Sujet proposé par Jef Wijsen

Ce projet s'intitule *Interpréteur pour l'Algèbre Relationnelle.*
Les exigences minimales du projet sont les suivantes: le but est d'implanter un outil didactique qui permet aux étudiants en bases de données de saisir une requête en algèbre relationnelle et d'en voir le résultat.
La syntaxe est comme suit:

```
<relational expression> ::= <relvar name> | <relational operation> | (<relational expression>)
<relational operation> ::= <select> | <project> | <join> | <rename> | <union> | <difference>
<select> ::= <relational expression> WHERE <boolean expression>
<project> ::= <relational expression> PROJECT <attribute name commalist>
<join> ::= <relational expression> JOIN <relational expression>
<rename> ::= <relational expression> RENAME <renaming commalist>
<union> ::= <relational expression> UNION <relational expression>
<difference> ::= <relational expression> MINUS <relational expression>
```

Pour la sémantique, voir le cours de bases de données ou un livre introductif tel que C. Date, *An Introduction to Database Systems,* Addison- Wesley, seventh edition, 2000.

Par exemple, soient *R* et *S* comme suit:

R	***A***	***B***
	a	1
	b	2
	c	3

S	***B***	***C***	***D***
	1	α	I
	2	β	II
	2	γ	III
	4	δ	IV

Alors le résultat aux requêtes suivantes sont :

(S WHERE B = "2") RENAME C AS E, D AS F

B	***E***	***F***
2	β	II
2	γ	III

et

R JOIN (S PROJECT B, C)

A	***B***	***C***
a	1	α
a	1	β
b	2	γ

Les extensions possibles du projet sont :
- Ajout des fonctions d'aggrégat (AVG, COUNT, MAX, MIN, SUM).
- Optimisation syntaxique des requêtes.
- Interface conviviale.

Interpréteur pour l'Algèbre Relationnelle

Jef Wijsen

November 27, 2003

1 La Syntaxe

Le but est d'implanter un outil didactique qui permet aux étudiants en bases de données de saisir une requête en algèbre relationnelle et d'en voir le résultat. La syntaxe est comme suit:

```
<relational expression> ::= <relvar name> | <relational operation> |
                            (<relational expression>)
<relational operation> ::= <select> | <project> | <join> | <rename> |
                           <union> | <difference>
<select> ::= <relational expression> WHERE <boolean expression>
<project> ::= <relational expression> PROJECT <attribute name commalist>
<join> ::= <relational expression> JOIN <relational expression>
<rename> ::= <relational expression> RENAME <renaming commalist>
<union> ::= <relational expression> UNION <relational expression>
<difference> ::= <relational expression> MINUS <relational expression>
```

Pour la sémantique, voir le cours de bases de données ou un livre introductif tel que [1]. Par exemple, soient R et S comme suit:

R

A	B

S

B	C	D

Alors:

(S WHERE B = "2") RENAME C AS E, D AS F

B	E	F
2	β	II
2	γ	III

et

R JOIN (S PROJECT B, C)

A	B	C

2 Le Simple Query Format

Spécifions le *simple query format (sqf)*. Fig. 1 montre le contenu du fichier `input.sqf`. Les tables `COURS` et `NOTES` sont faciles à comprendre. L'attribut `Ects` donne le nombre de crédits pour un cours. Les données encodées sont comme suit:

COURS

Nom	Prof	Ects
Gestion de projets		
Bases de donnees		
Analyse		

NOTES

Nom	C	Note
Ed	Gestion de projets	13
Ed	Bases de donnees	15
Tim	Bases de donnees	13
Eric	Gestion de projets	19

```
% Ceci est le contenu du fichier input.sqf.
% On decrit d'abord les tables.
@relation COURS
@attribute Nom
@attribute Prof
@attribute Ects
@data
Gestion de projets, Mens, 4
Bases de donnees, Wijsen, 10
Analyse, Troestler, 10
@relation NOTES
@attribute Nom
@atrribute C
@attribute Note
@data
Ed, Gestion de projets, 13
Ed, Bases de donnees, 15
Tim, Bases de donnees, 13
Eric, Gestion de projets, 19
% Puis on specifie les requetes.
@let CCOURS = COURS RENAME Nom AS C
@let M = (CCours WHERE Prof="Mens") PROJECT C
@let EM = (M JOIN NOTES) PROJECT Nom
@print EM
@let A_EU = (CCOURS JOIN NOTES) PROJECT Nom, Prof
@let TOUT = (COURS PROJECT Prof) JOIN (NOTES PROJECT Nom)
@let N_A_PAS_EU = (TOUT MINUS A_EU) RENAME Nom AS Etudiant
@print N_A_PAS_EU
```

Figure 1: Proposition du *simple query format* à l'aide du contenu du fichier `input.sqf`

Les requêtes `EM` et `N_A_PAS_EU` répondent aux questions:

- Qui a suivi un cours de Mens?
- Quel étudiant n'a pas eu quel professeur?

3 La commande

Je propose la commande `spjrud`:[1]

```
spjrud input.sqf output.sqf
```

Le résultat est montré par Fig. 2.

Il y a des situations où, au lieu d'avoir un seul fichier d'entrée, on sohaite stocker les données et les requêtes en deux fichiers séparées. Le fichier avec les requêtes ne contient que des lignes qui commencent avec `@let` et `@print`. Le fichier avec les données contient toutes les autres lignes. A ce but, je propose une option `-d`:

```
spjrud [-d data.sqf] queries.sqf output.sqf
```

Le résulat de cette commande serait équivalente à la commande

[1] Cette proposition me semble raisonnable. Néanmoins, je suis ouvert à d'autres propositions.

```
spjrud dataqueries.sqf output.sqf
```

si `dataqueires.sqf` était la concaténation du fichier `data.sqf` suivi par `queires.sqf`.

Par exemple, Fig. 3 montre un fichier `queries.sqf` avec une requête qui porte sur les données en `output.sqf` de la Fig. 2.[2] Au lieu de copier-coller les données et les requêtes en un seul fichier, j'exécute:

```
spjrud -d output.sqf queires.sqf outputbis.sqf
```

Fig. 4 montre le fichier qui en résulte.

4 Extensions Possibles

- Ajout des fonctions d'aggrégat (`AVG`, `COUNT`, `MAX`, `MIN`, `SUM`).
- Optimisation syntaxique des requêtes.
- Interface conviviale. Est-ce vraiment intéressant?

5 Remarque

Notez que l'ordre des attributs (et des tuples) n'a aucune importance dans cette algèbre. Donc, le résultat montré en Fig. 5 peut remplacer celui de Fig. 4.

Références

[1] C. Date. *An Introduction to Database Systems.* Addison-Wesley, seventh edition, 2000.

[2] Notez que le résulat d'une requête est toujours un fichier en *simple query format* qui contient des données avec leur schéma, mais qui ne contient pas de requêtes.

```
@relation EM
@attribut Nom
@data
Ed
Eric
@relation N_A_PAS_EU
@attribute Etudiant
@attribute Prof
@data
Ed, Troestler
Tim, Troestler
Eric, Troestler
Eric, Wijsen
Tim, Mens
```

Figure 2: Le fichier `output.sqf` qui résulte de la commande `spjrud input.sqf output.sqf`.

```
@let R = (EM RENAME Nom as Etudiant) JOIN N_A_PAS_EU
@print R
```

Figure 3: Le contenu du ficher `queries.sqf`

```
@relation R
@attribute Etudiant
@attribute Prof
@data
Ed, Troestler
Eric, Troestler
Eric, Wijsen
```

Figure 4: Le fichier `outputbis.sqf` qui est le résultat de la commande `spjrud -d output.sqf queries.sqf outputbis.sqf`.

```
@relation R
@attribute Prof
@attribute Etudiant
@data
Troestler, Ed
Wijsen, Eric
Troestler, Eric
```

Figure 5: L'ordere des attributs est sans importance.