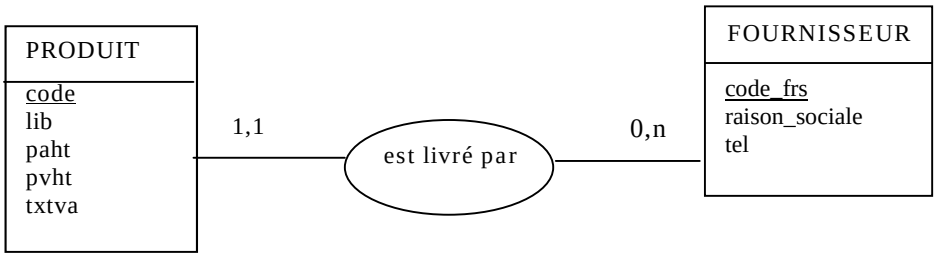
Type

```
Structure adresse
        num : entier
        rue: chaîne
        cp: chaîne
        ville: chaîne
FinStruct

Structure fournisseur
        code_frs : chaine
        raison_sociale: chaine
        ad_frs: adresse
        tel: chaine
FinStruct

Structure Produit
        code: chaîne
        lib: chaîne
        paht: réel
        pvht: réel
        txtva: réel
        frs: fournisseur
FinStruct

Var
   p: produit
```

Voilà l'instruction qui permet d'afficher le numéro de téléphone du fournisseur du produit p.frs.tel

```
Aff "téléphone du fournisseur de ", p.lib, " : ", p.frs.tel
```

Voilà le MCD correspondant. Déduisez en la correspondance entre l'imbrication des enregistrements et le type d'association entre 2 entités.



Fournisseur qui est en dépendance fonctionnelle sur produit est imbriqué dans produit.

De façon générale, une entité 1 en dépendance fonctionnelle sur une autre entité 2 est représentée en programmation par un type structuré imbriqué dans le type structuré correspondant à l'entité 1.

III. Les tableaux d'enregistrement (ou tables)

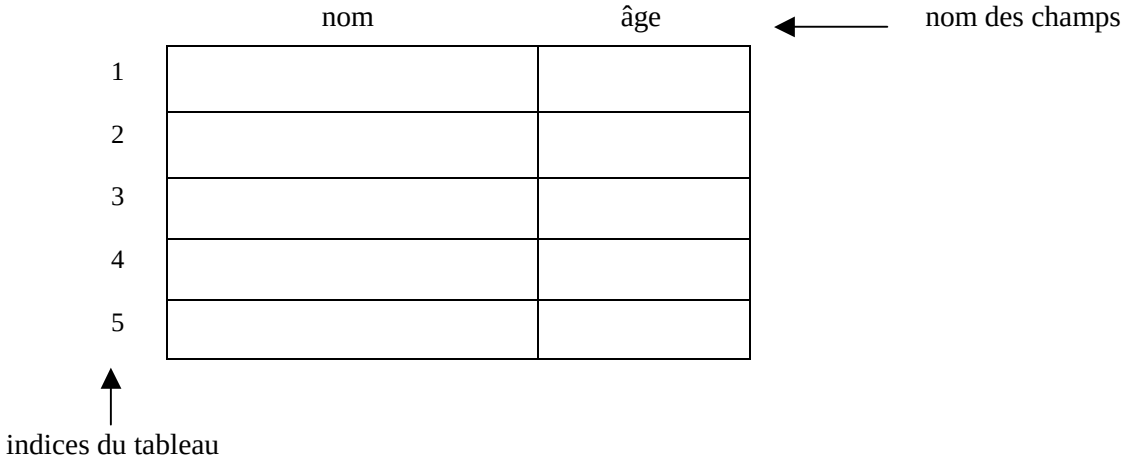
Il arrive souvent que l'on veuille traiter non pas un seul enregistrement mais plusieurs. Par exemple, on veut pouvoir traiter un groupe de personne. On ne va donc pas créer autant de variables du type personne qu'il y a de personnes. On va créer un tableau regroupant toutes les personnes du groupe. Il s'agit alors d'un tableau d'enregistrements.

```
Const
NP = 20                    // nombre de personnes du groupe

Type
Structure personne
    nom: chaîne
    age: entier
FinStruct
```

```
Var
groupe:  tableau[1..NP] de personnes
```

Chaque élément du tableau est un enregistrement, contenant plusieurs variables de type différent. On accède à un enregistrement par son indice dans le tableau.

groupe[2] représente la deuxième personne du groupe
groupe[2].nom représente le nom de la deuxième personne du groupe



Attention!

groupe.nom[3] n'est pas valide.

Pour accéder au nom de la troisième personne du tableau, il faut écrire g roupe[3].nom