**Introduction aux bases de données relationnelles**

# TP6 : Premiers contacts avec DATALOG

Le but de cette séance de TP est de vous familiariser avec l'environnement DATALOG et PROLOG. Nous utiliserons DES freeware de DATALOG de l'université de Madrid. Lui-même, DES repose sur SWI-PROLOG, freeware PROLOG de l'université d'Amsterdam. Vous pouvez récupérez des distributions (linux, windows ou Max OS X) de DES librement sur le site http://www.fdi.ucm.es/profesor/fernan/des/ et de SWI-PROLOG sur le site http://www.swi-prolog.org.

## DES

- Le binaire DES (version 3.10) est disponible au M5 dans */home/enseign/DES/des*.
- Le signal d'invité "DES>" attend que vous saisissiez un *but* (requete) que le moteur DATALOG tentera de satisfaire.
- Puisque vous n'avez pas encore chargé de base, vous ne pouvez pas encore faire grand chose. Une possibilité est de tester une comparaison 42=42. Observez le résultat. Comparez avec le résultat de 42=2015.
- Dans l'interprète, vous pouvez changer de répertoire avec /cd , faire afficher le répertoire courant avec /pwd, et charger une base (nommée *mabase.dl*) avec /consult mabase.dl
- Dans l'interprète, vous pouvez obtenir de l' aide en tapant /help.
- Pour quitter DES, tapez /terminate. Vous vous retrouverez ensuite au niveau de SWI-PROLOG, donc l'interprète vous affiche "?-". Pour quitter SWI-PROLOG, tapez halt. (avec le "." !).

*Rappel : en* DATALOG*, tout identificateur commençant par une majuscule est considéré comme une variable, et les prédicats avec paramètres doivent être immédiatement suivis d'une parenthèse ouvrante (il ne doit pas y avoir d'espace entre la fin du nom du prédicat et cette parenthèse).*

# 1 Histoires de jalousie

On considère la base de connaissance suivante :
- *mia* est une femme ;
- *jody* est une femme ;
- *yolande* est une femme ;
- *vincent* aime *mia* ;
- *vincent* aime *pierre* ;
- *marcellus* aime *mia* ;
- *mon_chou* aime *lapin* ;
- *lapin* aime *mon_chou* ;
- X est jaloux de Y s'ils aiment tous les deux une même personne.

**Question 1** Saisissez les assertions de cette base dans un fichier [1] sous la forme de *faits* DATALOGqui définissent les prédicats suivants : $femme/1, aime/2, est\_jaloux\_de/2$

**Question 2**

Chargez votre base. Pour cela vous utiliserez le prédicat consult (essayez help consult), en lui donnant en paramètre le nom de votre fichier qui doit avoir l'extension .dl : "consult <nom de fichier> " avec ou sans l'extension. Au chargement, votre fichier est évalué syntaxiquement par l'interprète DATALOG, toute erreur de syntaxe éventuelle est alors annoncée. Vous pouvez également simplement taper "/c <nom de fichier>" Voici différentes possibilités équivalentes pour charger un fichier qui s'appellerait exemple.dl dans le répertoire de travail courant (celui indiqué par "/pwd") :

```
DES> /c exemple
DES> /consult exemple
DES> /c exemple.dl
DES> /consult exemple.dl
```

1. Utilisez l'éditeur de textes que vous souhaitez : emacs, KWrite, Kate, etc. Emacs et Kate vous permettent d'avoir dans une même fenêtre l'éditeur et un shell dans lequel vous pouvez exécuter SWI-PROLOG

Lorsqu'un fichier est chargé (par /consult) la bases chargée précédemment est "oubliée". Donc un "\consult" réinitialise la base de donnée extensionnelle dans le fichier "consulté".

Sachez que Datalog mémorise les résultats déjà calculés. Vous pouvez les afficher à tout moment avec la commande /list_et .

**Question 3** Avec la commande

```
DES> /listing
```

vous pouvez afficher le contenu de la base de connaissance courante.

Ce prédicat peut également prendre en argument le prédicat dont on souhaite connaître la définition courante. Par exemple :

```
DES> /listing femme
```

**Question 4** Vérifiez (en définissant un *but* DATALOG) que *mia* et *yolande* sont des femmes mais que lapin n'est pas une femme.

**Question 5** Interrogez DATALOG pour connaître tous les noms de femmes.

**Question 6** Interrogez DATALOG pour connaître toutes les femmes que *vincent* aime.

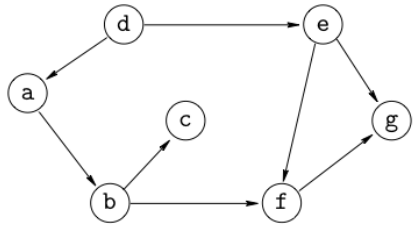
**Question 7** Interrogez DATALOG pour connaître tous les hommes que *vincent* aime. Quel problème observez-vous ? Pourquoi ? Comment y remédier ?

**Question 8** Interrogez DATALOG pour connaître tous ceux dont *vincent* est jaloux.

Lorsqu'on doit utiliser une variable mais qu'on ne désire pas connaître son instanciation on utilise une *variable anonyme*. Celle-ci est représentée par le symbole _ (tiret bas).

**Question 9** En utilisant un variable anonyme, interrogez DATALOG pour savoir si *vincent* est jaloux. En déduire la définition du prédicat correspondant.

# 2 Graphes




**Question 10** Saisissez le graphe de l'image dans un nouveau fichier *graph.dl*. Utilisez pour cela le $e/2$ définit en cours (e pour edge en anglais).

**Question 11** Utilisez le prédicat $p/2$ pour afficher toutes les paires de sommets connectés, quelque soit la longueur du chemin (p pour path en anglais). Rendez votre requête, et son résultat.

**Question 12** Ajoutez un ou plusieurs cycles au graphe. Puis, retestez $p/2$.

**Question 13** Pouvez-vous determiner combien de fois un cycle est traversé ?

**Question 14** Déclarez un prédicat $impair(X, Y)$ qui est vrai s'il existe un chemin de longueur impaire entre les sommets $X$ et $Y$. A l'aide d'une requête, affichez les paires de sommets, entre lesquels existe un chemin de longueur impaire.

# 3 Les princesses et les tigres (2)

*Pour résoudre cet exercise, nous vous invitons à utiliser la correction des princesses et tigres (partie 1) disponible sur moodle.*

Le premier prisonnier à réussi à s'en sortir. Le roi, mécontent (il était un peu sadique), modifie les affiches puis en fait venir un deuxième. Voici ce que le nouveau prisonnier pouvait lire :

| – 1 – | – 2 – |
|---|---|
| Il y a une princesse dans cette cellule et un tigre dans l'autre | Il y a une princesse dans une cellule et il y a un tigre dans une cellule |

Seulement le roi a aussi modifié les règles du jeu : maintenant, *une des deux affiches ment, et l'autre dit la vérité* ! Répondez aux questions suivantes :

**Question 15** Discutez avec vos voisins, essayez de trouver une solution sans formaliser. Ne passez pas plus de 5 minutes là-dessus.

**Question 16** Traduisez les deux affiches en formules logiques.

**Question 17** Écrivez une formule logique qui inclut aussi bien le contenu des affiches, que les règles du jeux pour ce jour.

**Question 18** Implémentez un programme qui résout le puzzle en Datalog, en suivant la solution du premier puzzle.

# 4 Résolution en SWI Prolog

Enregistrez le fichier *crisefinanciere.pl* disponible sur moodle, et lisez-le attentivement. Veillez à conserver le nom crisefinanciere.pl avec l'extension **.pl** pour prolog. Dans le répertoire où vous avez enregistré le fichier, lancez SWI prolog à partir du terminal (commande : swipl). Vous verrez un affichage similaire au suivant :

```
Welcome to SWI-Prolog (Multi-threaded, 64 bits,
Version 5.10.4)
Copyright (c) 1990-2011 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free
software, and you are welcome to redistribute it under
certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

?-
```

puis, chargez le fichier :

```
?- [crisefinanciere].
% crisefinanciere compiled 0.00 sec, 1,736 bytes
true.
?-
```

Sachez que SWI prolog

- attend un point en fin de requête (ce point est optionnel en DES).
- affiche les réponses une par une. Vous obtenez la réponse suivante en frappant le point-virgule. Quand vous frappez (ENTER) après l'affichage d'un résultat, SWI Prolog abandonne la requête courante, sans calculer d'autres réponses.

**Question 19** Posez une requête permettant de vous faire afficher les dettes.

**Question 20** Posez une requête pour afficher quelle fille évite quelle autre. Assurez-vous de faire afficher *toutes* les réponses. Qu'observez-vous ?

**Question 21** Comparez le résultat des mêmes requêtes en DES, et expliquez d'où viennent les différences entre les deux systèmes.

# 5 Parsing de langage naturel

Le but du dernier excercice de programmation logique est d'écrire, en Datalog, un parser pour un anglais simplifié. La méthode repose sur l'algorithme CYK pour le parsing des grammaires algébriques. Il n'est pas nécessaire de connaitre les détails de cet algorithme pour faire le travail pratique, mais vous êtes encouragés de rattraper cette lecture à l'occasion (voir Wikipedia, consulter aussi bien la version en anglais qu'en francais !).

Le point de départ est un dictionnaire avec plusieurs catégories de mots :

| Catégorie | mots |
|---|---|
| N | man, song, unicorn, telescope |
| Det | the, a |
| V | sees, sings |
| Prep | with |

où $N$ signifie noun, $V$ verb (verbe), $Det$ determiner. Un déterminant est, par exemple, un article défini (the) ou indéfini (a).

A partir du dictionnaire, la grammaire permet de construire des sous-phrases nominales $NP$ (noun phrase) et verbales $VP$ (verb phrase). Des exemples de $NP$ sont *the man* et *a song*. Un exemple de $VP$ est *sings a song*. Les règles correspondantes sont :

$$NP -> Det\ N$$
$$VP -> V\ NP$$

La règle pour construire une phrase $S$ (sentence) est :

$$S -> NP\ \ VP$$

**Question 22** Dessinez l'arbre syntaxique pour la phrase 1 : *The man sings a song*. Rendez une version ASCI en suivant l'exemple

$$((the\ DET)(man\ N)\ NP)$$

dans un fichier *arbresphrase*1.*txt*.

**Question 23** Dans un fichier nommé *phrase*1.*dl*, codez la phrase *The man sings a song* sous forme de faits datalog, suivant le schéma :

```
 the(0,1).
 man(1,2).
...
```

Sachez que vous pouvez charger plusieurs fichiers avec DES de la manière suivante :

```
DES> /[file1,file2,file3]
```

**Question 24** Codez le dictionnaire, dans un fichier *dictionnaire*1.*dl* . Il faut, pour chaque catégorie de mots apparaissant dans la grammaire, définir un prédicat IDB, et saisir chaque mot à l'aide d'une règle pour le prédicat correspondant.

Votre codage doit permettre d'obtenir les réponses suivantes : Pour la catégorie déterminant (*det*) :

```
DES> det(X,Y)

{
  det(0,1),
  det(3,4)
}
Info: 2 tuples computed.
```

Et pour la catégorie des substantifs (*noun*)

```
DES> n(X,Y)

{
  n(1,2),
  n(4,5)
}
Info: 2 tuples computed.
```

Notez que votre dictionnaire ne doit *pas* refléter l'ordre des mots dans la phrase 1.

**Question 25** Codez les trois règles de la grammaire anglaise, données en début de sujet, dans un fichier *grammaire*1.*dl*. Si votre codage fonctionne, vous devez parvenir à parser la phrase entière, ou la phrase nominale *the man* qui commence à la position 0. Test :

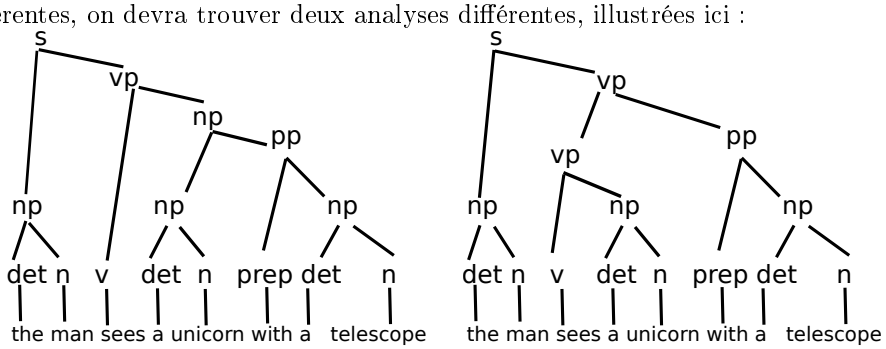
```
DES> np(0,X)

{
  np(0,2)
}
Info: 1 tuple computed.
```

**Question 26** Comment tester si la phrase *the man sings a song* peut être parsée ?

Le prochain but est de pouvoir parser de manière non-deterministe, ce qui permet d'obtenir différentes analyses syntaxiques pour une phrase, s'il en existe plusieurs.

**Exemple :** Pour la phrase 2 *The man sees a unicorn with a telescope*, ayant deux interprétations différentes, on devra trouver deux analyses différentes, illustrées ici :




**Question 27** Donnez, en français, une traduction de la phrase 2, qui préserve les ambiguïtés. Puis, expliquez les deux interprétations différentes de la phrase. Quelle est la différence sémantique ?

**Question 28** Coder la phrase *The man sees a unicorn with a telescope* en Datalog. Enregistrez-la dans un fichier *phrase2.dl*.

Le but concret est d'enrichir le dictionnaire et la grammaire pour pouvoir traiter des phrases prépositionnelles ($PP$), avec des prépositions telles que *with*. La première étape est de trouver les nouvelles règles, permettant les analyses alternatives de la phrase *The man sees a unicorn with a telescope*, illustrées dans les figures.

**Question 29** Quelles sont les règles de la grammaire pour traiter les PPs ?

**Question 30** Ajoutez le nécessaire au dictionnaire, et enregistrez le dictionnaire complet dans *dictionnaire2.dl*

**Question 31** Codez en Datalog les règles de la grammaire concernant les PPs. Enregistrez l'ensemble des règles de la grammaire dans un fichier *grammaire2.dl*.

**Question 32** Proposez une requête qui montre que votre parseur trouve deux analyses différentes pour la phrase 2. Alternativement, trouvez une trace des deux analyse différentes dans l'information mémorisée.

**Question 33** **Difficile.** Proposez une manière de traiter les adjectifs, par exemple, dans la phrase *The tall man sees a white unicorn with a long telescope*, mais egalement dans la phrase *The tall and lucky man sees a white unicorn with a telescope.*