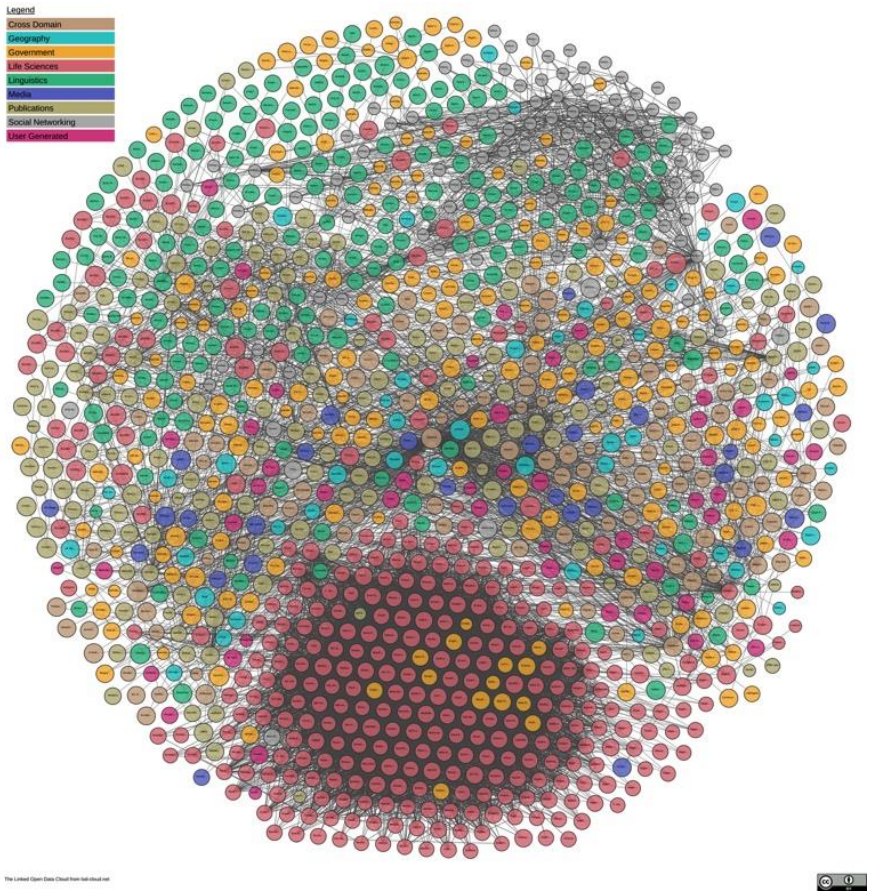
Module MLBDA
Master Informatique
Spécialité DAC

COURS 9 – SPARQL

Linked Open Data Cloud



SPARQL

SPARQL : Simple Protocol and RDF Query Language

Langage de requêtes du W3C pour RDF/RDFS

- SPARQL 1.0 : recommandation 2008
- SPARQL 1.1 : recommandation 2013

Principe : pattern matching sur des graphes

Requêtes de base

PREFIX : permet de déclarer les espaces de noms.

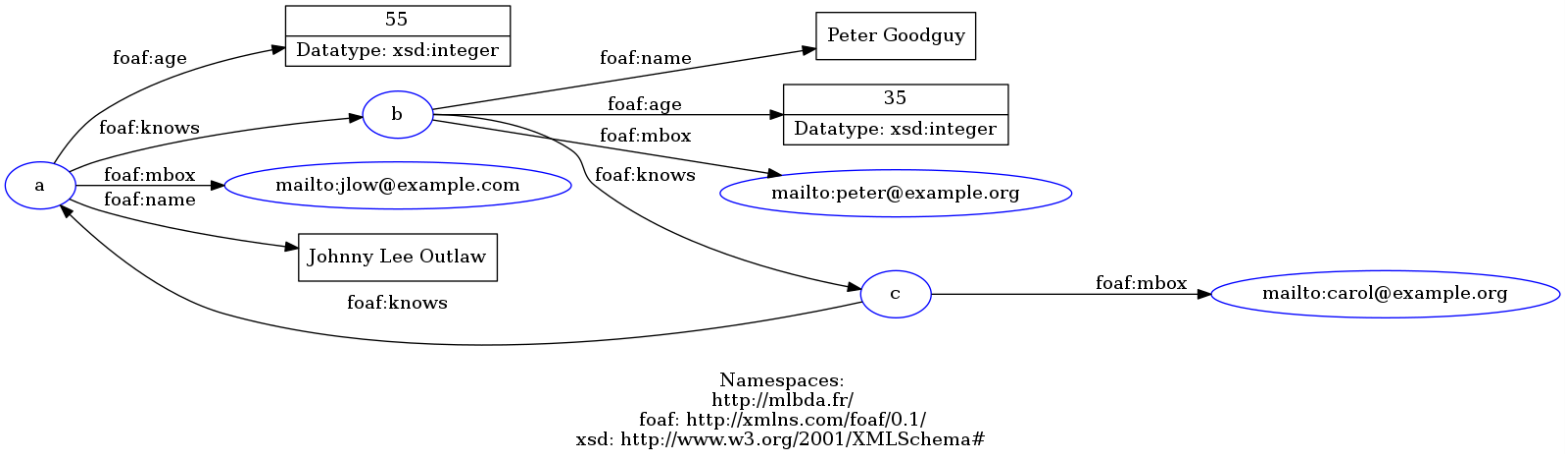
SELECT : **variables** affichées dans le résultat.

FROM (optionnel) : graph interrogé (graphe par défaut)

WHERE : **motif de graphe avec des variables**.

Exemple

```
@prefix : <http://mlbda.fr/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd:
<http://www.w3.org/2001/XMLSchema#> .
:a foaf:age 55 ;
   foaf:knows :b ;
   foaf:mbox <mailto:jlow@example.com> ;
   foaf:name "Johnny Lee Outlaw" .
:b foaf:age 35 ;
   foaf:knows :c ;
   foaf:mbox <mailto:peter@example.org> ;
   foaf:name "Peter Goodguy" .
:c foaf:knows :a ;
   foaf:mbox <mailto:carol@example.org> .
```



Exemple

Requête (motif de graphe) :

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name ?mbox

WHERE { ?x foaf:name ?name .

?x foaf:mbox ?mbox . }

Réponse :

name	mbox
Johnny Lee Outlaw	<mailto:jlow@example.com>
Peter Goodguy	<mailto:peter@example.org>

Turtle + variables → motif de triplet

L'ensemble des termes RDF est défini par T = U ∪ L ∪ B ∪ V où

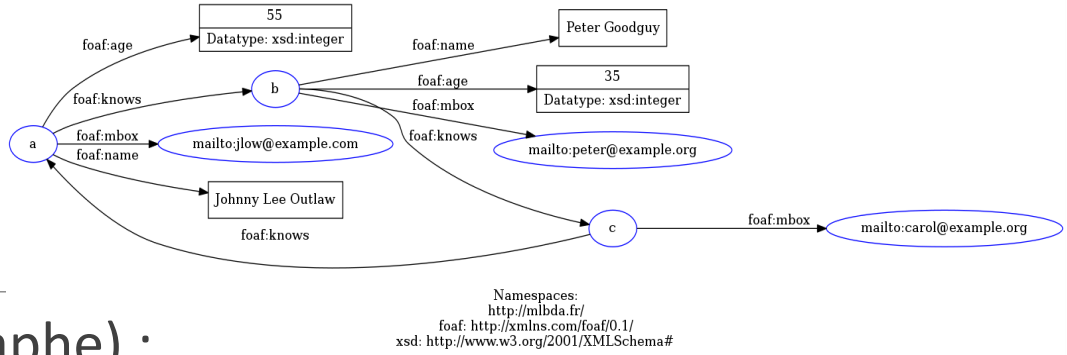
- U : ensemble des URI
  - <http://www.w3.org/TR/rdf-syntax-grammar>
  - dc:title
- L : litéraux RDF (valeurs) : "valeur"@motcle^^type
  - @motcle : langue, monnaie, encodage, … (optionnel)
  - ^^type : type XML Schema (optionnel)
  - "RDF1.1 XML Syntax"@en, "Dave Beckett", "false"^^xsd:boolean
- B : nœuds blancs
- V : ensembles de noms de variables (?nom ou $nom) :
  - ?x, ?nom, ?y, $nom, $a

Un motif de triplet RDF est un élément de l'ensemble :

$(U \cup B \cup V) \times (U \cup V) \times (U \cup L \cup B \cup V)$

Un graphe RDF est un ensemble de triplets RDF.

Exemple



Requête (motif de graphe) :

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE { ?x foaf:name ?name ;
          foaf:age "35"^^xsd:integer .
      }
```

Motif Turtle avec variables

name
Peter Goodguy

Exemples de motifs triplets

Motif ∈ U × U × L

- :a foaf:name   "Johnny Lee Outlaw" .
- :a foaf:know :b .

Motif ∈ V × U × V

- ?p foaf:age ?a .

Motif ∈ B × U × L

- [] foaf:name "Bob" .

Motifs de triplets

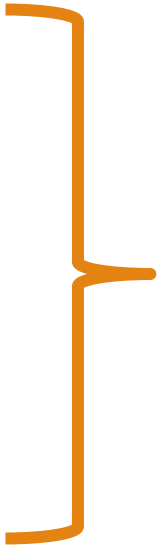
Pour simplifier l'écriture des motifs de triplets, on peut factoriser (comme dans Turtle) :

- factoriser les sujets:
  - ?x foaf:name ?name; foaf:mbox ?mbox .
- factoriser les sujets et les propriétés :
  - ?x foaf:nick "Robert", "Bob" .
- utiliser les collections RDF
  - ?x mlda:enfant (:bob :alice) .
  - ?x a :Personne . équivaut à ?x rdf:type :Personne .

Exemple : entiers versus chaînes de caractères

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE { ?x foaf:name ?name ;
          foaf:age "35"^^xsd:integer .  }
```

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE { ?x foaf:name ?name ;
          foaf:age 35 .  }
```



Réponse non vide

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE { ?x foaf:name ?name ;
          foaf:age "35" .  }
```

Réponse vide

Motifs de graphes

Définition de motifs de graphs (GP):

Un ensemble de motifs de triplets est un motif de graphe (élémentaire)

Si GP est un motif de graphe, { GP } est un motif de graphe (de groupe).

Si GP et GP' sont des motifs de graphe :

- GP FILTER (test) GP' : sélection/filtrage
- GP OPTIONAL GP' : graphes optionnels
- GP UNION GP’ : union de graphes

sont des motifs de graphe.

Exemples de motifs de graphe

@prefix foaf: <http://xmlns.com/foaf/0.1/> .

Motif élémentaire :

```
?a foaf:age 26 . ?a foaf:mbox ?m .
```

Motif de groupe :

```
{ ?a foaf:age 26 . ?a foaf:mbox ?m . }
```

Motif de groupe avec FILTRE :

```
{?a foaf:age ?age . FILTER (?age < 26) ?a foaf:mbox ?m . }
```

Motif de groupe avec OPTIONAL :

```
{?a foaf:age 26 . OPTIONAL {?a foaf:mbox ?m . }}
```

Sémantique : solutions de motif

Une **solution de motif** (de graphe ou de triplet) S est une fonction de *substitution* d'un ensemble de variables V vers l'ensemble des termes $U \cup B \cup L$:

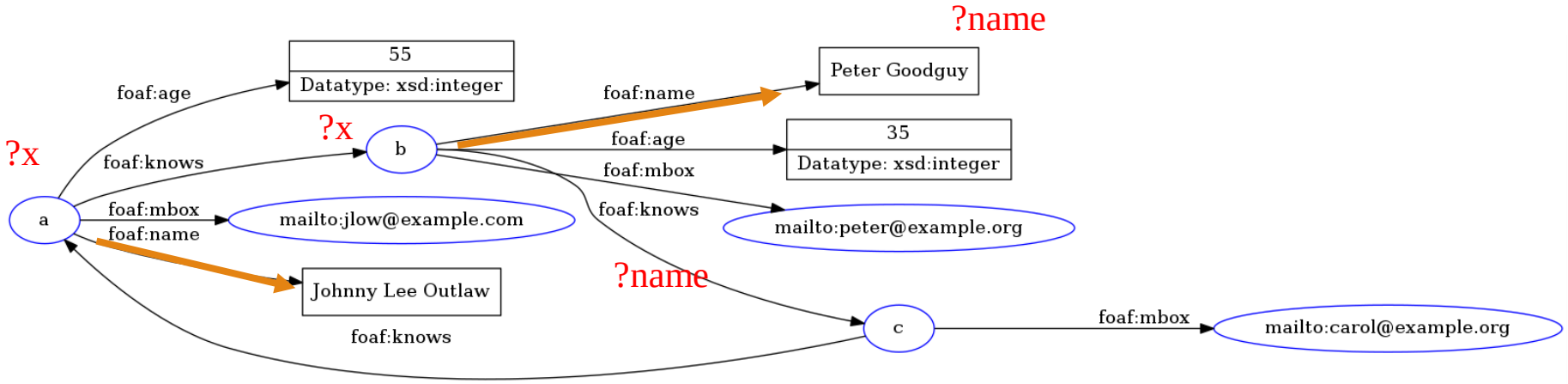
$$S : V \rightarrow U \cup B \cup L$$

Une solution de motif S est une **solution pour un motif M dans un graphe G** si S appliqué au motif, S(M) est un sous-graphe de G.

Le résultat d'une requête est construit à partir de l'ensemble de toutes ses solutions de motifs.

Sémantique : substitution de variables

Motif M : { ?x foaf:name ?name . }



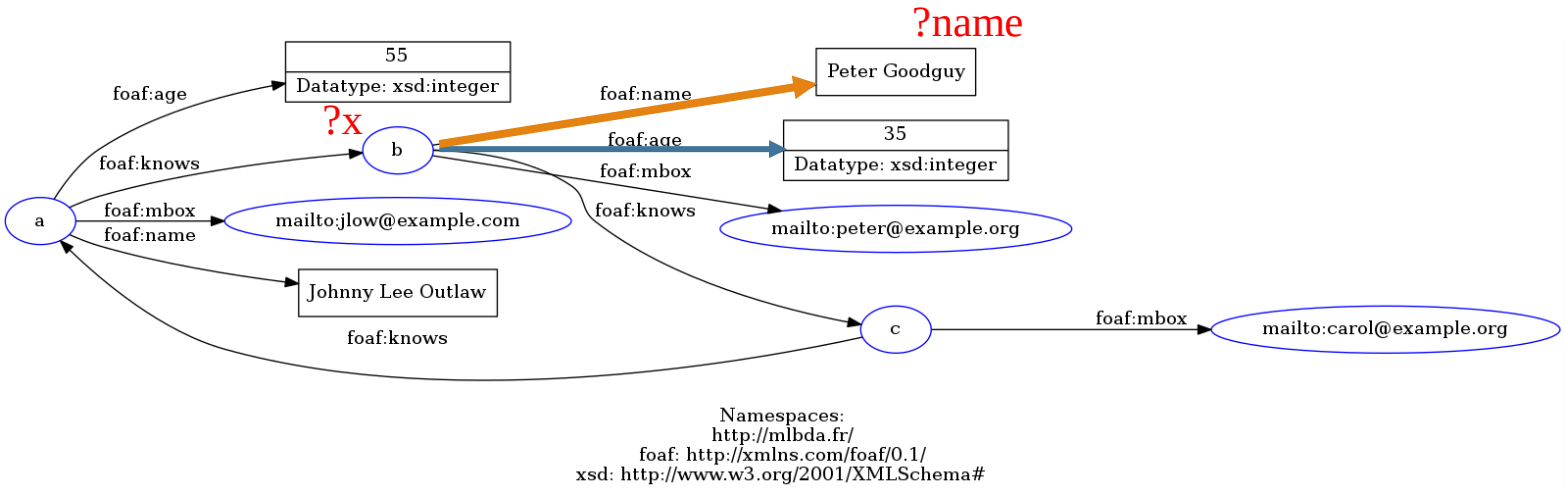
Deux solutions du motif M :

Namespaces:
http://mlbda.fr/
foaf: http://xmlns.com/foaf/0.1/
xsd: http://www.w3.org/2001/XMLSchema#

?x	?name
a	Johnny Lee Outlaw
b	Peter Goodguy

Sémantique : substitution de variables

Motif M: { ?x foaf:name ?name . ?x foaf:age "35"^^xsd:integer . }



Une solution du motif M:

?x	?name
b	Peter Goodguy

Filtres

Le mot-clef FILTER permet de restreindre les solutions sur tout le groupe où le filtre apparaît. La position du filtre dans le groupe n'a pas d'importance.

Les motifs suivants sont équivalents :

```
{ ?x foaf:name ?name . ?x foaf:age ?a .  FILTER (?a < 40) }
```

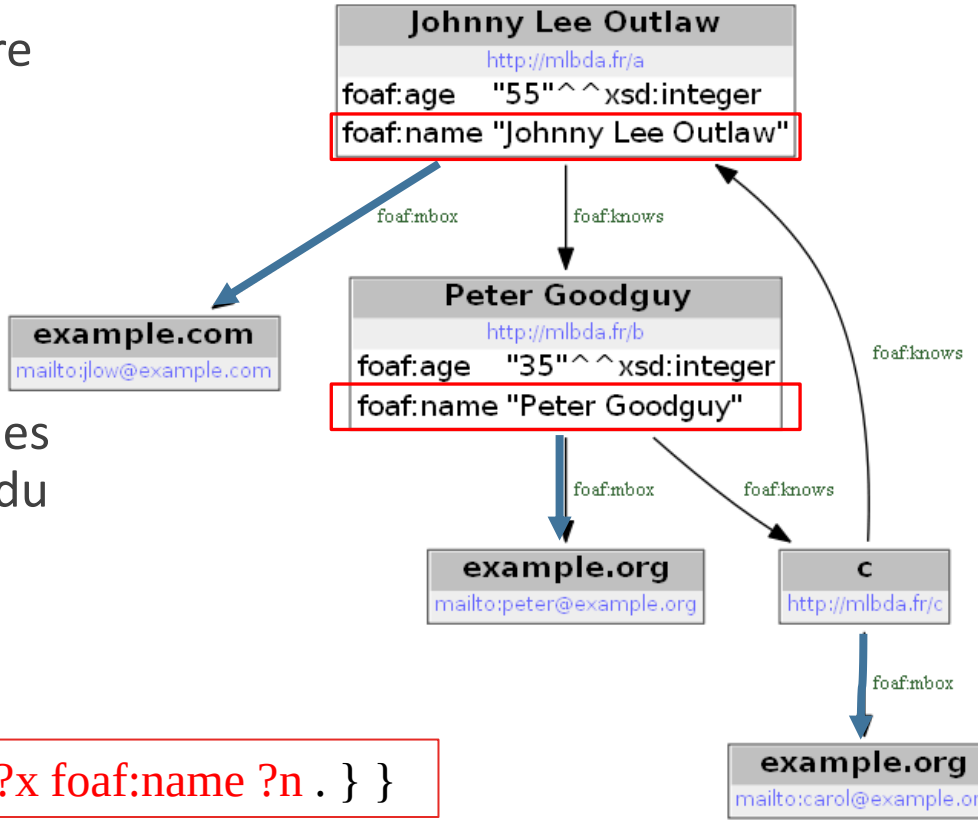
```
{ ?x foaf:name ?name . FILTER (?a < 40)  . ?x foaf:age ?a }
```

```
{ FILTER (?a < 40) ?x foaf:name ?name . ?x foaf:age ?a }
```

Motif Optionnel

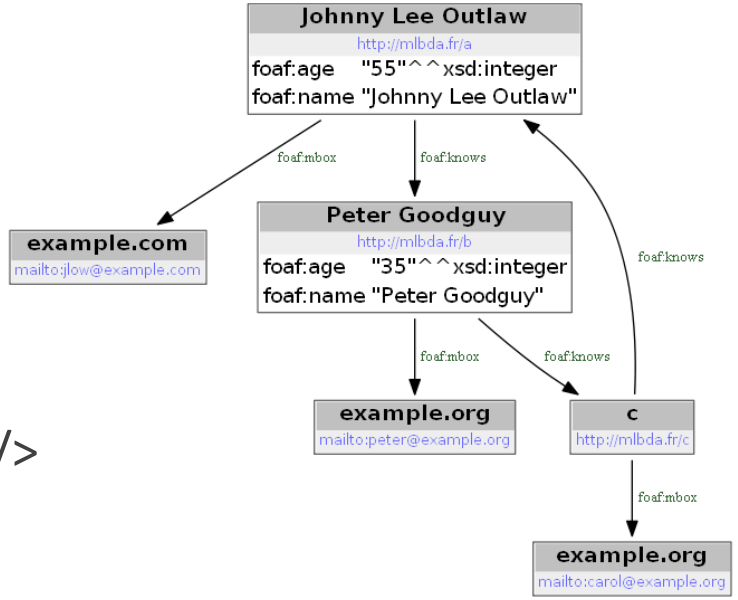
Un motif de graphe élémentaire permet de rechercher des solutions qui correspondent entièrement au motif d'interrogation.

Le filtrage optionnel (mot-clé OPTIONAL) permet d'obtenir des solutions même si des parties du motif d'interrogation ne correspondent pas.




```
{ ?x foaf:mbox ?m . OPTIONAL { ?x foaf:name ?n . } }
```

Exemple



PREFIX : <http://mlbda.fr/>

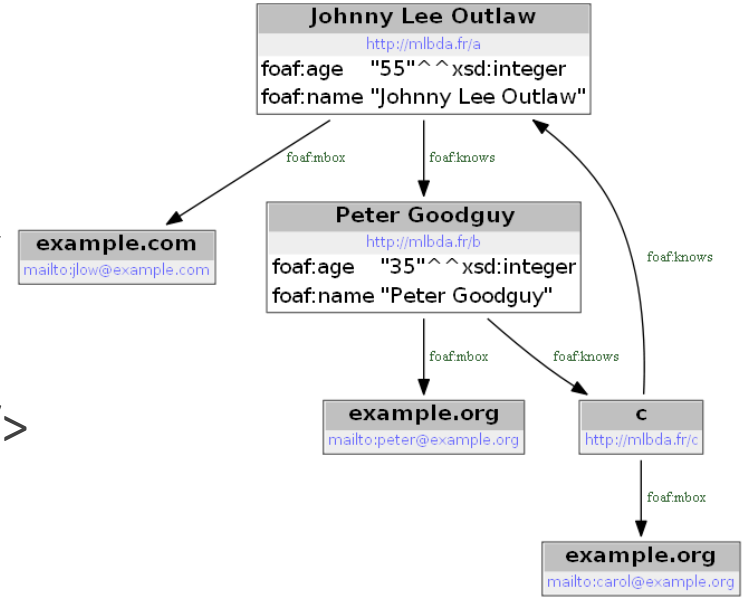
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?x ?n ?m

WHERE { ?x foaf:mbox ?m . OPTIONAL { ?x foaf:name ?n . } }

```
 m                              |          n          |          x
-------------------------------------------------------------------------
<mailto:carol@example.org>|-                      |<http://mlbda.fr/c>
<mailto:jlow@example.com> |"Johnny Lee Outlaw"|<http://mlbda.fr/a>
<mailto:peter@example.org>|"Peter Goodguy"    |<http://mlbda.fr/b>
```

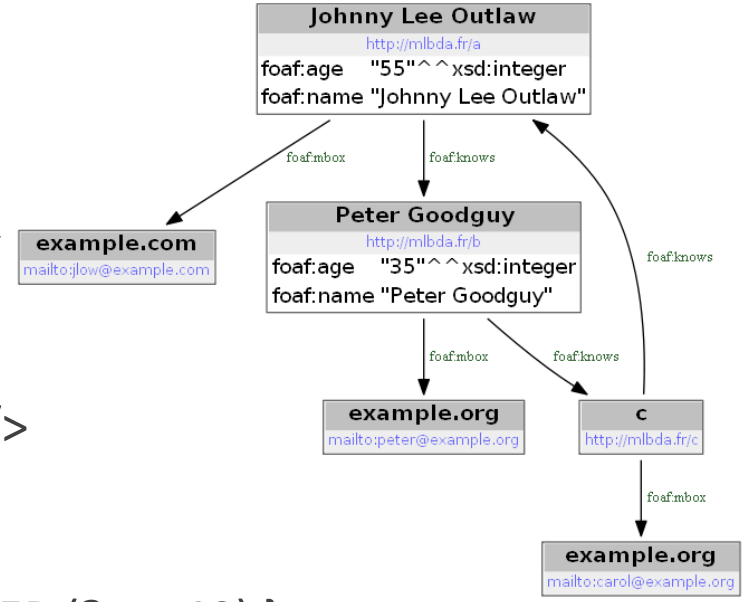
Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?n ?m ?a
WHERE { ?x foaf:mbox ?m .
        OPTIONAL { ?x foaf:name ?n . }
        OPTIONAL { ?x foaf:age ?a . }
}
```

```
 a                  |            m                |          n          | x
-------------------------------------------------------------------------
"35"^^xsd:integer  |<mailto:peter@example.org>|"Peter Goodguy"     |:b
"55"^^xsd:integer  |<mailto:jlow@example.com> |"Johnny Lee Outlaw"|:a
-                  |<mailto:carol@example.org>|-                   |:c
```

Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?m ?a
WHERE { ?x foaf:mbox ?m .
        OPTIONAL { ?x foaf:age ?a . FILTER (?a < 40) }
}
```

```
 a                 |             m                | x
-------------------------------------------------------
"35"^^xsd:integer |<mailto:peter@example.org>|:b
-                 |<mailto:jlow@example.com> |:a
-                 |<mailto:carol@example.org>|:c
```

UNION

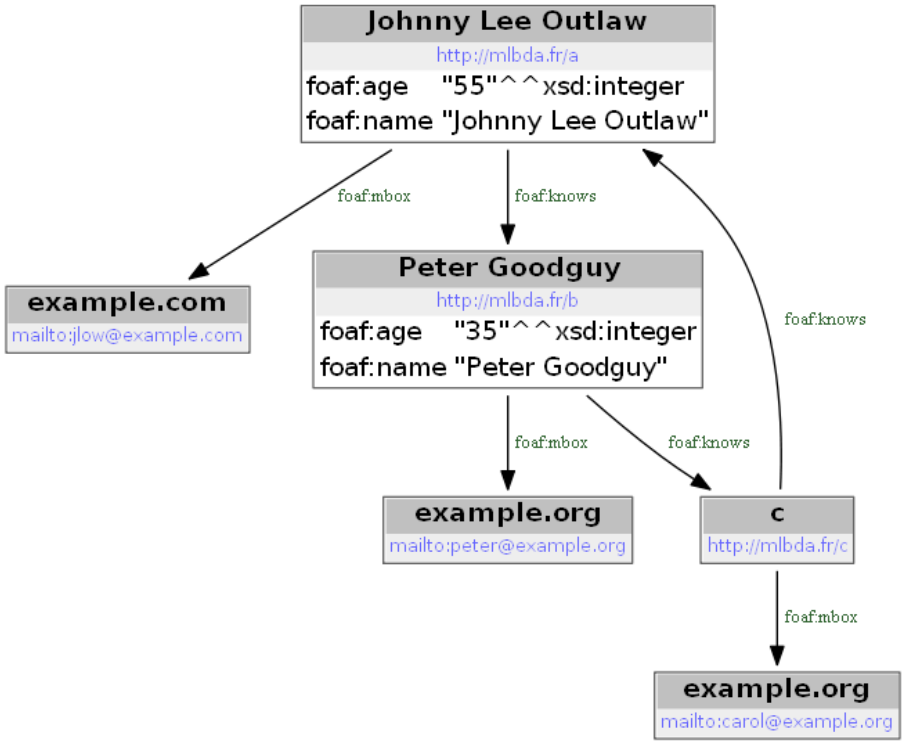
Le mot-clef UNION permet d'indiquer des alternatives de motifs (un motif OU un autre peut correspondre).

Si plusieurs alternatives correspondent, toutes les solutions sont trouvées.

```
{   { ?x foaf:mbox ?m .
      ?x foaf:name ?n . }
      UNION
    { ?x foaf:mbox ?m .
      ?x foaf:age ?a . }
}
```

Exemple

```
SELECT ?x ?n ?m ?a
WHERE { { ?x foaf:mbox ?m .
          ?x foaf:name ?n . }
        UNION
        { ?x foaf:mbox ?m .
          ?x foaf:age ?a . }
```




```
 a |      m                        |    n                |  x
-------------------------------------------------------------------------
35 | <mailto:peter@example.org>|-                     |<http://mlbda.fr/b>
55 | <mailto:jlow@example.com> |-                     |<http://mlbda.fr/a>
-  | <mailto:jlow@example.com> |"Johnny Lee Outlaw"|<http://mlbda.fr/a>
-  | <mailto:peter@example.org>|"Peter Goodguy"    |<http://mlbda.fr/b>
```

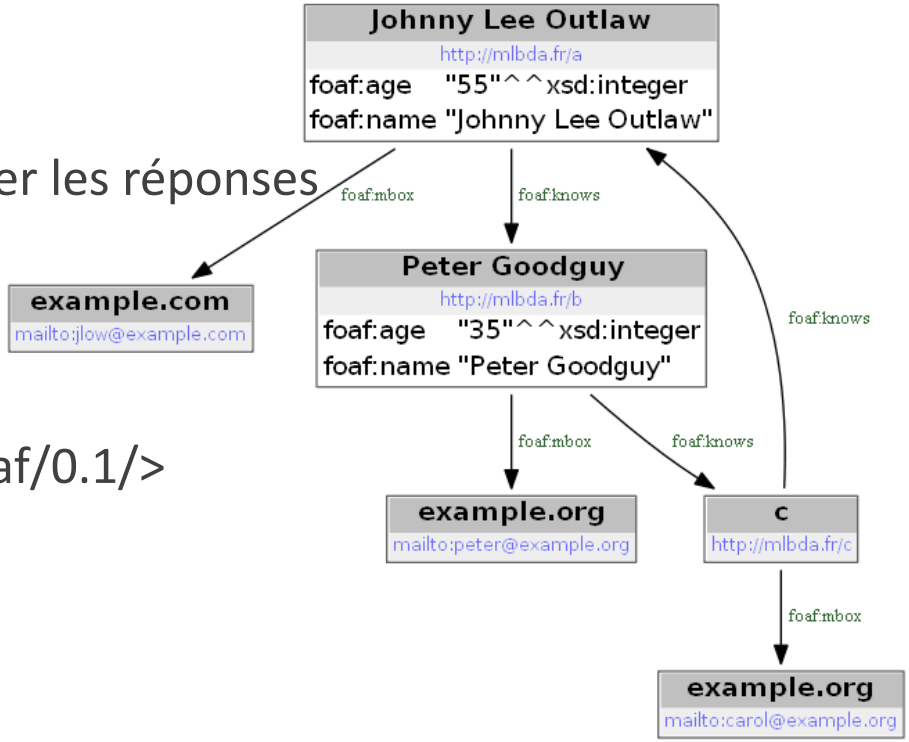
Négation

2 façons d'exprimer la négation

- FILTER NOT EXIST { …} teste la non-existence d'un motif
- MINUS permet de retirer des solutions provenant d'un autre motif de graphe

MINUS

Le mot-clef MINUS permet d'enlever les réponses qui satisfont un motif:

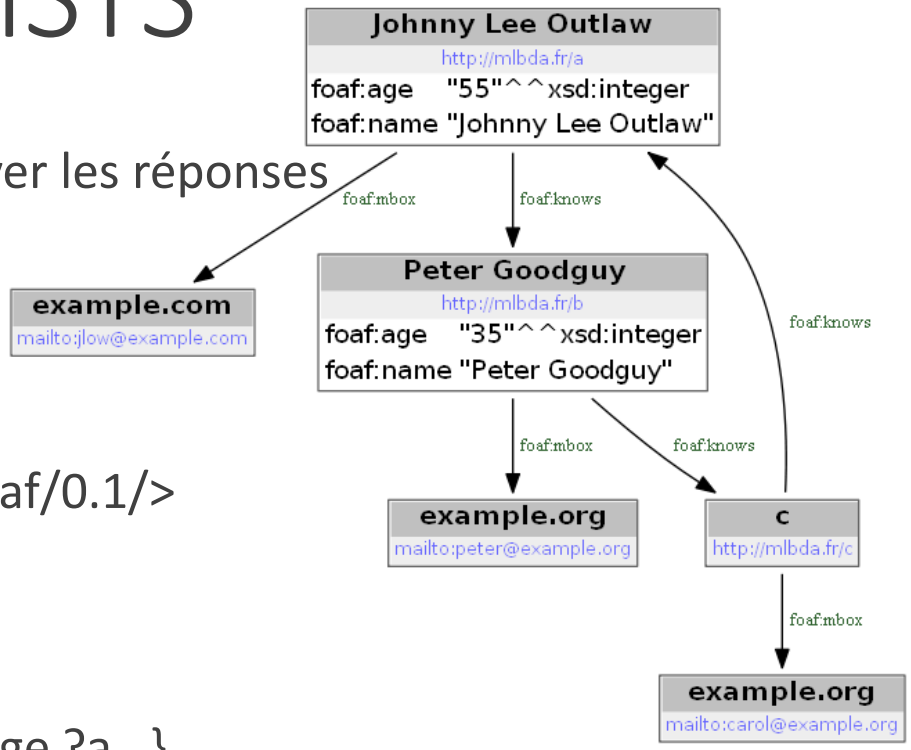



```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x
WHERE {
      { ?x foaf:mbox ?m . }
      MINUS
      { ?x foaf:age ?a . }
}
```

```
   x
--------------------
<http://mlbda.fr/c>
```

FILTER NOT EXISTS

Le mot-clef MINUS permet d'enlever les réponses qui satisfont un motif:




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x
WHERE {
      { ?x foaf:mbox ?m . }
      FILTER NOT EXISTS   { ?x foaf:age ?a . }
}
```

```
   x
--------------------
<http://mlbda.fr/c>
```

FILTER NOT EXISTS vs MINUS

SELECT * WHERE { ?s ?p ?o FILTER NOT EXISTS { ?x ?y ?z } }

La réponse est vide : *le motif* ?x ?y ?z *existe.*

SELECT * WHERE { ?s ?p ?o MINUS { ?x ?y ?z } }

La reponse contient tout le graphe (triplets): *il n'y a pas de variable commune et MINUS ne retire aucune solution.*

Chemins de propriétés

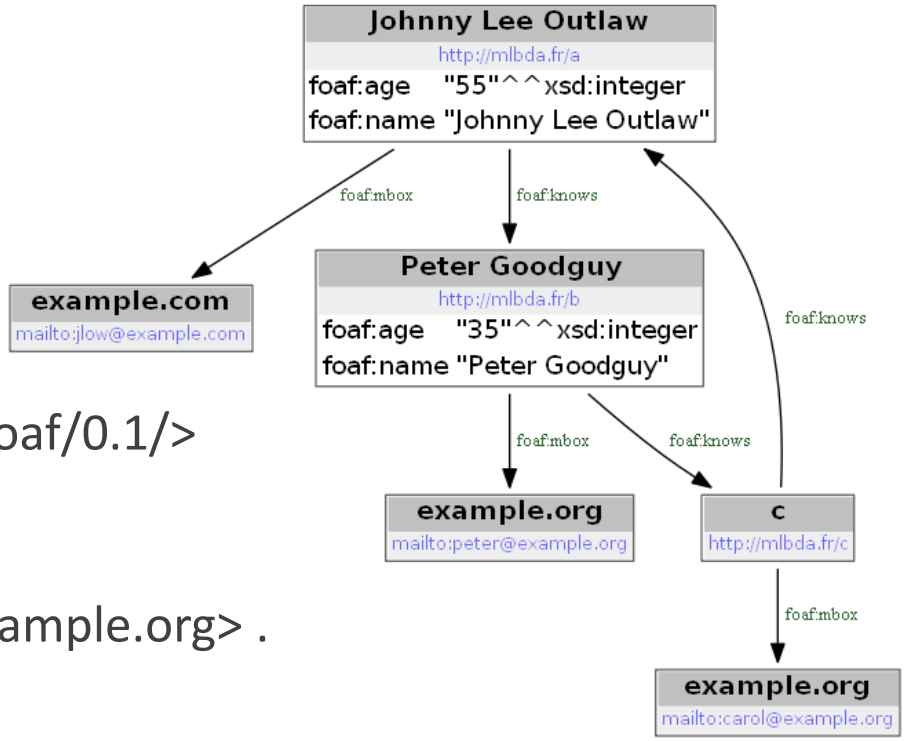
Un chemin de propriété est une expression régulière sur les noms de propriétés.

Les chemins de propriété permettent de trouver des ressources reliées par des chemin de longueur arbitraire.

Opérateurs :

- Séquence : /
- Alternative : |
- Répétition :
  - * (0 ou plusieurs occurrences)
  - + (1 ou plusieurs occurrences)
- Option : ?
- Arc inverse : ^
- Négation : !

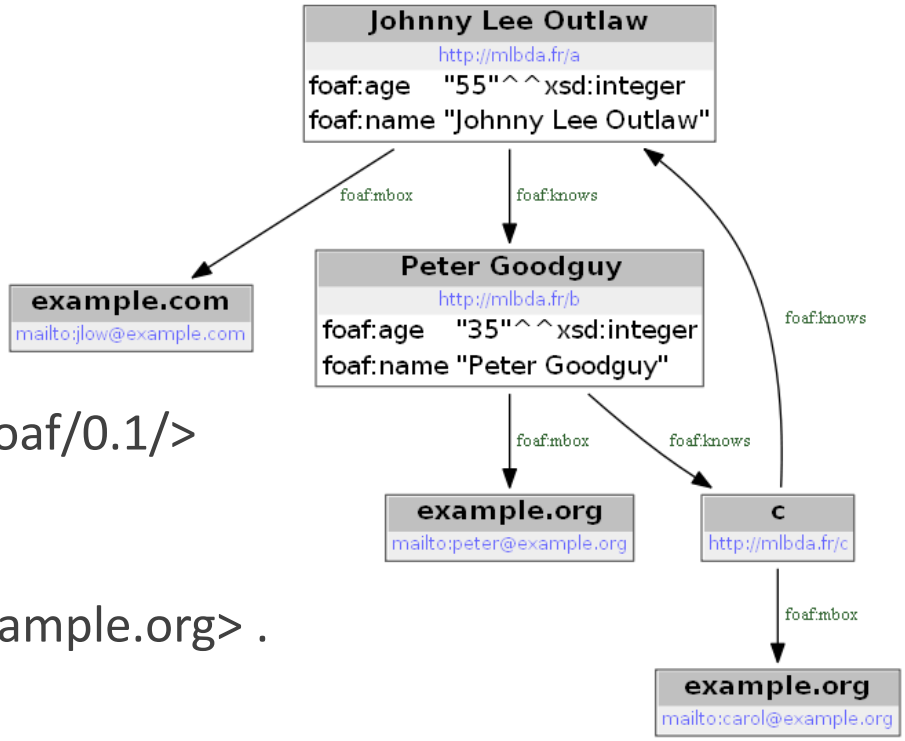
Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?n
WHERE {
   ?x foaf:mbox <mailto:carol@example.org> .
   ?x foaf:knows+/foaf:name ?n .
 }
```

```
  n                    |          x
---------------------------------------
"Johnny Lee Outlaw"|<http://mlbda.fr/c>
"Peter Goodguy"    |<http://mlbda.fr/c>
```

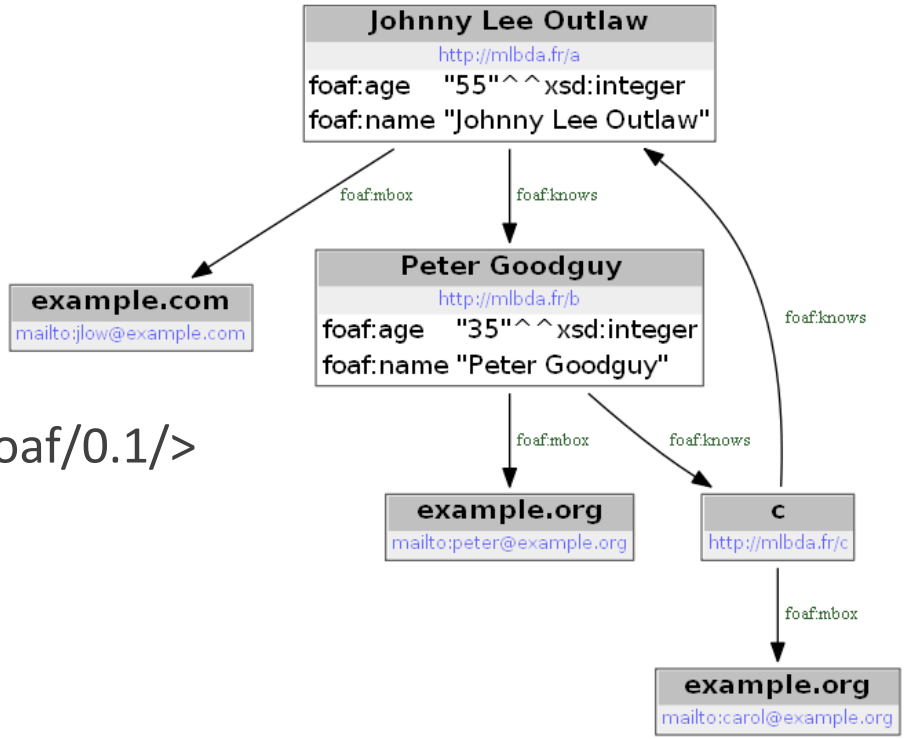
Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?y
WHERE {
   ?x foaf:mbox <mailto:carol@example.org> .
   ?x (^foaf:knows) ?y .
 }
```

```
 x                    |          y
--------------------------------------
<http://mlbda.fr/c>|<http://mlbda.fr/b>
```

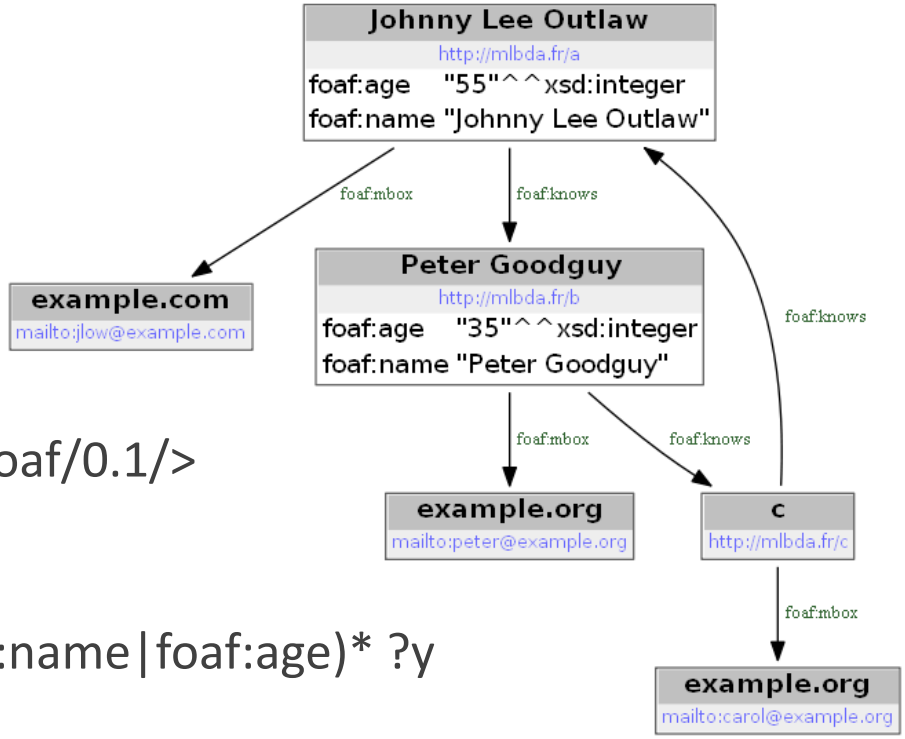
Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?y
WHERE {
   ?x (foaf:mbox|foaf:name) ?y
 }
```

```
   x                    |             y
-------------------------------------------------
<http://mlbda.fr/a>|"Johnny Lee Outlaw"
<http://mlbda.fr/a>|<mailto:jlow@example.com>
<http://mlbda.fr/b>|"Peter Goodguy"
<http://mlbda.fr/b>|<mailto:peter@example.org>
<http://mlbda.fr/c>|<mailto:carol@example.org>
```

Exemple




```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x ?y
WHERE {
   ?x (foaf:mbox|foaf:knows|foaf:name|foaf:age)* ?y
 }
```

Toutes les paires de nœuds ?x et ?y connectées par un chemin.

Requête SPARQL

Une requête SPARQL est de la forme

<FORMAT>

FROM <source>

WHERE { <motif> }

<TRANSFORM>

- <FORMAT> définit le format du résultat : **SELECT, CONSTRUCT, DESCRIBE, ASK**
- **FROM** définit la <source> RDF (optionnel si une source par défaut a été définie)
- **WHERE** définit le <motif> est un motif de graphe
- <TRANSFORM> est un transformateur : **ORDER BY, LIMIT, OFFSET, GROUP BY, HAVING**

Formats

format	**résultat**
SELECT, SELECT DISTINCT	table de données
CONSTRUCT	graphe RDF
ASK	valeur Booléenne (match non-vide)
DESCRIBE	description des ressources trouvées

Clause FROM

La clause FROM est utilisée pour indiquer l'URI d'un graphe sur lequel effectuer la requête.

En l'absence de clause FROM, l'interrogation s'effectue sur le graphe par défaut.

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name

FROM <http://example.org/foaf/aliceFoaf>

WHERE { ?x foaf:name ?name }

Séquences de solutions

Les motifs de solution génèrent une séquence non ordonnée de solutions, chacun étant une fonction partielle de variables présentes dans les motifs vers des termes RDF.

Ces solutions sont ensuite traitées comme une séquence sur laquelle on peut appliquer des opérateurs (transformations)

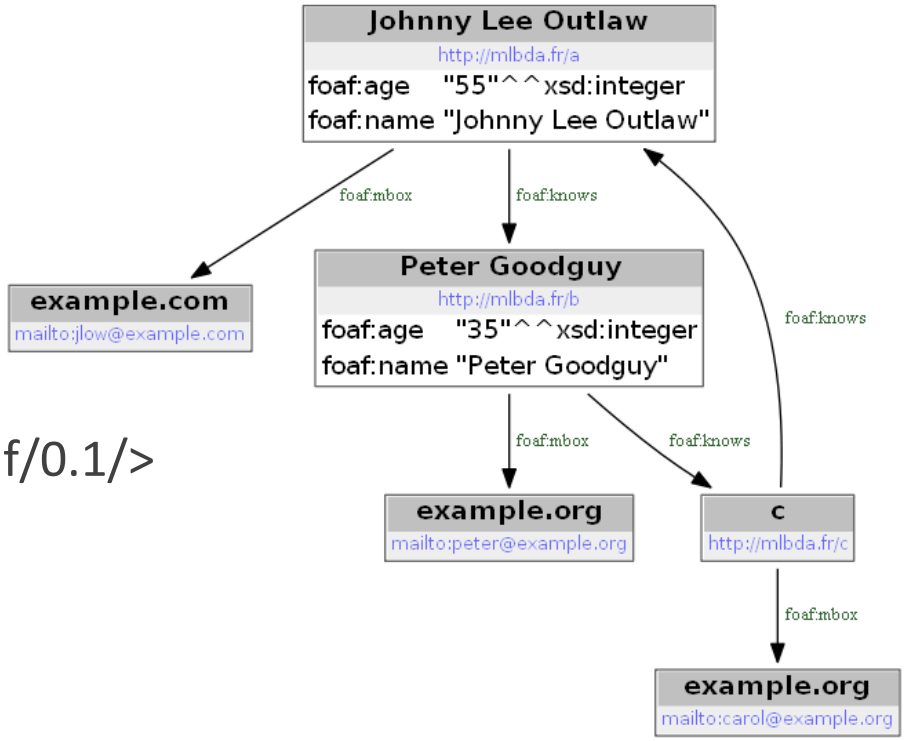
Dans la clause SELECT:

- DISTINCT ou REDUCED : éliminer les doublons (forcé ou optionnel)

Après la clause WHERE:

- ODER BY : permet de trier les solutions
- OFFSET : indique la position où commencer dans la séquence de solutions
- LIMIT : restreint le nombre de solutions

ORDER BY




```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name

WHERE { ?x foaf:name ?name }

ORDER BY ?name
```

```
 name
--------------------
"Johnny Lee Outlaw"
"Peter Goodguy"
```

DISTINCT

Données :

@prefix : <http://mlbda.fr/> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .

:a foaf:name "Alice " ; foaf:mbox <mailto:alice@work.example> .

:x foaf:name "Alice " ; foaf:mbox <mailto:smith@work.example> .

Requête :

PREFIX : <http://mlbda.fr/>

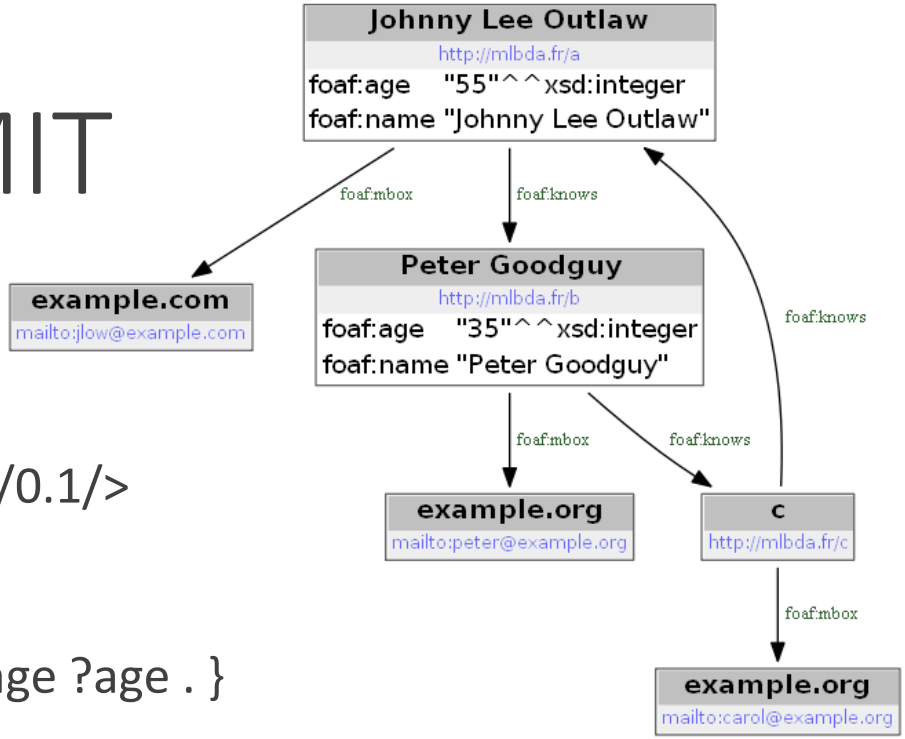
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT DISTINCT ?name

WHERE { ?x foaf:name ?name }

```
name
--------
Alice
```

ORDER BY + LIMIT




```
PREFIX : <http://mlbda.fr/>

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name ?age

WHERE { ?x foaf:name ?name; foaf:age ?age . }

ORDER BY DESC(?age)

LIMIT 1
```

```
 age    |        name
-------------------------------
55      | Johnny Lee Outlaw"
```

ORDER, OFFSET et LIMIT

OFFSET n : commencer à la solution n+1. OFFSET 0 n'a pas d'effet.

LIMIT n : limite à n le nombre de solutions

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE   { ?x foaf:name ?name }
ORDER BY ?name
LIMIT   5
OFFSET  10
```

Le résultat de cette requête aura au maximum 5 solutions, à partir de la 11ème dans la séquence de solutions.

Fonctions

Logique : !, &&, ||

Math : +, -, *, /

Comparaison : =, !=, >, <, …

Tests (SPARQL) : isURI, isBlank, isLiteral, bound

Autres (SPARQL) : str, lang, datatype, sameTerm, langMatches, regex

Quantification existentielle

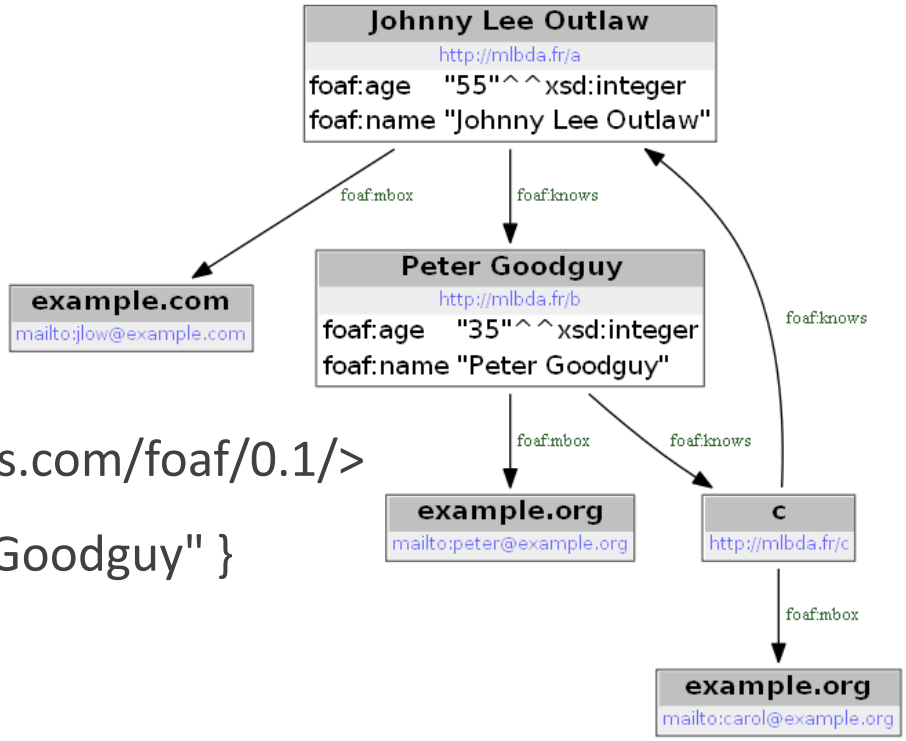
OPTIONAL + bound() permet d'exprimer la quantification existentielle

Bound() renvoie true si la variable est liée, false sinon.

```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x
WHERE {
    { ?x foaf:mbox ?m . }
    OPTIONAL { ?x foaf:age ?a . }
    FILTER (!BOUND(?a))
}
```

```
PREFIX : <http://mlbda.fr/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?x
WHERE {
    { ?x foaf:mbox ?m . }
    MINUS
    { ?x foaf:age ?a . }
```

ASK



Requêtes :

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
ASK  { ?x foaf:name "Peter Goodguy" }
```

renvoie true

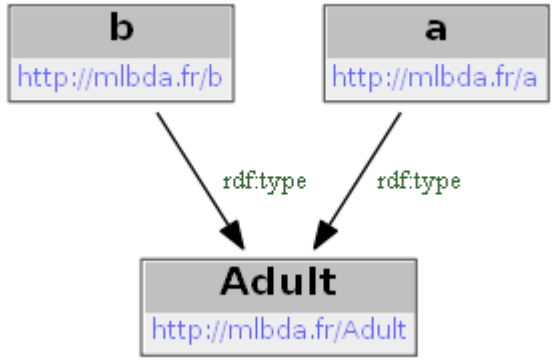
```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
ASK  { ?x foaf:name "Bill Badguy" }
```

renvoie false

CONSTRUCT

CONSTRUCT renvoie un graphe RDF décrit par un gabarit de graphe.

```
PREFIX : <http://mlbda.fr/>

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT { ?x a :Adult }

WHERE {

    { ?x foaf:age ?a . }

    FILTER (?a > 18)
```

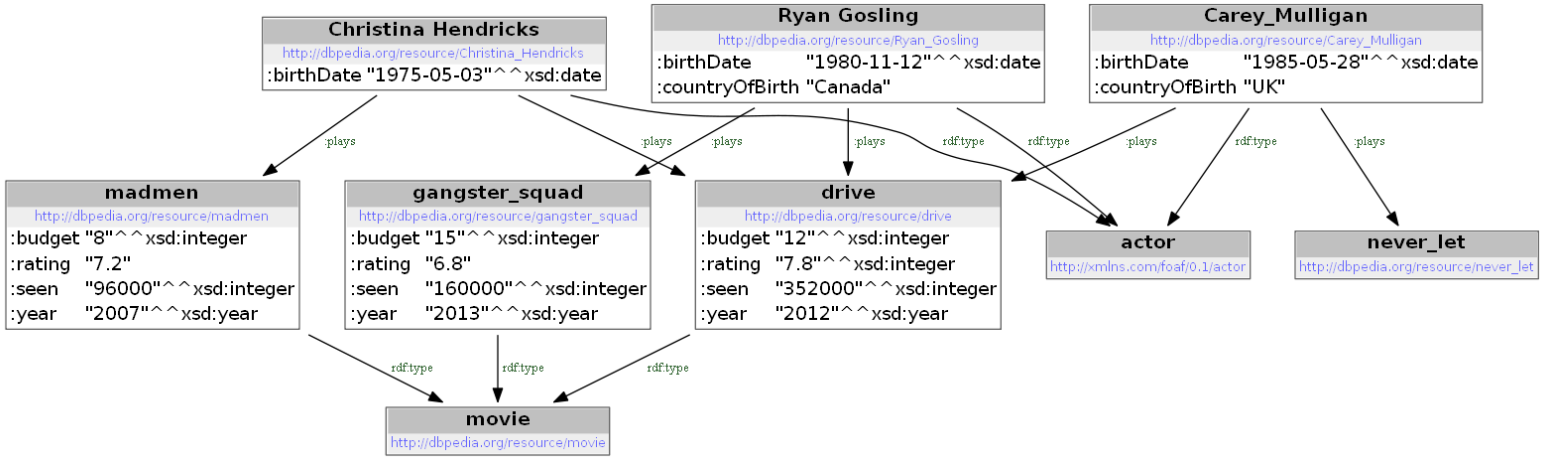



```
@prefix ns1: <http://mlbda.fr/> .

ns1:a a ns1:Adult .

ns1:b a ns1:Adult .
```

```
@prefix : <http://dbpedia.org/resource/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
:Ryan_Gosling
   a foaf:actor ;
   rdfs:label "Ryan Gosling";
   :birthDate "1980-11-12"^^xsd:date ;
   :countryOfBirth "Canada" .
:Christina_Hendricks
   a foaf:actor ;
   rdfs:label "Christina Hendricks" ;
   :birthDate "1975-05-03"^^xsd:date .
:Carey_Mulligan
   a foaf:actor ;
   rdfs:label "Carey_Mulligan" ;
   :birthDate "1985-05-28"^^xsd:date ;
   :countryOfBirth "UK".
```

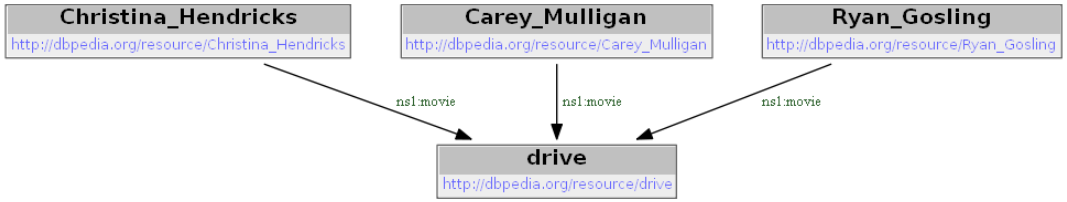
```
:Ryan_Gosling   :plays :drive , :gangster_squad .
:Christina_Hendricks   :plays :drive , :madmen .
:Carey_Mulligan :plays :drive, :never_let .

:drive
   a :movie;
   :budget "12"^^xsd:integer ;
   :year "2012"^^xsd:year ;
   :rating "7.8"^^xsd:integer;
   :seen "352000"^^xsd:integer.
:madmen
   a :movie;
   :budget "8"^^xsd:integer;
   :year "2007"^^xsd:year;
   :rating "7.2";
   :seen "96000"^^xsd:integer.
:gangster_squad
   a :movie;
   :budget "15"^^xsd:integer;
   :year "2013"^^xsd:year;
   :rating "6.8";
   :seen "160000"^^xsd:integer.
```

Exemple


Christina Hendricks
http://dbpedia.org/resource/Christina_Hendricks
:birthDate "1975-05-03"^^xsd:date
Ryan Gosling
http://dbpedia.org/resource/Ryan_Gosling
:birthDate "1980-11-12"^^xsd:date
:countryOfBirth "Canada"
Carey_Mulligan
http://dbpedia.org/resource/Carey_Mulligan
:birthDate "1985-05-28"^^xsd:date
:countryOfBirth "UK"
:plays
rdf:type
madmen
http://dbpedia.org/resource/madmen
:budget "8"^^xsd:integer
:rating "7.2"
:seen "96000"^^xsd:integer
:year "2007"^^xsd:year
gangster_squad
http://dbpedia.org/resource/gangster_squad
:budget "15"^^xsd:integer
:rating "6.8"
:seen "160000"^^xsd:integer
:year "2013"^^xsd:year
drive
http://dbpedia.org/resource/drive
:budget "12"^^xsd:integer
:rating "7.8"^^xsd:integer
:seen "352000"^^xsd:integer
:year "2012"^^xsd:year
actor
http://xmlns.com/foaf/0.1/actor
never_let
http://dbpedia.org/resource/never_let
movie
http://dbpedia.org/resource/movie

CONSTRUCT

```
prefix : <http://dbpedia.org/resource/>

construct { ?actor :movie ?movie }

where{ ?actor :plays ?movie.

        ?movie :budget 12
}
```

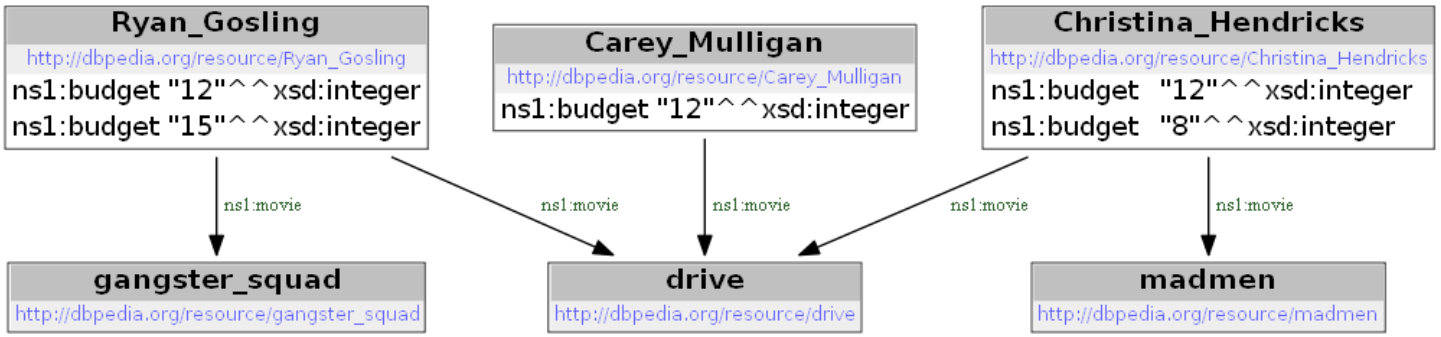



```
@prefix ns1: <http://dbpedia.org/resource/> .

ns1:Carey_Mulligan ns1:movie ns1:drive .

ns1:Christina_Hendricks ns1:movie ns1:drive .

ns1:Ryan_Gosling ns1:movie ns1:drive .
```

CONSTRUCT

```
prefix : <http://dbpedia.org/resource/>
construct { ?actor :movie ?movie;
                   :budget ?budget }
where{ ?actor :plays ?movie .
       ?movie :budget ?budget
    optional { ?movie :budget ?budget
          FILTER (?budget > 10) }
}
```

```
@prefix ns1: <http://dbpedia.org/resource/> .
@prefix xsd:
<http://www.w3.org/2001/XMLSchema#> .

ns1:Carey_Mulligan ns1:budget 12 ;
    ns1:movie ns1:drive .

ns1:Christina_Hendricks ns1:budget 8,
        12 ;
    ns1:movie ns1:drive,
        ns1:madmen .

ns1:Ryan_Gosling ns1:budget 12,
        15 ;
    ns1:movie ns1:drive,
        ns1:gangster_squad .
```



SELECT – GROUP BY

```
prefix : <http://dbpedia.org/resource/>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
select ?a (avg(?b) as ?mb)
 where { ?a :plays [ :budget ?b ] }
group by ?a
```

a	mb
<http://dbpedia.org/resource/Christina_Hendricks>	11.667
<http://dbpedia.org/resource/Ryan_Gosling>	13.5

CONSTRUCT

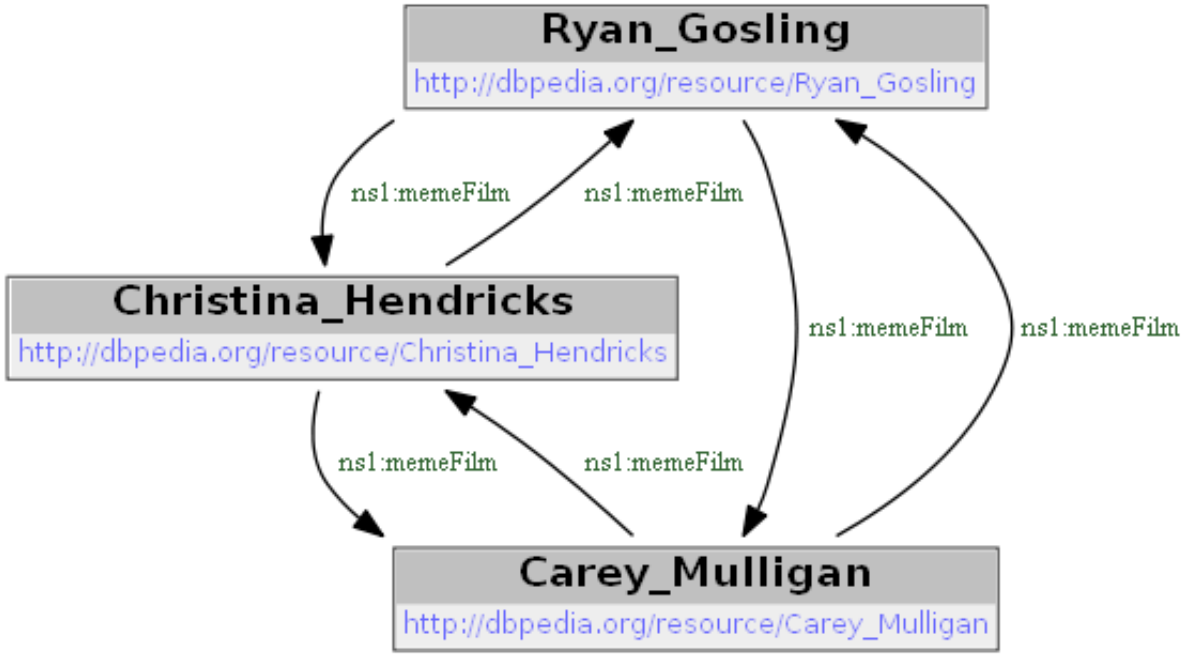
```
prefix : <http://dbpedia.org/resource/>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
construct {  ?actor :paysNaiss ?birthCount ; :nom ?nom . }
where { ?actor :plays :drive ; rdfs:label ?nom .
        optional { ?actor :countryOfBirth ?birthCount }
}
```

```
@prefix ns1: <http://dbpedia.org/resource/> .

ns1:Carey_Mulligan ns1:nom "Carey_Mulligan" ;
    ns1:paysNaiss "UK" .

ns1:Christina_Hendricks ns1:nom "Christina Hendricks" .

ns1:Ryan_Gosling ns1:nom "Ryan Gosling" ;
    ns1:paysNaiss "Canada" .
```

CONSTRUCT + FILTER

```
prefix : <http://dbpedia.org/resource/>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
construct {  ?actor1 :memeFilm ?actor2 }
where { ?actor1 :plays ?m . ?actor2 :plays ?m
  FILTER (?actor1 != ?actor2)
}
```

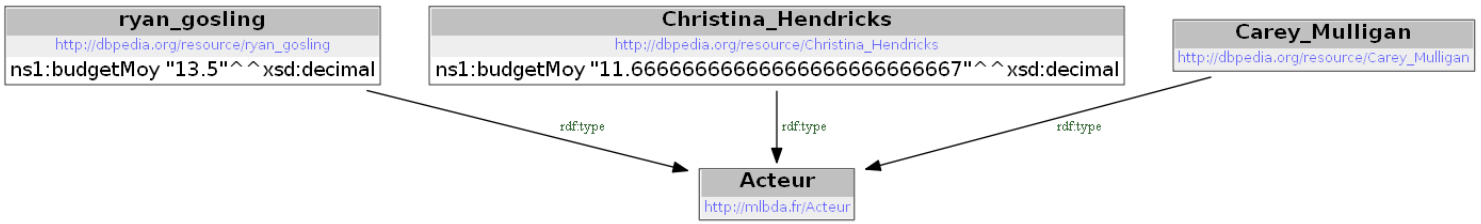
```
@prefix ns1: <http://dbpedia.org/resource/> .

ns1:Carey_Mulligan ns1:memeFilm ns1:Christina_Hendricks,
        ns1:Ryan_Gosling .

ns1:Christina_Hendricks ns1:memeFilm ns1:Carey_Mulligan,
        ns1:Ryan_Gosling .

ns1:Ryan_Gosling ns1:memeFilm ns1:Carey_Mulligan,
        ns1:Christina_Hendricks .
```


Ryan_Gosling
http://dbpedia.org/resource/Ryan_Gosling
ns1:memeFilm
ns1:memeFilm
Christina_Hendricks
http://dbpedia.org/resource/Christina_Hendricks
ns1:memeFilm
ns1:memeFilm
ns1:memeFilm
ns1:memeFilm
Carey_Mulligan
http://dbpedia.org/resource/Carey_Mulligan

CONSTRUCT et SELECT

```
…
construct {  ?a a :Acteur ; :budgetMoy ?mb  . }
where
{ ?a a foaf:actor
  { select ?a (avg(?b) as ?mb)
    where { ?a dbpedia:plays [ dbpedia:budget ?b ] }
    group by ?a
   }
}
```

```
...
ns2:Carey_Mulligan a ns1:Acteur .

ns2:Christina_Hendricks a ns1:Acteur ;
    ns1:budgetMoy 11.667 .

ns2:Ryan_Gosling a ns1:Acteur ;
    ns1:budgetMoy 13.5 .
```



Conclusion

Interrogation de graphes sémantiques

- Définition de motifs
- Recherche de données correspondant au motif dans le graphe (appariement de graphes)

Autres fonctionnalités

- Règles d'inférence : pour déduire des informations
- Nouveaux opérateurs et fonctions de filtre (agrégats et GROUP BY.. HAVING, …) (SPARQL1.1)
- Requêtes imbriquées

La suite des BD en Master

M1 S2 : SAM (Stockage et Accès aux Mégadonnées)

- Méthodes d'accès : index, arbres B+, hachage
- Optimisation de requêtes et algorithmes de jointures
- Conception et interrogation de BD réparties
- Transactions réparties
- SGBD parallèles

M2 S1 : BDLE (Bases de données à large échelle)

- Bases de données multidimensionnelles
- Map Reduce - Spark et Scala
- Interrogation de données semi-structurées en Spark – Optimisation
- Stockage à l'échelle du Web
- Grands graphes de données

M2SI : Linked Open Data et Apprentissage Symbolique

- Web sémantique : RDF, Sparql, OWL