Travaux pratiques n°12

Bases de données

F. Cuvellier, J. Tanoh

M.P.S.I. 1, 2013–2014

1 Présentation du problème

1.1 Exemples

Exercice 1. On veut constituer un carnet d'adresses. Chaque entrée comporte plusieurs attributs :

- Le nom,
- L'adresse,
- Le numéro de téléphone privé,
- La nature de la relation (amicale, professionnelle, etc.),
- L'activité professionnelle, éventuellement,
- Le numéro de téléphone professionnel, éventuellement.

C'est ainsi que l'on trouve

- Alain, 17 rue des Alpes, 451.23.22, amicale, mathématicien, 3.14.15.92,
- Bernard, 12 rue des Étoiles, 1.01.01.00, amicale, astronaute, 9.99.999,
- Claude, 17 rue des Alpes, 451.22.23, professionnelle, aviateur, 9.87.65.43,
- David, 5 boulevard Liouville, 2.03.005.0007, amicale, mathématicien,
- Ernest, 3 avenue des Alsaciens, 67.68.67.68, amicale,
- Fulbert, 2 place de Toulouse, 3.45.43, professionnelle, trapéziste, 8.88.880.

1. Proposer une structure de données, qui permette de rendre compte de toutes ces informations.
2. Écrire une fonction qui retourne les prénoms, une autre qui retourne l'adresse.
3. Écrire une fonction qui retourne l'ensemble des personnes exerçant une activité professionnelle donnée en argument.
4. Pour chacun des problèmes que traitent les fonctions précédentes, une autre structure de données permettrait-elle de simplifier la fonction ?

Exercice 2. Dans un établissement universitaire, trois classes de filières A et B regroupent un certain nombre d'étudiants, dont on connaît le lycée d'origine.

- Classe A1 : Alain (lycée Galois), Bernard (lycée Fourier),
- Classe A2 : Claude (lycée Poisson), David (lycée Galois),

- Classe B1 : Ernest (lycée Poisson), Fulbert (lycée Galois).

On se sert d'une liste pour ces données.

```
univ = [ ("A", 1, [("Alain", "Galois"), ("Bernard", "Fourier")]),
("A", 2, [("Claude", "Poisson"), ("David", "Galois")]),
("B", 1, [("Ernest", "Poisson"), ("Fulbert", "Galois")]) ]
```

1. Écrire une fonction qui retourne la liste des étudiants d'une classe donnée.
2. Donner une fonction qui retourne l'ensemble des étudiants qui proviennent d'un lycée donné.
3. Comment modifier la structure de données choisie pour simplifier la fonction précédente ?

1.2 Représentation dans le modèle relationnel

Dans le second des deux exemples précédents, chaque étudiant provient d'un certain lycée et se trouve dans une certaine classe. Il existe donc deux types d'appartenance le concernant, la structure de données choisie doit en privilégier une.
On convient de faire abstraction de tout type d'appartenance, mais on constate qu'on peut représenter de la même manière une classe et un étudiant : la classe est un couple (filière, numéro), un étudiant un triplet (nom, classe, lycée d'origine).
On appelle *relation* (ou *table*) une relation un tel couple, un tel triplet, ou d'une façon générale, un tel n-uplet. Cette définition sera précisée par la suite.
Étant donné un ensemble fini A, dont les éléments sont appelés *attributs* (par exemple, *filière*, *numéro*, *lycée*), on appelle *schéma relationnel* un n-uplet $(a_1, \ldots, a_n)$ d'attributs de A deux à deux distincts.
Le *domaine* d'un attribut a est un ensemble dont les éléments sont les valeurs possibles de l'attribut a. On le note $\mathrm{dom}(a)$.
Si S est un schéma relationnel, on note $b \in S$ pour signifier que b est un des attributs de S. Étant donné un ensemble X d'attributs, on note encore $X \subset S$ pour exprimer que chaque attribut de X est un attribut de S. On peut alors voir X comme sous-n-uplet de S en notant ses éléments dans l'ordre où ils apparaissent dans S.

Exercice 3. Préciser dans chacun des deux exemples précédents les attributs, les domaines des attributs et les schémas relationnels correspondants.

Une *relation* associée à un schéma relationnel S est un ensemble de n-uplets $(v_1, \ldots, v_n)$, où $v_1 \in \mathrm{dom}(a_1)$, $\ldots$, $v_n \in \mathrm{dom}(a_n)$. Pour préciser cette association, on note $R(S)$ la relation.

Exercice 4.

1. Donner un exemple de schéma relationnel pour la description d'une bibliothèque et deux exemples de relation.
2. Donner un exemple de schéma relationnel pour la description d'un zoo et deux exemples de relation.

1.3 Recherches

On veut exploiter les informations contenues dans une base de données. Pour cela on se sert d'un langage de requêtes, le langage SQL (*Structured Query Language*).
Par exemple, à partir d'une relation associée à un schéma (prénom, adresse, profession, téléphone), on doit être capable de sélectionner les valeurs dont un attribut donné a correspond à la valeur v ou à la valeur v' ou etc. On doit aussi pouvoir regrouper des ensembles de valeurs. Enfin, on doit pouvoir extraire un ou plusieurs attributs de chacune des valeurs.
Les opérations fondamentales correspondantes s'appellent la *sélection*, l'*union* et la *projection*.

2 Création d'une base de données

2.1 Usage de SQLite 3

Il existe de nombreux outils de gestion de bases de données, par exemple MySql, Oracle, etc. Tous respectent la syntaxe SQL, mais ils diffèrent par leur interface et certaines commandes pratiques.
Ici, on se propose d'employer SQLite 3.
Pour ouvrir une session interactive, on peut invoquer la commande sqlite3 dans une ligne de commande.
Pour créer une nouvelle base de données, par exemple BdD, il suffit de se servir de la commande `sqlite3 BdD.db`.
Si le fichier BdD existe, il est ouvert ; sinon, il est créé.
Pour créer une table *memos* et y insérer quelques enregistrements

```
sqlite3 BdD.db
SQLite version 3.?.?
Enter ‘‘.help’’ for instructions
sqlite> create table memos(text, priority INTEGER);
sqlite> insert into memos values(’reunion sur le projet’, 10);
sqlite> insert into memos values(’repas avec Berthe’, -20);
sqlite> insert into memos values(’etudier la Geometrie algebrique’, 1000);
sqlite> select * from memos;
reunion sur le projet|10
repas avec Berthe|-20
etudier la Geometrie algebrique|1000
sqlite>
```

On peut aussi se servir de méta-commandes pour contrôler les formats de sortie, les fichiers de base de données attachés, etc. Toutes sont précédées d'un point (.). Pour les visualiser, on peut se servir de la méta-commande `.help`. Par exemple,

```
sqlite> .help
.databases             List names and files of attached databases
.exit                  Exit this program
.import FILE TABLE     Import data from FILE into TABLE
.indices TABLE         Show names of all indices on TABLE
.mode MODE ?TABLE?     Set output mode where MODE is one of:
    csv      Comma-separated values
    column   Left-aligned columns.  (See .width)
    html     HTML <table> code
    insert   SQL insert statements for TABLE
    line     One value per line
    list     Values delimited by .separator string
    tabs     Tab-separated values
    tcl      TCL list elements
.nullvalue STRING      Print STRING in place of NULL values
.output FILENAME       Send output to FILENAME
.output stdout         Send output to the screen
.quit                  Exit this program
.read FILENAME         Execute SQL in FILENAME
.schema ?TABLE?        Show the CREATE statements
.separator STRING      Change separator used by output mode and .import
```

On peut créer des tables dans des fichiers appropriés et les importer dans la base de données.

2.2 Exercices

Exercice 5. Créer une base de données dont les tables proviennent des exemples précédents.

Exercice 6.

1. Créer une table de données pour une bibliothèque. Son schéma relationnel aura trois attributs : titre, auteur, année de parution *(dans cet ordre)*. (Les noms n'ont pas d'importance, ce sont des chaînes de caractères, mais l'année est un nombre.)
2. Créer une table de données pour un lycée. Son schéma relationnel aura trois attributs : classe, professeur de mathématiques, effectif. (L'effectif est un nombre, les autres attributs des chaînes de caractères.)
3. Effectuer des requêtes sur l'effectif, sur l'auteur.
4. Renommer les attributs de la seconde table pour qu'ils soient les mêmes que dans la première table.
5. Calculer l'intersection, l'union des tables.