

# Logic Programming

---

## Section #8

One

14/12/2009

## Expert System Project

ال section دة بيتكلم عن ال expert systems ودة من أهم التطبيقات اللي بنحتاج فيها ال Logic Programming . طيب ايه هو ال expert system ال ES هو عامل زي الدكتور وظيفته ال Diagnosis يعني ال ES دة ممكن يكون برنامج بيقدر يتعرف على ال plants ويقدر يعرف نوعها أو برنامج بيقدر يتعرف على ال animals أو ممكن يبقى دكتور يشخص حالة المريض وهكذا... طيب ايه هي كيفية عمل ال ES يعني ال input بتبقى عبارة عن ايه وال output ايه؟ ال input بيبقى عبارة عن مجموعة أسئلة وانت ك user بتجاوب عليها وبعدين على حسب اجاباتك يقدر يشخص ال computer أو يتعرف على الحاجة اللي انت عايزها. طبعاً الموضوع مش بالسهولة دي ☺ دلوقتي هنشوف ال Example دة اللي أخذناه في ال section اللي هو ال "Animal Kingdom".

- **Example: Animal Kingdom**

Cheetah	Tiger	Giraffe
1. It is a mammal 2. Has tawny-color 3. Has dark-spots	1. It is a mammal 2. Has tawny-color 3. Has black-strips	1. It is an ungulate 2. Has long-neck 3. Has long-legs 4. Has dark- spots

-It is a mammal if it has a hair.

-It is an ungulate if it has a hooves.

احنا هنا مسكنا animal animal وخطينا ليه ال properties بتاعته.

الفكرة كلها في ال ES اني ازاى السؤال اللي سألته مسألوش ثاني في ال case الثانية + ازاى أرتب الأسئلة بحيث أخليه smart أكثر.

---

- **Console Project**

### database

```
xpositive(symbol,symbol)
xnegative(symbol,symbol)
```

عملنا database file وخطينا فيه 2 facts ال xpositive و ال xnegative طب احنا هنا عملنا database ليه؟ عشان لو فكرت في ال program هتلاقي انك محتاج مكان تخزن فيه ال previous answers عشان لما انتقل

لل case الجديدة مسائل ال user نفس الأسئلة تاني اذا كان في حاجة common بين ال cases. هنتكلم عن ال facts قدام شوية . طيب نشوف ال region اللي بعده.

### **predicates**

```
nondeterm animal(symbol)
nondeterm positive(symbol,symbol)
nondeterm negative(symbol,symbol)
nondeterm it_is(symbol)
ask(symbol,symbol,symbol)
nondeterm run
```

هنا في ال region بتاع ال Predicates عرفنا 6 relations هنعرف واحد واحد لما نخش في ال clauses.

---

### **clauses**

```
animal(cheetah):-
    it_is(mammal),
    positive(has,tawny_color),
    positive(has,dark_spots) .

animal(tiger):-
    it_is(mammal),
    positive(has,tawny_color),
    positive(has,black_stripes) .

animal(giraffe):-
    it_is(ungulate),
    positive(has,long_neck),
    positive(has,long_legs),
    positive(has,dark_spots) .
```

احنا هنا في ال region clauses عرفنا 3 animal rules زي ما احنا شايفين . يعني لو مسكنا أول rule بتقول ايه؟ بتقول ان ال animal هيبقى Cheetah لو اتحقق ال facts 3 دول انه يبقى mammal وفي نفس الوقت يبقى tawny color و عنده dark spots . طيب عايزين نمسك fact fact ونشوف بيعملوا ايه.

```

it_is(mammal):-
    positive(has,hair) .
it_is(ungulate):-
    positive(has,hooves) .

```

نشوف ال fact it\_is() بتعمل ايه:

لو مسكنا أول rule بتقول انه mammal لو it has a hair ومحطوة ك parameters في fact اسمها positive بتعمل ايه ؟ تعالوا نشوف.

```

positive(X,Y):-
    xpositive(X,Y) , ! .
positive(X,Y):-
    not(xnegative(X,Y)) , ask(X,Y,yes) , ! .

positive(X,Y):-
    not(xnegative(X,Y)) ,
    assert(xnegative(X,Y)) , fail .

```

هو دة الجزء اللي فيه النتيجة ☺ ليه؟ تعال نشوف ال rule اللي اسمها positive دي بتعمل ايه. أولا positive دي بتقول هل ال animal دة عنده ال X,Y property ولا لا. طيب ال positive متقسمة 3 cases زي ما أحنا شايفين اللي هما:-

(a) ال property X,Y موجودة فعلاً في ال database في ال fact xpositive وساعتها هيرجع true وي '!' Cut يعني ميخشش في أي positive لانه أتأكد ان ال property دي موجودة في ال animal دة.

(b) ال case الثانية بتقول انه failed في ال case 1 ودخل في Case 2 طيب هيعمل ايه؟ أول حاجة هيت check ان ال property X,Y مش موجودة في ال xnegative. ال xnegative دي fact موجودة في ال database وهي عكس ال xpositive وبتقول ان ال property الفلانية مش موجودة في ال animal الفلاني... نرجع لل case قولنا هت check ال property موجودة في ال xneg. لو رجعت no ودة اللي أنا عايزه هن negate ب "not" فترجع true. تمام نخش على ال sub goal اللي هو ال ask. ما هو مادام ال property مش موجودة في ال xpos. ولا ال xneg خلاص نسال ال user ويجاوب ب yes أو no هنعرف ال ask rule دي شكلها ازاي بعد شوية. وطبعاً لو ال 2 sub goals رجعوا true ساعتها برضة هي Cut.

(c) ال case 3 امتى هيخشها. هيخشها لو ال 2 cases failed previous يعني ال property مش موجودة في ال xpos. ولما ال ask ال user جاوب عليه ب no ساعتها هيخش case 3 وهيأكد برضه انها مش موجودة في ال xneg. وهيضيفها في ال xneg. وبعدين هي fail ليه؟ عشان هو لما وصل case 3 معناها ان property X,Y مش موجودة في ال animal Z فهي fail ويخش على ال animal اللي بعد كدة. جرب تشيلها و trace هتعرف قيمتها ☺

```

negative(X,Y):-
    xnegative(X,Y),!.
negative(X,Y):-
    not(xpositive(X,Y)),ask(X,Y,no),!.

negative(X,Y):-
    not(xpositive(X,Y)),
    assert(xpositive(X,Y)),fail.

```

ال negative بتمشي بنفس ال behavior بتاع ال positive و ال negative case دي احنا عاملناها زيادة بحيث ان كان عندك animal فيها property منفية زي مثلاً لما أقول في ال cheetah doesn't have hooves.

---

بعد كدا في ال ask clauses:

```

ask(X,Y,yes):-
    write(X," it ",Y," ? "),
    readln(Reply),
    frontchar(Reply,'y',_),
    assert(xpositive(X,Y)).

ask(X,Y,no):-
    write(X,"it",Y),
    readln(Reply),
    frontchar(Reply,'n',_),
    assert(xnegative(X,Y)).

```

ال ask rule دي وظيفتها انها تسأل ال user ب write built in fn. وبعد كدة ت read من ال user الأجابة فلو دخلوا yes أو y أو yup أو أي حاجة بتبتدي ب 'Y' ساعتها يبقى ال user موافق ان ال property X,Y موجودة في ال animal اللي بيدور عليه وبالتالي هنضيفه في ال database.

ال fn. frontchar(string,'s','tring') دي هي المسنولة عن انها بتتأكد ان أول char. في ال string اللي دخله ال user 'Y'.

وطبعاً ال ask الثانية بتشتغل بنفس ال behavior بتاع ال ask الأولانية بس الفرق هنا ان لو ال user أجاب ب no.

---

نروح لآخر حاجة في ال code..

```

run:-
    animal(Z),write(Z),nl.
run:-
    write("Unknown"),nl.

goal
run.

```

هنا ال goal بتاعي run بترجعلي yes أو no. ال rule الأولانية بتاعت run بتروح تنادي animal ولو رجعت ال code بتاع animal ب true هيحط اسم ال animal في Z وهي write ال Z على ال screen و هينزل سطر.

أما ال run الثانية هيوخش فيها لو failed في الأولانية و هي "Unknown" write . بكدة يبقى خلصنا شرح ال .code

### ▪ V.I. Note:

ال program دة ممكن نقول smart بنسبة 70% دورك انت بقى تعقد تغير في ترتيب ال facts بتاعة ال animal بحيث يبقى البرنامج Faster and efficient . يعني من الاخر ال cases ترتيبها مهم وال properties اللي في ال cases ترتيبها أهم و دي اللي بتفرق expert system الصح عن أي واحد تاني.

---

**Good Luck**