TP7 - Bases de données déductives

Paul Chaignon - Clément Gautrais

November 27, 2013

1 Bases de données déductives

1.1 Questions

Listing 1: baseauto.pro

```
/**
TP 7 Base de DonnÃ©es DÃ©ductives (BDD) - Prolog

@author Paul Chaignon
@author ClÃ©ment ClÃ©ment
@version Annee scolaire 2013/2014
*/


/*
===============================================================================
===============================================================================
 DÃ©finition des prÃ©dicats
===============================================================================
*/
% =============================================================================
% SECTION 1 : Base de donnÃ©es
% =============================================================================

assemblage(voiture, porte, 4).
assemblage(voiture, roue, 4).
assemblage(voiture, moteur, 1).
assemblage(roue, jante, 1).
assemblage(porte, tole, 1).
assemblage(porte, vitre, 1).
assemblage(roue, pneu, 1).
assemblage(moteur, piston, 4).
assemblage(moteur, soupape, 16).


piece(p1, tole, lyon).
piece(p2, jante, lyon).
piece(p3, jante, marseille).
piece(p4, pneu, clermontFerrand).
piece(p5, piston, toulouse).
piece(p6, soupape, lille).
```

```prolog
piece(p7, vitre, nancy).
piece(p8, tole, marseille).
piece(p9, vitre, marseille).


demandeFournisseur(dupont, lyon).
demandeFournisseur(michel, clermontFerrand).
demandeFournisseur(durand, lille).
demandeFournisseur(dupond, lille).
demandeFournisseur(martin, rennes).
demandeFournisseur(smith, paris).
demandeFournisseur(brown, marseille).


fournisseurReference(f1, dupont, lyon).
fournisseurReference(f2, durand, lille).
fournisseurReference(f3, martin, rennes).
fournisseurReference(f4, michel, clermontFerrand).
fournisseurReference(f5, smith, paris).
fournisseurReference(f6, brown, marseille).


livraison(f1, p1, 300).
livraison(f2, p2, 200).
livraison(f3, p3, 200).
livraison(f4, p4, 400).
livraison(f6, p5, 500).
livraison(f6, p6, 1000).
livraison(f6, p7, 300).
livraison(f1, p2, 300).
livraison(f4, p2, 300).
livraison(f4, p1, 300).


% =============================================================================
% SECTION 2 : Operation relationnelles
% =============================================================================

%fromcity(+City, -piece(NumPiece, Nom, City))
fromcity(City, piece(NumPiece, Nom, City)):-
  piece(NumPiece, Nom, City).

%infospieces(?Nom, ?Lieu)
infospieces(Nom, Lieu):-
  piece(_, Nom, Lieu).

%inter(?Nom, ?Lieu)
inter(Nom, Lieu):-
  demandeFournisseur(Nom, Lieu),
  fournisseurReference(_, Nom, Lieu).

%union(?Nom, ?Lieu)
union(Nom, Lieu):-
  demandeFournisseur(Nom, Lieu).
union(Nom, Lieu):-
  fournisseurReference(_, Nom, Lieu),
  not(demandeFournisseur(Nom, Lieu)).

%diff(?Nom, ?Lieu)
diff(Nom, Lieu):-
  demandeFournisseur(Nom, Lieu),
  not(fournisseurReference(_, Nom, Lieu)).

%prodcart(-uplet(NumFourn1, Nom, Ville, NumFourn2, Piece, Quantite))
prodcart(uplet(NumFourn1, Nom, Ville, NumFourn2, Piece, Quantite)):-
  fournisseurReference(NumFourn1, Nom, Ville),
  livraison(NumFourn2, Piece, Quantite).

%jointure(-uplet(NumFourn, Nom, Ville, Piece, Quantite))
jointure(uplet(NumFourn, Nom, Ville, Piece, Quantite)):-
  fournisseurReference(NumFourn, Nom, Ville),
  livraison(NumFourn, Piece, Quantite).

%jointureSelect(-uplet(NumFourn, Nom, Ville, Piece, Quantite))
jointureSelect(uplet(NumFourn, Nom, Ville, Piece, Quantite)):-
  jointure(uplet(NumFourn, Nom, Ville, Piece, Quantite)),
  Quantite > 350.

jointureSelect2(uplet(NumFourn, Nom, Ville, Piece, Quantite)):-
  livraison(NumFourn, Piece, Quantite),
  Quantite > 350,
  fournisseurReference(NumFourn, Nom, Ville).

pasOk(Fourn1):-
  piece(PieceLyon, _, lyon),
  not(livraison(Fourn1, PieceLyon, _)).
%div(-Fourn)
div(Fourn):-
  fournisseurReference(Fourn, _, _),
  not(pasOk(Fourn)).

sumQuantites([], 0).
sumQuantites([Prem|List], Total):-
  sumQuantites(List, Total2),
  Total is Total2 + Prem.

totalPieces(Fourn, NB):-
  findall(Quantite, livraison(Fourn, _, Quantite), Quantites),
  sumQuantites(Quantites, NB).

%totalPieces(-uplet(Fourn, NB))
totalPieces(uplet(Fourn, NB)):-
  fournisseurReference(Fourn, _, _),
  totalPieces(Fourn, NB).



% ============================================================================
% SECTION 3 : Au dela de l algebre relationnelle
% ============================================================================

%Q1
```

```
%composant(+Composant, -ComposeDe)
composant(Composant, ComposeDe):-
  assemblage(Composant, ComposeDe, _).

composant(Composant, ComposeDe):-
  assemblage(Composant, SousComposant, _),
  composant(SousComposant, ComposeDe).


% Q2

nbExemplaires(Composant, Piece, N):-
  assemblage(Composant, Piece, N).

nbExemplaires(Composant, Piece, N):-
  assemblage(Composant, ComposantInter, M),
  nbExemplaires(ComposantInter, Piece, P),
  N is M*P.

estPieceCut(P):-
  piece(_, P, _),
  !.

%piecesTotal(+Composant, -composant(R,N))
piecesTotal(Composant, composant(R,N)):-
  composant(Composant, R),
  estPieceCut(R),
  nbExemplaires(Composant, R, N).


%Q3
sumQuantProd([], 0).

sumQuantProd([Prod|R], NB):-
  nbComposantFourni(Prod, Nombre),
  sumQuantProd(R, NbR),
  NB is NbR+Nombre.

nbComposantFourniNom(Nom, NB):-
  findall(CodeComp, piece(CodeComp, Nom, _), Prods),
  sumQuantProd(Prods, NB).

nbComposantFourni(CodeComp, NB):-
  findall(Quantite, livraison(_, CodeComp, Quantite), Quantites),
  sumQuantites(Quantites, NB).

ratioComposant([], []).

ratioComposant([composant(NomComp,N)|Compo], [Rat1|Rest]):-
  nbComposantFourniNom(NomComp, T),
  Rat1 is T/N,
  ratioComposant(Compo, Rest).

minBis([], Min, Min).

minBis([P|R], Min, Res):-
  P < Min,
  !,
  minBis(R, P, Res).

minBis([P|R], Min, Res):-
  minBis(R, Min, Res).

minL([P|R], N):-
  minBis(R, P, N).

%nbVoit(-NB)
nbVoit(NB):-
  findall(composant(Comp,N), piecesTotal(voiture, composant(Comp, N)), Composants),
  ratioComposant(Composants, Ratios),
  minL(Ratios, NBfloat),
  NB is truncate(NBfloat).
```

1.2 Tests

Listing 2: baseauto_tests.pro

```
/*
===============================================================================
===============================================================================
 Tests
===============================================================================
*/

/*
Q2.1

fromcity(lyon, R).

R = piece(p1, tole, lyon)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = piece(p2, jante, lyon)
Yes (0.00s cpu, solution 2)
*/

/*
Q2.2
infospieces(N, L).

N = tole
L = lyon
Yes (0.00s cpu, solution 1, maybe more) ? ;

N = jante
L = lyon
Yes (0.00s cpu, solution 2, maybe more) ? ;

N = jante
L = marseille
Yes (0.00s cpu, solution 3, maybe more) ? ;
```

```

N = pneu
L = clermontFerrand
Yes (0.00s cpu, solution 4, maybe more) ?
...
*/

/*
Q2.3
inter(N, L).

N = dupont
L = lyon
Yes (0.00s cpu, solution 1, maybe more) ? ;

N = michel
L = clermontFerrand
Yes (0.00s cpu, solution 2, maybe more) ? ;

N = durand
L = lille
Yes (0.00s cpu, solution 3, maybe more) ? ;

N = martin
L = rennes
Yes (0.00s cpu, solution 4, maybe more) ? ;

N = smith
L = paris
Yes (0.00s cpu, solution 5, maybe more) ? ;

N = brown
L = marseille
Yes (0.00s cpu, solution 6)



union(N, L).

N = dupont
L = lyon
Yes (0.00s cpu, solution 1, maybe more) ? ;

N = michel
L = clermontFerrand
Yes (0.00s cpu, solution 2, maybe more) ? ;

N = durand
L = lille
Yes (0.00s cpu, solution 3, maybe more) ? ;

N = dupond
L = lille
Yes (0.00s cpu, solution 4, maybe more) ? ;

N = martin
L = rennes
Yes (0.00s cpu, solution 5, maybe more) ? ;
```

```

N = smith
L = paris
Yes (0.00s cpu, solution 6, maybe more) ? ;

N = brown
L = marseille
Yes (0.00s cpu, solution 7, maybe more) ? ;

No (0.00s cpu)


diff(N, L).

N = dupond
L = lille
Yes (0.00s cpu, solution 1, maybe more) ? ;

No (0.00s cpu)
*/

/*
Q2.4

prodcart(R).

R = uplet(f1, dupont, lyon, f1, p1, 300)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = uplet(f1, dupont, lyon, f2, p2, 200)
Yes (0.01s cpu, solution 2, maybe more) ? ;

R = uplet(f1, dupont, lyon, f3, p3, 200)
Yes (0.01s cpu, solution 3, maybe more) ? ;

R = uplet(f1, dupont, lyon, f4, p4, 400)
Yes (0.01s cpu, solution 4, maybe more) ? ;

R = uplet(f1, dupont, lyon, f6, p5, 500)
Yes (0.01s cpu, solution 5, maybe more) ? ;

R = uplet(f1, dupont, lyon, f6, p6, 1000)
Yes (0.01s cpu, solution 6, maybe more) ? ;

R = uplet(f1, dupont, lyon, f6, p7, 300)
Yes (0.01s cpu, solution 7, maybe more) ? ;
... (60 solutions)

*/

/*
Q2.5
jointure(R).

R = uplet(f1, dupont, lyon, p1, 300)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = uplet(f1, dupont, lyon, p2, 300)
```

```
Yes (0.00s cpu, solution 2, maybe more) ? ;

R = uplet(f2, durand, lille, p2, 200)
Yes (0.00s cpu, solution 3, maybe more) ? ;

R = uplet(f3, martin, rennes, p3, 200)
Yes (0.01s cpu, solution 4, maybe more) ? ;

R = uplet(f4, michel, clermontFerrand, p4, 400)
Yes (0.01s cpu, solution 5, maybe more) ? ;

R = uplet(f4, michel, clermontFerrand, p2, 300)
Yes (0.01s cpu, solution 6, maybe more) ? ;

R = uplet(f4, michel, clermontFerrand, p1, 300)
Yes (0.01s cpu, solution 7, maybe more) ? ;

R = uplet(f6, brown, marseille, p5, 500)
Yes (0.01s cpu, solution 8, maybe more) ? ;

R = uplet(f6, brown, marseille, p6, 1000)
Yes (0.01s cpu, solution 9, maybe more) ? ;

R = uplet(f6, brown, marseille, p7, 300)
Yes (0.01s cpu, solution 10)



jointureSelect(R).

R = uplet(f4, michel, clermontFerrand, p4, 400)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = uplet(f6, brown, marseille, p5, 500)
Yes (0.00s cpu, solution 2, maybe more) ? ;

R = uplet(f6, brown, marseille, p6, 1000)
Yes (0.00s cpu, solution 3, maybe more) ? ;

No (0.00s cpu)
*/

/*
Q2.6
div(F).

F = f1
Yes (0.00s cpu, solution 1, maybe more) ? ;

F = f4
Yes (0.00s cpu, solution 2, maybe more) ? ;

No (0.00s cpu)
*/

/*
Q2.7
totalPieces(R).
```

```

R = uplet(f1, 600)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = uplet(f2, 200)
Yes (0.00s cpu, solution 2, maybe more) ? ;

R = uplet(f3, 200)
Yes (0.00s cpu, solution 3, maybe more) ? ;

R = uplet(f4, 1000)
Yes (0.00s cpu, solution 4, maybe more) ? ;

R = uplet(f5, 0)
Yes (0.00s cpu, solution 5, maybe more) ? ;

R = uplet(f6, 1800)
Yes (0.00s cpu, solution 6)
*/

/*
Q3.1
composant(voiture, R).

R = porte
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = roue
Yes (0.00s cpu, solution 2, maybe more) ? ;

R = moteur
Yes (0.00s cpu, solution 3, maybe more) ? ;

R = tole
Yes (0.00s cpu, solution 4, maybe more) ? ;

R = vitre
Yes (0.00s cpu, solution 5, maybe more) ? ;

R = jante
Yes (0.00s cpu, solution 6, maybe more) ? ;

R = pneu
Yes (0.00s cpu, solution 7, maybe more) ? ;

R = piston
Yes (0.00s cpu, solution 8, maybe more) ? ;

R = soupape
Yes (0.00s cpu, solution 9, maybe more) ? ;

No (0.00s cpu)
*/

/*
Q3.2
piecesTotal(voiture, R).

```

```
R = composant(tole, 4)
Yes (0.00s cpu, solution 1, maybe more) ? ;

R = composant(vitre, 4)
Yes (0.00s cpu, solution 2, maybe more) ? ;

R = composant(jante, 4)
Yes (0.00s cpu, solution 3, maybe more) ? ;

R = composant(pneu, 4)
Yes (0.00s cpu, solution 4, maybe more) ? ;

R = composant(piston, 4)
Yes (0.00s cpu, solution 5, maybe more) ? ;

R = composant(soupape, 16)
Yes (0.00s cpu, solution 6, maybe more) ? ;

No (0.00s cpu)
*/

/*
Q3.3
nbVoit(NB).

NB = 62.0
Yes (0.00s cpu)
*/
```