TD1 – TdL – Corrige

Exercice 3

Question 1. Nous nous plaçons ici dans le cadre de l'analyse lexicale permettant de reconnaître les lexèmes utilisés dans une requête SQL. Donner une grammaire permettant d'identifier les différents lexèmes nécessaires pour les requêtes de ce langage SQL.

Solution. Encore une fois, il y a plusieurs réponses possible en fonction des lexèmes que vous avez choisis.

Ici, nous supposons que les lexèmes reconnus lors de l'analyse lexicales sont : select, from, where, champ, table, condition , virgule et point-virgule

Pour simplifier, on pose les 2 ensembles suivants :

$$\text{lettre} = \{a, ..., z, A, ..., Z\}$$
$$\text{chiffre} = \{0, ..., 9\}$$

On a $G =< T, N, S, P >$ avec :

$T = \{\textbf{SELECT}, \textbf{FROM}, \textbf{WHERE}, \textbf{OR}, \textbf{AND}, ,, ', <, =, >, (,), ;\} \cup \text{lettre} \cup \text{chiffre}$

$N = \{\text{lexemeSQL}, \text{select}, \text{from}, \text{where}, \text{comparaison}, \text{oper}, \text{identificateur}, \text{constante}, \text{chaîne}, \text{entier}, \text{condition}, \text{champ}, \text{table}, \text{point-virgule}\}$

$S = \text{lexemeSQL}$

$P =$

lexemeSQL	$\rightarrow$	select | from | where | champ | table | condition | virgule |point-virgule
virgule	$\rightarrow$	,
identificateur	$\rightarrow$	lettre | identificateur lettre | identificateur chiffre
entier	$\rightarrow$	chiffre | entier chiffre
chaîne	$\rightarrow$	lettre | chiffre | chaîne lettre | chaîne chiffre
constante	$\rightarrow$	' chaîne ' | entier
oper	$\rightarrow$	**AND** | **OR**
comparaison	$\rightarrow$	$< \mid = \mid >$
condition	$\rightarrow$	identificateur comparaison constante | condition oper condition
table	$\rightarrow$	identificateur
champ	$\rightarrow$	identificateur
select	$\rightarrow$	**SELECT**
from	$\rightarrow$	**FROM**
where	$\rightarrow$	**WHERE**

point-virgule --> ;

Les règles telles que entier $\rightarrow$ chiffre **doivent être interprétées comme une succession de symboles** | **:** entier $\rightarrow 0|1|...|9$.

Question 2. Cette fois-ci, nous nous allons procéder à l'analyse syntaxique d'une requête SQL. En fonction des lexèmes obtenus lors de la question précédente, proposer une grammaire hors-contexte qui reconnait les requêtes syntaxiquement bien formées.

Les lexèmes obtenus précédemment deviennent donc les éléments terminaux de cette grammaire.

Solution. On a $G =< T, N, S, P >$ avec :

$$T = \{\text{select} \mid \text{from} \mid \text{where} \mid \text{champ} \mid \text{table} \mid \text{condition} \mid \text{virgule} \mid \text{point-virgule}\}$$
$$N = \{\text{requête}, \text{listeChamps}\}$$
$$S = \text{requête}$$
$$P = \left\{ \begin{array}{rl} \text{requête} \rightarrow & \text{select listeChamps from table where condition point-virgule} \\ \text{listeChamps} \rightarrow & \text{champ} \mid \text{listeChamps virgule listeChamps} \end{array} \right\}$$

Exercice 4.

Langages binaires

Soit A = {a, b} un alphabet binaire.

Question 1. Proposer une grammaire formelle définissant chacun des langages suivants :

Solution :
On suppose : T = A, S = S et N contient tous les symboles introduits par les règles de production.

Les règles de production des grammaires.

1. Le langage des mots de taille 6 : A^6

S → LLLLLL
L → a | b

2. Le langage de tous les mots : A*

S → ε | aS | bS

3. Le langage des mots commencant par le symbole a et finissant par le symbole b : {a·u·b | u ∈ A*}

S → aKb
K → ε | aK | bK

4. Le langage des palindromes : {ε, a, b, aa, bb, aba, bab, …}

S → ε | a | b | aSa | bSb

5. Le langage des mots tels que le symbole **a** n'est jamais suivi d'un autre symbole a :
{ε, a, ab, ba, bb, aba, abb, bab, bba, bbb, …}

S → ε | a | abS | bS

6. L'intersection des deux langages précédents

S → ε | a | b | aba | abSba | bSb

7. Le langage des mots contenant autant de **a** que de **b**

S → ε | SS | aSb | bSa

Il existe toujours plusieurs grammaires pour représenter un langage.

Ici, les propositions ne sont que des exemples de grammaires solutions.

Pour prouver que ces grammaires produisent bien le langage voulu il faut raisonner en deux temps :
(a) Soit **u** un mot du langage. Quelle forme peut avoir **u** ? Comment produire **u** à partir de la grammaire ?
(b) Soit **u** un mot produit par la grammaire. Quelle forme peut avoir **u** ? Pourquoi **u** appartient-il bien au langage ?