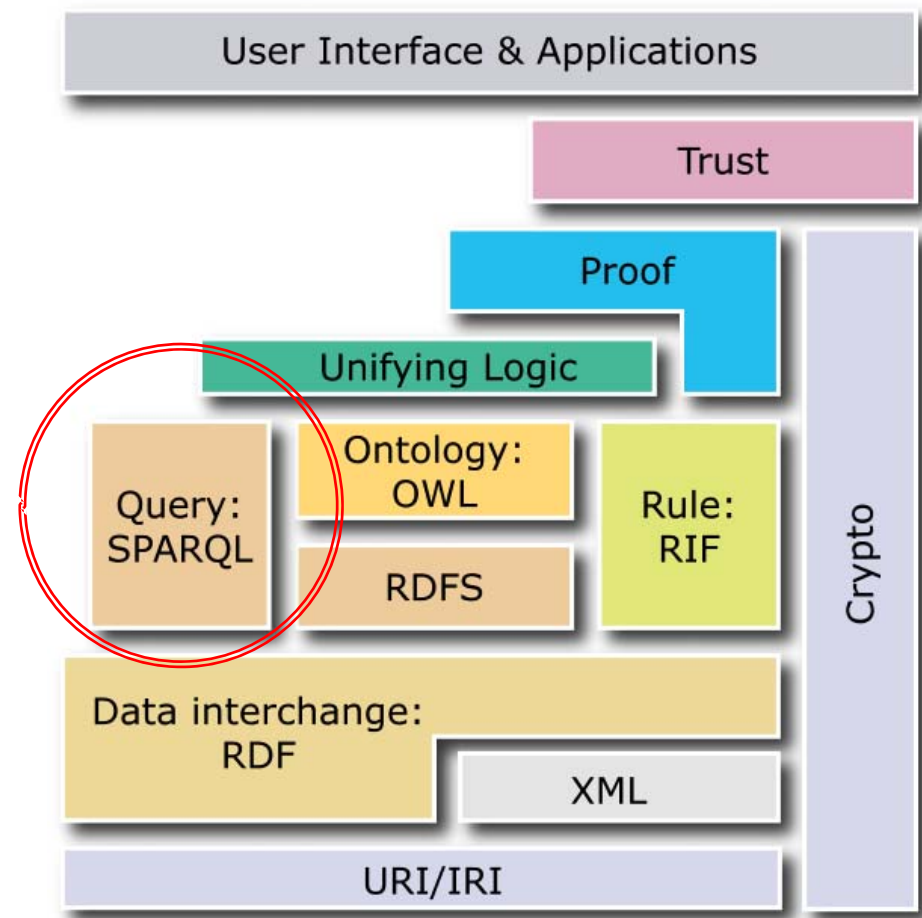


Interrogation du Web de Données - SPARQL -

Catherine COMPAROT
IRIT/UT2J

Equipe MELODI

Mars 2017



SPARQL ("sparqueul")

Ensemble de spécifications (langages, protocoles) pour interroger et manipuler des graphes RDF

- **SPARQL 1.1 : Recommandation du W3C, depuis mars 2013**
- Comparé à SPARQL 1.0 (janvier 2008), SPARQL 1.1 propose de nouvelles propriétés au langage de requête telles que :
 - les sous-requêtes,
 - l'assignation de variables,
 - l'expression de chemin dans un graphe,
 - le traitements de groupes de triplets, etc..
- SPARQL 1.1 inclut d'autres nouveautés (SPARQL Update par exemple) sur lesquelles on reviendra plus tard

Ce partie du cours traite essentiellement du langage de requête

(<http://www.w3.org/TR/sparql11-query/>)

La notion de "Endpoint SPARQL"

Un endpoint SPARQL est un service de traitement de requêtes SPARQL supportant le protocole SPARQL (SPARQL 1.1 Protocol) pour interroger des entrepôts RDF distants

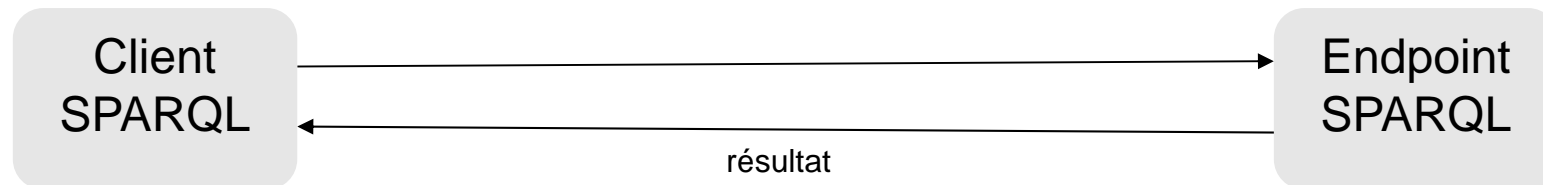
Il accepte des requêtes SPARQL et retourne des résultats via HTTP

Exemple de endpoint :

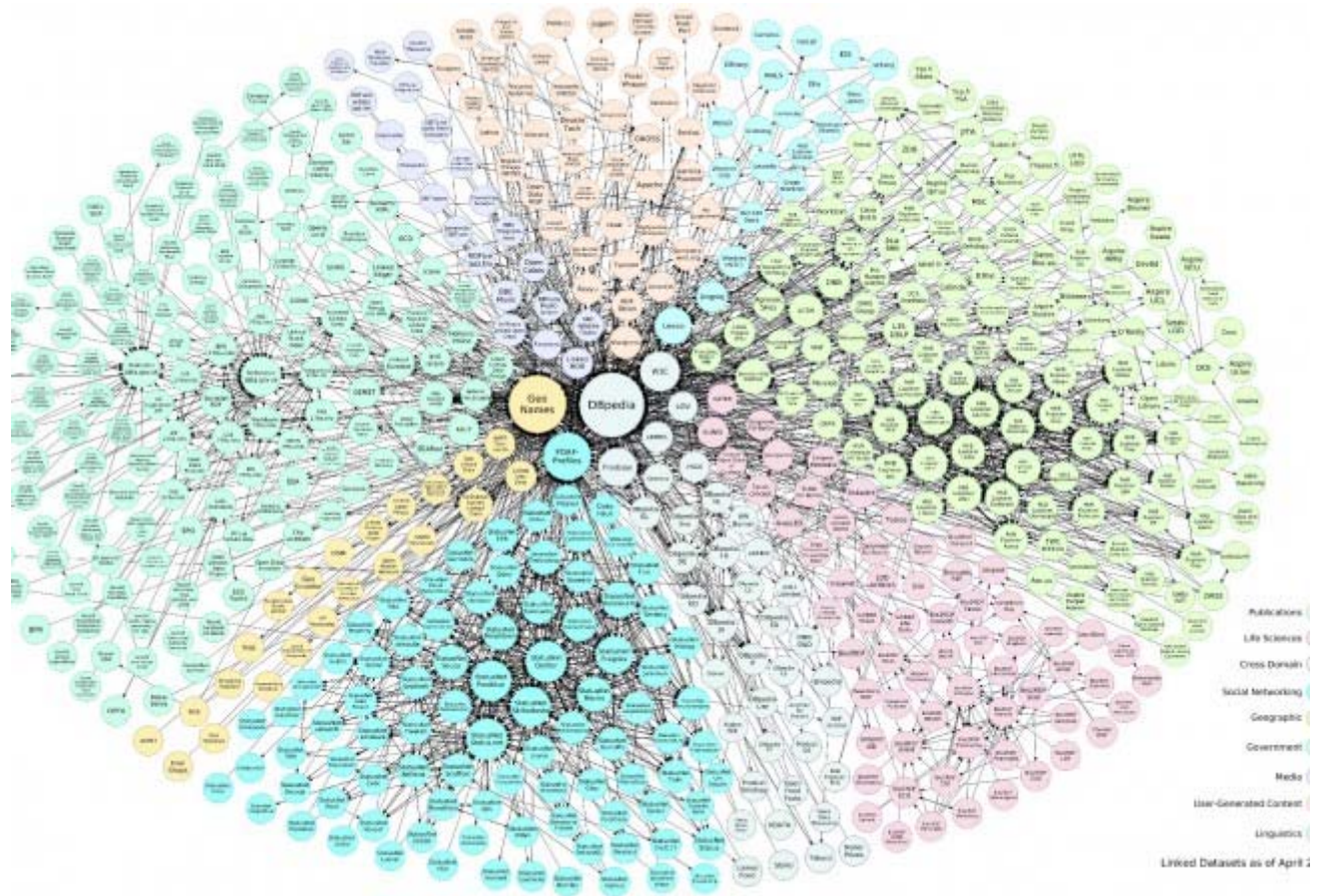
- DBpédia : <http://dbpedia.org/sparql>

Principe :

<http://dbpedia.org/sparql/?query=select+%3Fp+where+%3D%3A%7B%3Fp+foaf%3Ahomepage+%3D+%3Fp.%0D%0A%7D+limit+10>



Une Liste de endpoints publiques et de leur statut (disponibilité, compatibilité SPARQL, etc.) : <http://sparqls.ai.wu.ac.at>



SPARQL :

INTERROGER/MANIPULER DU RDF

Interroger/manipuler du RDF ?

- Des ressources avec des URI

Ex. : Désignation de l'aluminium dans l'**espace de noms** des éléments chimiques :



<http://www.daml.org/2003/01/periodictable/PeriodicTable#Al>

← *Espace de noms* → *Nom local*

← *Nom qualifié (QName) ou URI* →

- Des ressources décrites sous forme de **triplets** (assertions RDF) de la forme :
< sujet, prédicat, objet >

Ex. : « L'aluminium appartient au Groupe 13 des éléments chimiques » , peut être représenté en RDF par :

< <http://www.daml.org/2003/01/periodictable/PeriodicTable#Al> ,
 <http://www.daml.org/2003/01/periodictable/PeriodicTable#group> ,
 http://www.daml.org/2003/01/periodictable/PeriodicTable#Group_13 >



Interroger/manipuler du RDF ?

Exemple : la table périodique des éléments chimiques

<http://www.daml.org/2003/01/periodictable/PeriodicTable.owl>

Group	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Period																		
1	1 H																	2 He
2	3 Li	4 Be											5 B	6 C	7 N	8 O	9 F	10 Ne
3	11 Na	12 Mg											13 Al	14 Si	15 P	16 S	17 Cl	18 Ar
4	19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
5	37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe
6	55 Cs	56 Ba	* 71 Lu	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi	84 Po	85 At	86 Rn
7	87 Fr	88 Ra	** 103 Lr	104 Rf	105 Db	106 Sg	107 Bh	108 Hs	109 Mt	110 Ds	111 Rg	112 Uub	113 Uut	114 Uuq	115 Uup	116 Uuh	117 Uus	118 Uuo
*Lanthanoids			* 57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb		
**Actinoids			** 89 Ac	90 Th	91 Pa	92 U	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No		

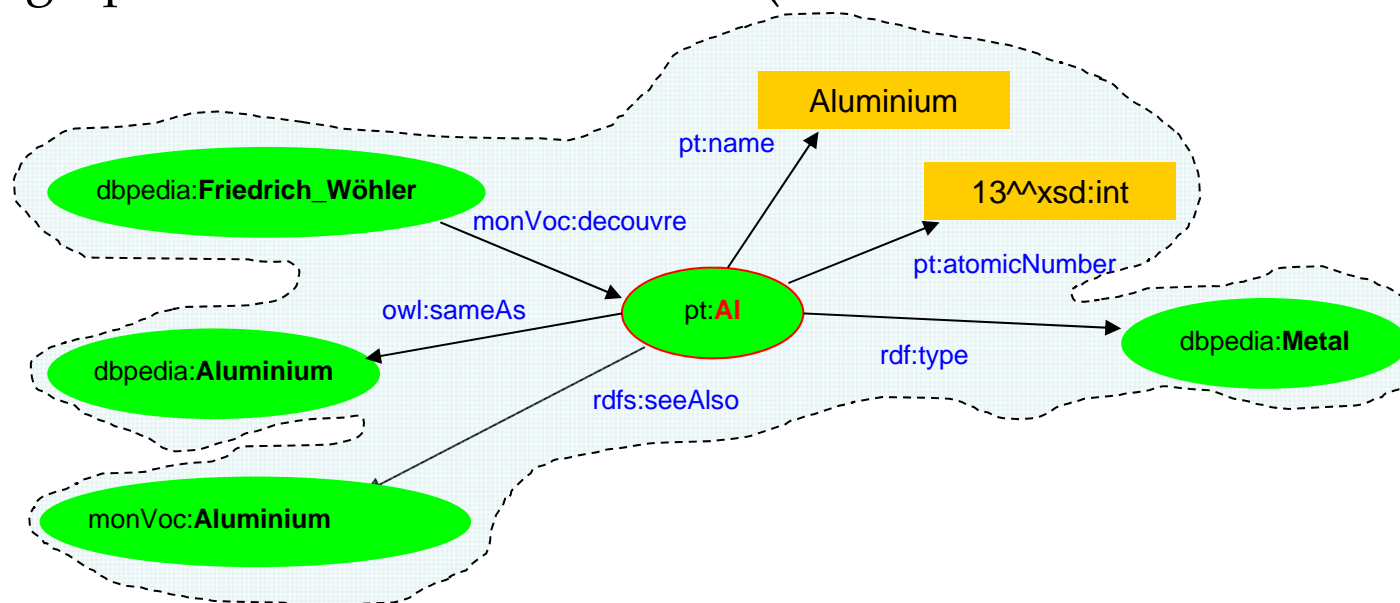
L'aluminium



Interroger/manipuler du RDF ?

- Des graphes RDF, i.e. des triplets avec des URI issus de divers vocabulaires

Ex. : Un graphe décrivant l'Aluminium (`<http://www.daml.org/2003/01/periodictable/PeriodicTable#Al>`)



Les **littéraux** sont représentés par un rectangle, les **ressources** (URI) sont représentées en bleu (prédicats d'un triplet) ou par une ellipse (sujets ou objets d'un triplet)

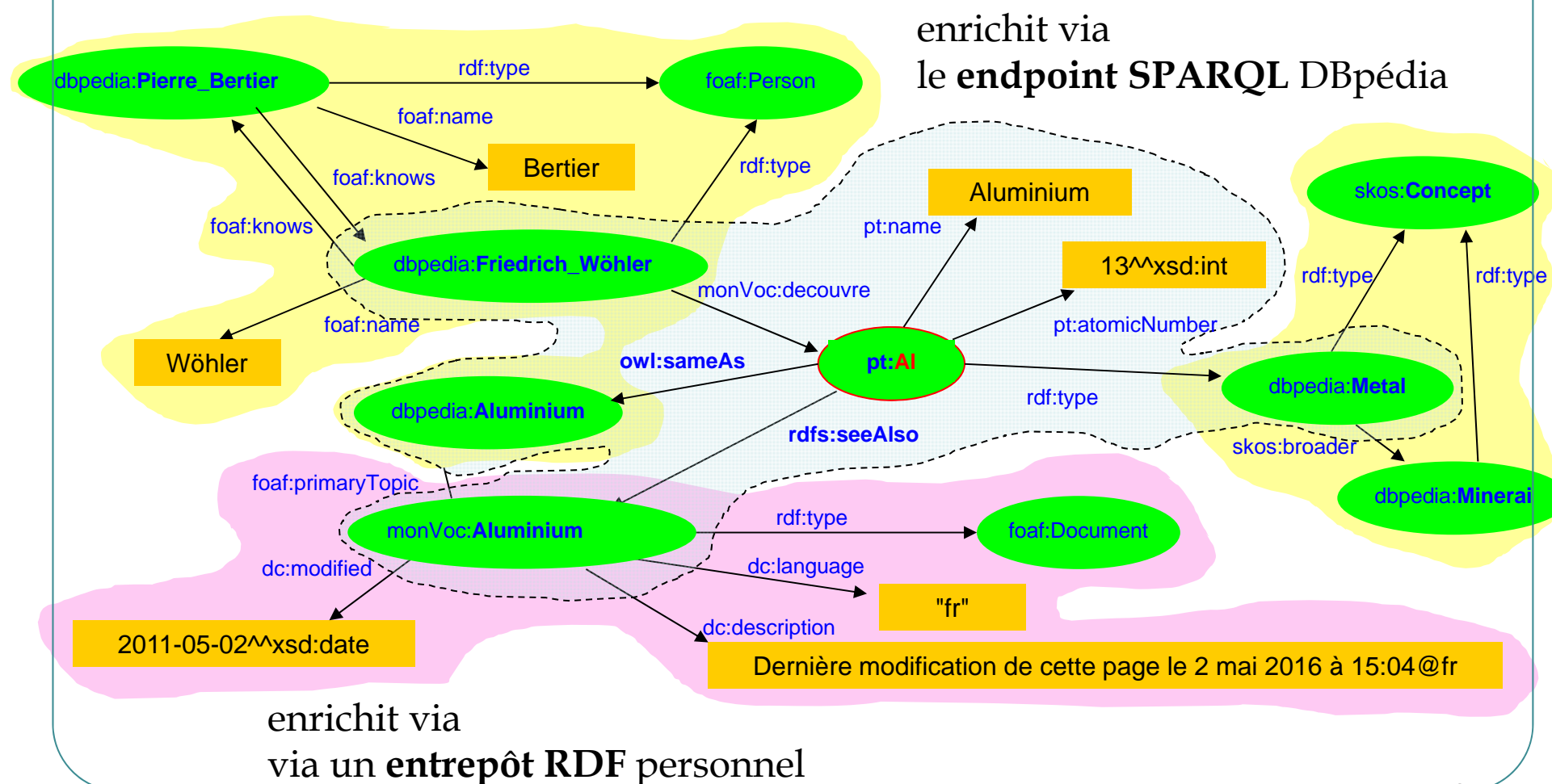
Les noms des espaces de nom sont désignés par des **préfixes** :

- *dbpedia* est le préfixe pour `http://dbpedia.org/resource/`
- *pt* est le préfixe pour `http://www.daml.org/2003/01/periodictable/PeriodicTable#`

Interroger/manipuler du RDF ?

- Des ensemble de données, ou datasets RDF, interconnectés

Ex. : Le graphe décrivant l'Aluminium




Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (1)

- **Découverte de quelques triplets** (pour appréhender le vocabulaire) :

```
SELECT ?s ?p ?o
WHERE {
    ?s ?p ?o .
}
LIMIT 30
```

Les 30 (clause *LIMIT*) premiers triplets (?s ?p ?o) de l'entrepôt sont recherchés (clause *WHERE*) et retournés tels quels (clause *SELECT*) par la requête.

Le résultat de la requête est présenté sous forme d'un tableau de données



s	p	o
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#symbol	"Tm"
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#period	http://www.daml.org/2003/01/periodictable/PeriodicTable#period_6
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#name	"thulium"
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#color	"silvery white"
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#NamedIndividual
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.daml.org/2003/01/periodictable/PeriodicTable#Element
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#casRegistryID	"7440-30-4"
http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm	http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (1) - Résultat

s	p	o
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#symbol>	"Tm"
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#period>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#period_6>
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#name>	"thulium"
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#color>	"silvery white"
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#Element>
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#casRegistryID>	"7440-30-4"
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Tm>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#block>	<http://www.daml.org/2003/01/periodictable/PeriodicTable#block_6>

Découverte du vocabulaire utilisé :

- le nom des espaces de noms,
- les propriétés (relations),
- Le domaine et le co-domaine des relations,
- etc.

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (2)

- **Découverte de quelques triplets** (pour appréhender le vocabulaire) :

```
SELECT DISTINCT ?classe  
WHERE { ?instance a ?classe . }  
LIMIT 30
```

- Retrouve tous les triplets ayant pour prédicat *rdf:type*
- Seul l' "objet" de ces triplets (désigné par *?classe*) est retourné

a (dans le patron de triplet) est le raccourci SPARQL pour désigner la propriété RDF *type* (*rdf:type*) indiquant la classe dont une ressource est instance.

?instance a ?classe est le **patron de triplet** recherché :

- Le sujet du triplet est désigné par la variable *?instance*
- La prédicat du triplet est la propriété *rdf:type*
- L' "objet" du triplet est désigné par la variable *?classe*

DISTINCT évite qu'une classe instanciée plusieurs fois, apparaisse plusieurs fois dans le résultat de la requête (suppression des doublons)

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (2) - Résultat

- **Découverte de quelques triplets** (pour appréhender le vocabulaire) :

```
SELECT DISTINCT ?classe
WHERE { ?instance a ?classe . }
LIMIT 30
```



classe

[<http://www.daml.org/2003/01/periodictable/PeriodicTable#Period>](http://www.daml.org/2003/01/periodictable/PeriodicTable#Period)

[<http://www.daml.org/2003/01/periodictable/PeriodicTable#StandardState>](http://www.daml.org/2003/01/periodictable/PeriodicTable#StandardState)

[<http://www.w3.org/2002/07/owl#Restriction>](http://www.w3.org/2002/07/owl#Restriction)

[<http://www.w3.org/2002/07/owl#NamedIndividual>](http://www.w3.org/2002/07/owl#NamedIndividual)

[<http://www.daml.org/2003/01/periodictable/PeriodicTable#Group>](http://www.daml.org/2003/01/periodictable/PeriodicTable#Group)

[<http://www.daml.org/2003/01/periodictable/PeriodicTable#Element>](http://www.daml.org/2003/01/periodictable/PeriodicTable#Element)

[<http://www.daml.org/2003/01/periodictable/PeriodicTable#Couche>](http://www.daml.org/2003/01/periodictable/PeriodicTable#Couche)

- Retrouve tous les triplets ayant pour prédicat *rdf:type*
- Seul l' "objet" de ces triplets (désigné par *?classe*) est retourné

Vocabulaire :

- Spécifique au domaine,
- OWL
- Etc.

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (3)


- Découverte de quelques classes et instances (pour appréhender le modèle) :

```
SELECT ?classe ( count( ?instance ) AS ?ninstances )  
WHERE { ?instance a ?classe . }  
GROUP BY ?classe  
ORDER BY DESC(?ninstances) LIMIT 30
```

GROUP BY ?classe (SPARQL 1.1) : Les triplet sont regroupés par valeur de *?classe*

La fonction *count* est appliquée à chaque groupe ; la colonne correspondant dans le résultat est la variable nommée *?ninstances*

Les résultats sont triés (*ORDER BY*) par valeur décroissante (*DESC*) de *?ninstances*



classe	ninstances
<http://www.w3.org/2002/07/owl#NamedIndividual>	"159"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Element>	"118"^^xsd:integer
<http://www.w3.org/2002/07/owl#Restriction>	"22"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#Group>	"20"^^xsd:integer


Les requêtes de groupe donc sont supportées par l'interpréteur SPARQL

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (3 bis)

- Découverte de quelques classes et instances (pour appréhender le modèle) :

```
SELECT ?classe ( count( ?instance ) AS ?nbinstances )
WHERE {
    ?instance a ?classe .
    FILTER( contains( str( ?classe ), "PeriodicTable" ) )
}
GROUP BY ?classe
ORDER BY DESC(?nbinstances) LIMIT 30
```

Seules les ressources propres au domaine (déterminées par l'espace de nom de leur URI) sont retournées (*FILTER*)



classe	nbinstances
< http://www.daml.org/2003/01/periodictable/PeriodicTable#Element >	"118"^^xsd:integer
< http://www.daml.org/2003/01/periodictable/PeriodicTable#Group >	"20"^^xsd:integer
< http://www.daml.org/2003/01/periodictable/PeriodicTable#Period >	"7"^^xsd:integer
< http://www.daml.org/2003/01/periodictable/PeriodicTable#Block >	"4"^^xsd:integer
< http://www.daml.org/2003/01/periodictable/PeriodicTable#StandardState >	"4"^^xsd:integer

Les classes les plus représentatives

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (4)

- Découverte de quelques propriétés (pour appréhender le modèle) :

```
SELECT DISTINCT ?p
WHERE {
  [] ?p ?o .
  FILTER( ! contains( str(?p), "PeriodicTable" ) )
} LIMIT 30
```

[] est équivalent à une variable ; est aussi appelé “**noeud blanc**” (ou “noeud anonyme” ou “blank node”)

p



[<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>](http://www.w3.org/1999/02/22-rdf-syntax-ns#type)

[<http://www.w3.org/2002/07/owl#onProperty>](http://www.w3.org/2002/07/owl#onProperty)

[<http://www.w3.org/2002/07/owl#allValuesFrom>](http://www.w3.org/2002/07/owl#allValuesFrom)

[<http://www.w3.org/2002/07/owl#cardinality>](http://www.w3.org/2002/07/owl#cardinality)

[<http://www.w3.org/1999/02/22-rdf-syntax-ns#rest>](http://www.w3.org/1999/02/22-rdf-syntax-ns#rest)

[<http://www.w3.org/1999/02/22-rdf-syntax-ns#first>](http://www.w3.org/1999/02/22-rdf-syntax-ns#first)

[<http://www.w3.org/2000/01/rdf-schema#comment>](http://www.w3.org/2000/01/rdf-schema#comment)

Propriétés d'autres
vocabulaires

Requêtes utiles pour appréhender un entrepôt RDF avec SPARQL (5)

- **Découverte de quelques propriétés** (pour appréhender le modèle) :

```
SELECT ?p ( count( ?s ) AS ?ninstances )  
WHERE {  
    ?s ?p ?o .  
}  
GROUP BY ?p  
ORDER BY DESC( ?ninstances) LIMIT 5
```

p	ninstances
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	"364"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#element>	"236"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#name>	"127"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#atomicNumber>	"118"^^xsd:integer
<http://www.daml.org/2003/01/periodictable/PeriodicTable#block>	"118"^^xsd:integer

Les 5 propriétés les plus utilisées dans les données

A propos de la sérialisation RDF

- Différents formats de sérialisation tels que :
 - **RDF/XML** :
 - Vocabulaire spécifique pour décrire les triplets
 - Syntaxe complexe et verbeuse (pas facile à lire par un humain)
 - **JSON**
 - **CSV/TSV** (SPARQL 1.1)
- **Turtle** (Terse RDF Triple Language) : recommandation du W3C, février 2014,
 - <https://www.w3.org/TR/2014/REC-turtle-20140225/>
 - Syntaxe textuelle compacte et naturelle
 - Sert de base pour l'expression de requêtes SPARQL

NB : TriG (<http://www.w3.org/TR/2014/REC-trig-20140225/>) : version de Turtle étendue aux graphes nommés (named graph) RDF

A propos de la sérialisation RDF

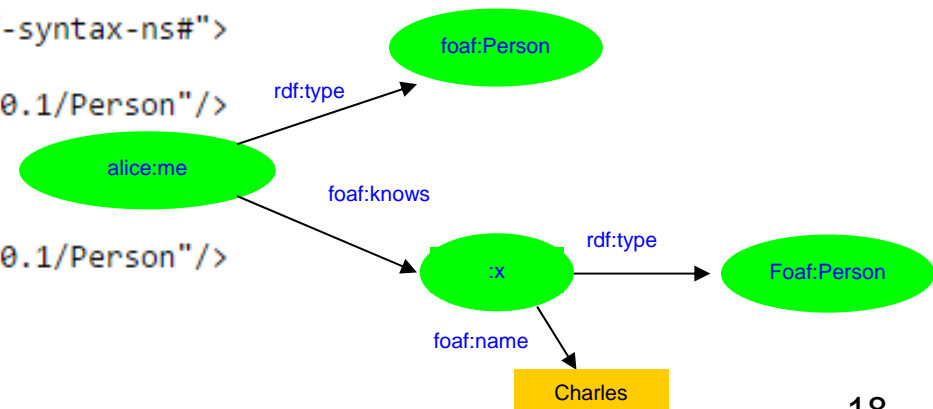
- Exemple de sérialisation RDF/XML :

XML est un standard ISO depuis 1999 :

Une représentation RDF/XML de données permet d'exploiter des efforts de développement et d'apprentissage antérieurs pour traiter du RDF : analyseurs syntaxiques (parsers), langages de schéma (DTD, XML-Schema, Relax-NG...), langages de requêtes (XPath, XQuery), langages de transformation (XSL-T), etc.

Ex. : un fichier RDF/XML (.xml) contenant un graphe RDF de 4 triplets

```
<rdf:RDF xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  <rdf:Description rdf:nodeID="x">
    <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
    <foaf:name>Charles</foaf:name>
  </rdf:Description>
  <rdf:Description rdf:about="alice/me">
    <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
    <foaf:knows rdf:nodeID="x"/>
  </rdf:Description>
</rdf:RDF>
```



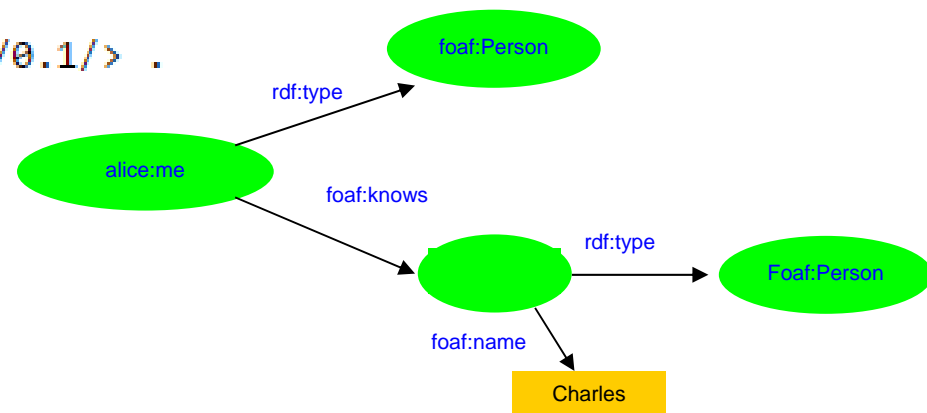
A propos de la sérialisation RDF

- **Exemple de sérialisation Turtle (RDF 1.1 Turtle) :**

Un triplet RDF est décrit en Turtle par une suite de termes terminée par .

Ex. : un fichier turtle (.ttl) contenant un graphe RDF de 4 triplets

```
@prefix alice: <alice/> .  
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
  
alice:me a foaf:Person.  
alice:me foaf:knows _:c.  
_:c a foaf:Person.  
_:c foaf:name "Charles".
```



Description de triplets en SPARQL

- S'appuie sur Turtle

prefix pt: <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .

prefix xsd: <http://www.w3.org/2001/XMLSchema#>

pt:Al

a

pt:Element .

pt:Al

pt:symbol

"Al"^^xsd:string ;

pt:atomicNumber "13"^^xsd:int ;

pt:atomicWeight "26.981538" ;

pt:group

pt:group_13 .

pt:Al

pt:name

"aluminium"@fr , "aluminium"@en .

pt:Element

pt:group_13

a : raccourci pour *rdf:type* (lie une instance à sa classe).

= : raccourci pour *owl:sameAs*

Le **;** indique que le triplet suivant a le même « sujet » (i.e. *pt:Al*) ce qui évite de réécrire le sujet.

^^ pour typer un littéral

@lang indique la langue dans laquelle la chaîne de caractères qui précède, est formulée

La **,** permet l'écriture simplifiée d'une propriété multi-valuée, et évite de réécrire le sujet et l'objet



SPARQL :

LE LANGAGE DE REQUÊTES

Exécution d'une requête SPARQL

- Une requête SPARQL s'adresse à un dataset RDF qui peut être constitué de :
 - Un fichier RDF local,
 - Un endpoint SPARQL qui renvoie le résultat via HTTP.
- Il y a différents formats pour le résultat d'une requête SPARQL :
 - XML. La table résultant d'une requête est représentée en XML
 - JSON. Le résultat peut être directement exploité dans une application web.
 - CSV/TSV. Le résultat peut être directement exploité un tableur
 - RDF. Le résultat est constitué de triplets RDF qui peuvent être sérialisés de différentes façons (RDF/XML, N-Triples, Turtle, etc.)
 - HTML. Une transformation XSL d'une table en XML (pour la lisibilité).

Requêtes “SPARQL Query Language”



Trois types de requêtes :

- **SELECT** et **DESCRIBE** pour interroger des datasets RDF
- **ASK** pour tester l'existence de graphes
- **CONSTRUCT** pour construire des triplets RDF

Syntaxe de base du **SELECT** (reprise en partie dans les autres types de requêtes) ;
entre [], les éléments facultatifs.

[BASE <i><espace de noms d'IRI></i>]	}	prologue
PREFIX <i>préfixe: <contenu abrégé des IRI></i>		
SELECT résultat, i.e. projection sur des <i>variables</i>	}	entête
[FROM <i>dataset RDF interrogé</i>]		
WHERE { patron de graphe }	}	corps
[ORDER BY expression]		
[LIMIT valeur > 0]		

Retour sur la syntaxe SPARQL

- Trouver tous les noms (propriété *name*) mentionnés dans le dataset interrogé

```
@prefix : <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .
@prefix owl: <http://www.w3.org/2002/07/owl#>

:Group a owl:Class.
:Group_13 a :Group .
:Group_16 a :Group .
:Element a owl:Class.
:Al a :Element .
:Al :symbol "Al" .
:Al :name "aluminium" .
:Al :group :Group_13.
:Se a :Element .
:Se :symbol "Se" .
:Se :name "selenium" .
:Se :group :Group_16.
```

...

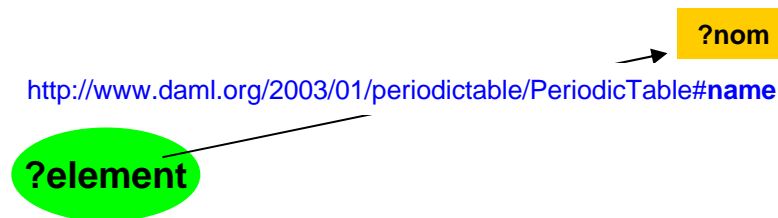
← Identification du ***patron de triplet*** ...

... :name " ... "

<http://www.daml.org/2003/01/periodictable/PeriodicTable.owl> (syntaxe turtle)

Retour à la syntaxe SPARQL

- Autrement dit :
 1. Données recherchées : tous les sujets (*?element*) et les objets (*?nom*) des triplets RDF liés par le prédicat *:name*, soit le ***patron de triplet*** :



2. Résultat : toutes les valeurs de *?nom*
- Soit, en SPARQL :

PREFIX : `<http://www.daml.org/2003/01/periodictable/PeriodicTable#>`
SELECT `?nom`
WHERE {
 `?element :name ?nom .`
}

 ← Le ***patron de triplet***

Retour à la syntaxe SPARQL

- Résultat : une table avec les valeurs qui satisfont la requête :

nom
Actinoid^^http://www.w3.org/2001/XMLSchema#string
Alkali metal^^http://www.w3.org/2001/XMLSchema#string
Alkaline earth metal^^http://www.w3.org/2001/XMLSchema#:
Chalcogen^^http://www.w3.org/2001/XMLSchema#string
Coinage metal^^http://www.w3.org/2001/XMLSchema#string
Halogen^^http://www.w3.org/2001/XMLSchema#string
Lanthanoid^^http://www.w3.org/2001/XMLSchema#string
Noble gas^^http://www.w3.org/2001/XMLSchema#string
Prictogen^^http://www.w3.org/2001/XMLSchema#string
actinium^^http://www.w3.org/2001/XMLSchema#string
aluminium^^http://www.w3.org/2001/XMLSchema#string
americium^^http://www.w3.org/2001/XMLSchema#string
antimony^^http://www.w3.org/2001/XMLSchema#string
argon^^http://www.w3.org/2001/XMLSchema#string
arsenic^^http://www.w3.org/2001/XMLSchema#string
astatine^^http://www.w3.org/2001/XMLSchema#string
barium^^http://www.w3.org/2001/XMLSchema#string

Patron de graphes d'une requête

Exemple : Nom, symbole et no des éléments chimiques ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

SELECT *

WHERE {

?x :symbol ?symbol .

?x :name ?name .

?x :atomicNumber ?number .

}

Patron de graphes :

http://www.daml.org/2003/01/periodictable/PeriodicTable#symbol

http://www.daml.org/2003/01/periodictable/PeriodicTable#name

http://www.daml.org/2003/01/periodictable/PeriodicTable#atomicNumber

Le dataset interrogé :

@prefix : <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .

...

:Al a :Element.

:Al :name "aluminium" ; :symbol "Al" ; :number "13".

:Se a table:Element.

:Se :name "selenium" ; :symbol "Se".

...

:Element a owl:Class

x	name	symbol	number
Th	thorium	Th	90
C	carbon	C	6
He	helium	He	2
Fr	francium	Fr	87
Li	lithium	Li	3
Gd	gadolinium	Gd	64

Requête : patron de graphes

Exercice : Nom et numéro de <<http://www.daml.org/2003/01/periodictable/PeriodicTable#Al>> ?

Le dataset interrogé :

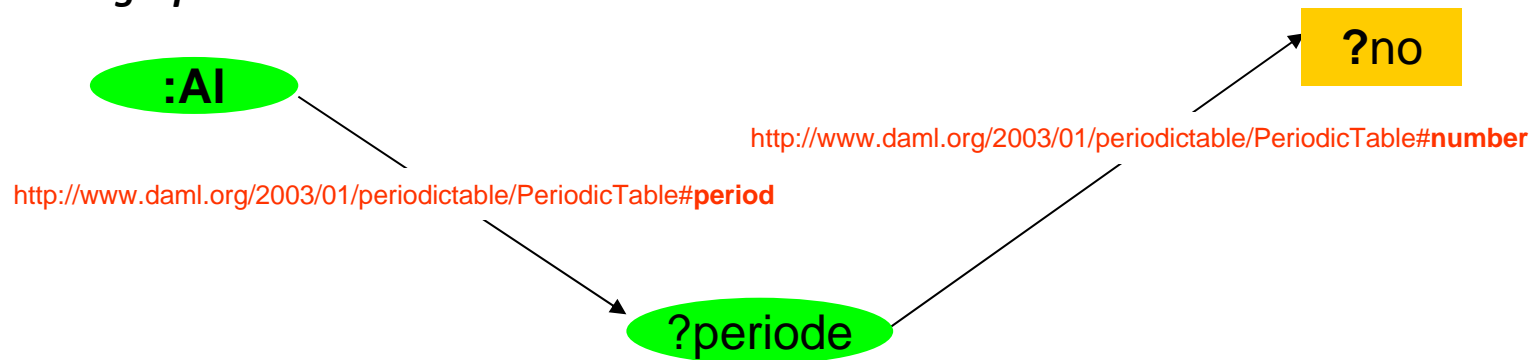
```
@prefix : <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .  
@prefix owl : <http://www.w3.org/2002/07/owl#>  
Al a :Element.  
:Al :name "aluminium" ; :symbol "Al" ; :number "13"..  
:Se a table:Element.  
:Se :name "selenium" ; :symbol "Se" ; :number "34"...  
...
```

Patron de traversée de graphes



Exemple : No de la période de l'aluminium ?

Patron de graphes :



PREFIX : <`http://www.daml.org/2003/01/periodictable/PeriodicTable#`>

SELECT `?no`

WHERE {

`:Al :period ?periode`.

`?periode :number ?no`.

}

Ou (SPARQL 1.1), avec un chemin de propriétés
(Cf. plus loin) :

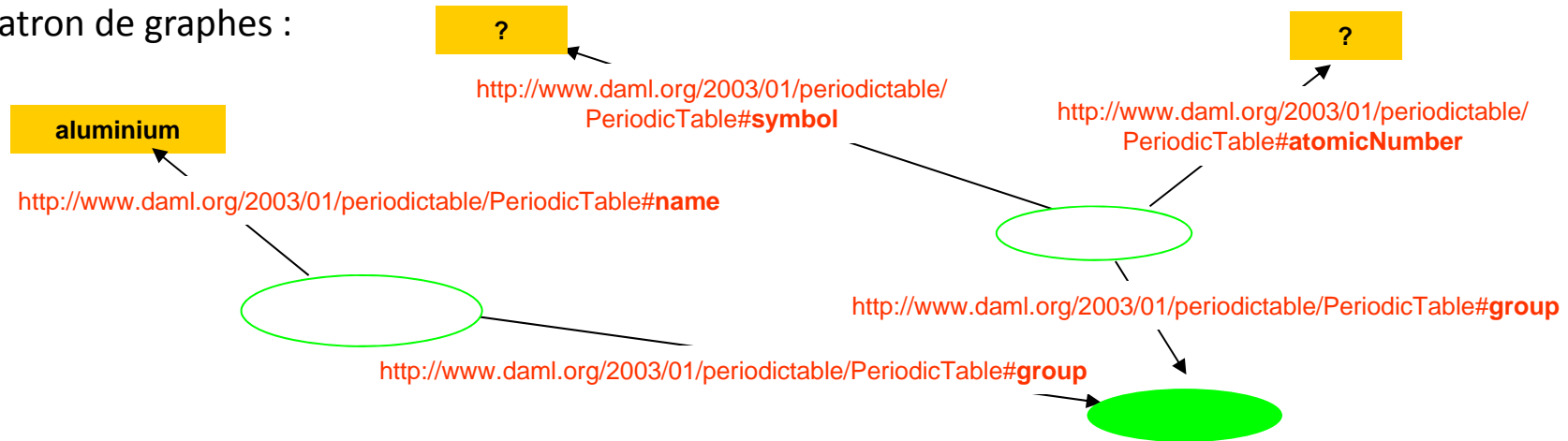
`:Al :period/number ?no`.

Patron de graphes



Exemple : No et symbole des éléments du même groupe que l'aluminium ?

Patron de graphes :



PREFIX : <`http://www.daml.org/2003/01/periodictable/PeriodicTable#`>

SELECT `?n ?s`

WHERE {

`?e1 :name "aluminium" ; :group ?g.`

`?e2 :group ?g ; :symbol ?s ; :atomicNumber ?n.}`

n	s
5	B
13	Al
31	Ga
49	In
81	Tl
113	Uut

Opérateurs des chemins de propriétés

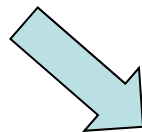
Les chemins de propriétés sont définis dans SPARQL 1.1 ; ils permettent la simplification de l'écriture de patrons de graphes

Les opérateurs pour décrire des chemins de propriétés sont :

- L'alternative : **|** Ex.: ?x :name**|** rdfs:label ?y.
- La séquence : **/** Ex.: ?x :group**/** rdfs:label ?y.
- Le chemin inverse (objet ➔ sujet) : **^** Ex.: ?x **^** rdfs:label ?y.
- Un chemin de longueur arbitraire : **+** Ex.: ?x rdfs:SubClassOf**+** ?y
- La négation de chemin : **!** Ex.: ?x **!** rdfs:label ?y.
- Une application :
 - Les ressources et types inférés : **rdf:type/rdf:subClassOf***

Exemple :

```
SELECT ?x WHERE {  
    ?x ^rdfs:label ?y.  
}
```



```
x  
"aluminium"@fr  
"argent"@fr  
"actinium"@fr  
"argon"@fr
```

Filtres de base : FILTER



Opérateurs pour le filtrage :

- Mathématiques : **+**, **-**, *****, **/**
 - (SPARQL 1.1) : **abs**, **round**, **ceil**, **floor**, **RAND**
- (SPARQL 1.1) Date/heure : **now**, **year**, **month**, **day**, **hours**, **minutes**, **seconds**, **timezone**, **tz**
- Chaînes : **lang**, **langMatches**
 - (SPARQL 1.1) **strlen**, **substr**, **ucase**, **strstarts**, **strbefore**, **concat**, etc.
- Etc.

Exemple : Couleur des éléments du groupe 13 ayant un poids inférieur à 100 ?

PREFIX : `<http://www.daml.org/2003/01/periodictable/PeriodicTable#>`

SELECT distinct ?c

WHERE {

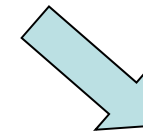
 ?element :color ?c ;

 :group : group_13 ;

 :atomicWeight ?w .

FILTER (?w <= 100)

}



c
"silvery"^^<http://www.w3.org/2001/XMLSchema#string>
"silvery white"^^<http://www.w3.org/2001/XMLSchema#string>
"black"^^<http://www.w3.org/2001/XMLSchema#string>

Filtres : sur les chaines (regex)



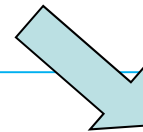
regex : fonction qui filtre les littéraux sans étiquette de langue

- Cf. le langage d'expression régulière de XQuery

Exemple : les atomes comportant un « a » ou un « l » dans leur nom ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

```
SELECT ?name WHERE {  
    ?e :name ?name.  
    FILTER regex(?name, "[al]", "i")  
}
```



name
"Halogen"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Alkaline earth"
"calcium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"gallium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"californium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"cobalt"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"palladium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Coinage"
"tantalum"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"aluminium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"thallium"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Alkali"
"Chalcogen"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>



Filtres : sur les chaines (Xpath)

Fonctions **Xpath** : Cf. <http://www.w3.org/TR/xpath-functions/>

Espace de noms pour les fonctions XPATH :

<http://www.w3.org/2005/xpath-functions#>

Les atomes dont le nom ne commence pas par les mêmes lettres que leur symbole atomique ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

PREFIX fn: <http://www.w3.org/2005/xpath-functions#>


SELECT ?s ?n

WHERE {

 ?e :name ?n ; :symbol ?s.

 FILTER (! fn:starts-with(?n , fn:lower-case(?s)))

}



s	\hat{A}	n
"Ag"		"silver"
"As"		"arsenic"
"At"		"astatine"
"Au"		"gold"
"Bh"		"bohrium"
"Bk"		"berkelium"

Filtres : EXISTS

EXISTS { *patron de graphe* } : fonction qui retourne VRAI si le patron de graphes a des correspondances dans le dataset interrogé

Exemple : Symbole des atomes ayant une couleur spécifique (qui leur est propre)?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

SELECT ?s WHERE {

 ?e1 :color ?c; :symbol ?s.

FILTER not EXISTS {

 ?e2 :color ?c .

FILTER (?e1 != ?e2). **}**

}

Le dataset interrogé :

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .

PREFIX owl : <http://www.w3.org/2002/07/owl#>

:Group a owl:Class.

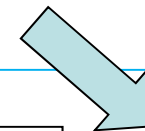
:Group_13 a :Group .

:Group_16 a :Group .

:Element a owl:Class.

:Al a :Element ; :symbol "Al" ; :name "aluminium" ; :color "silvery".

:Se a :Element ; :symbol "Se" ; :name "selenium" ; :color "grey, metallic lustre".



s
"B"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Os"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"I"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Ni"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Ta"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"Uuo"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"S"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>
"O"^^<http://www.w3.org/2003/01/periodictable/PeriodicTable#>

La clause FILTER : EXISTS (suite)

Exercice : Nom et poids de l'atome le plus léger ?

Le dataset interrogé :

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#> .

PREFIX owl : <http://www.w3.org/2002/07/owl#>

:Group a owl:Class.

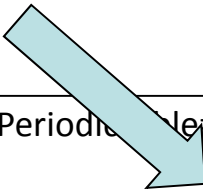
:Group_13 a :Group .

:Group_16 a :Group .

:Element a owl:Class.

:Al a :Element ; :symbol "Al" ; :name "aluminium" ; :atomicWeight "26.981539".

:Se a :Element ; :symbol "Se" ; :name "selenium" ; :atomicWeight "78.96".

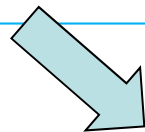


n	w1
"hydrogen"^^<http://www.w3.org/2001/XMLSchema#string>	"1.00794"^^<http://www.w3.org/2001/XMLSchema#float>

½ jointure

Exemple : Nom et couleur des atomes ?

```
PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>
SELECT ?name ?color
WHERE {
    ?x a :Element ;
        :name ?name .
    ?x :color ?color.
}
ORDER BY ?color
```



**Beaucoup d'atomes ne figurent pas
dans le résultat !!**

?????

½ jointure : OPTIONAL

Solution : **Nom et couleur des atomes ?**

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

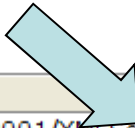
SELECT ?name ?color

WHERE {

 ?x a :Element ; :name ?name .

OPTIONAL { ?x :color ?color. }

} ORDER BY ?color



name	color
"ununseptium"^^<http://www.w3.org/2001/XMLSchema#string>	
"ununtrium"^^<http://www.w3.org/2001/XMLSchema#string>	
"ununpentium"^^<http://www.w3.org/2001/XMLSchema#string>	
"boron"^^<http://www.w3.org/2001/XMLSchema#string>	"black"^^<http://www.w3.org/2001/XMLSchema#string>
"osmium"^^<http://www.w3.org/2001/XMLSchema#string>	"bluish grey"^^<http://www.w3.org/2001/XMLSchema#string>
"zinc"^^<http://www.w3.org/2001/XMLSchema#string>	"bluish pale grey"^^<http://www.w3.org/2001/XMLSchema#string>
"lead"^^<http://www.w3.org/2001/XMLSchema#string>	"bluish white"^^<http://www.w3.org/2001/XMLSchema#string>
"neon"^^<http://www.w3.org/2001/XMLSchema#string>	"colourless"^^<http://www.w3.org/2001/XMLSchema#string>

Tous les atomes n'ont pas une couleur ou leur couleur n'est pas nécessairement renseignée. La variable ?color est dite **unbound** (! bound) dans ce cas (elle n'a pas de valeur).

Avec **OPTIONAL**, si le patron de graphe n'est pas trouvé, la requête complète n'est pas mise en échec (le patron correspond à une partie optionnelle du modèle de graphe recherché)

Patrons alternatifs : UNION

Exemple : Les atomes du groupe 13 et/ou le symbole de l'aluminium ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

SELECT ?e ?s

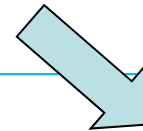
WHERE {

{ ?e :group :group_13. }

UNION

{ :Al :symbol ?s . }

}



e	s
Al	
Tl	
Ga	
B	
In	
Uut	

"Al"^^<http://www

Un patron de triplets ou de graphes alternatifs est un ensemble de patrons de graphes liés par une clause **UNION**, permettant que les solutions trouvées pour ces différents patrons, soient retournées

Assignation de variables : BIND

Exemple : Combien de neutrons possède chaque élément ?

```
SELECT ?element ?protons ?neutrons
WHERE {
    [] a :Element ;
       :atomicNumber ?protons ;
       :atomicWeight ?weight ;
       :name ?element .
    BIND( round( ?weight ) - ?protons AS ?neutrons )
} ORDER BY ?protons
```

BIND permet d'assigner une valeur à une variable dans le corps d'une requête

Entête d'une requête (projection)

Exemple : Combien de neutrons possède chaque élément ?

```
SELECT ?element ?protons ( round( ?weight ) - ?protons AS ?neutrons )  
WHERE {  
  [] a :Element ;  
  :atomicNumber ?protons ;  
  :atomicWeight ?weight ;  
  :name ?element .  
} ORDER BY ?protons
```

Le résultat d'une requête peut contenir des valeurs dérivées de constantes, d'appels de fonctions, or d'autres expressions.

Une colonne du résultat d'une requête peut être une expression quelconque.

Les expressions de projection doivent être entre parenthèses et être nommées via le mot clé **AS**

Agrégats et fonctions de groupe

Un groupement de triplets (SPARQL 1.1) est spécifié par **GROUP BY**
Les opérations sur les groupes sont : **COUNT, SUM, MIN, MAX, AVG, SAMPLE, GROUP_CONCAT**

La sélection de groupes s'exprime avec la clause : **HAVING**

Exemple : Groupes d'éléments comportant plus de 3 éléments chimiques ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

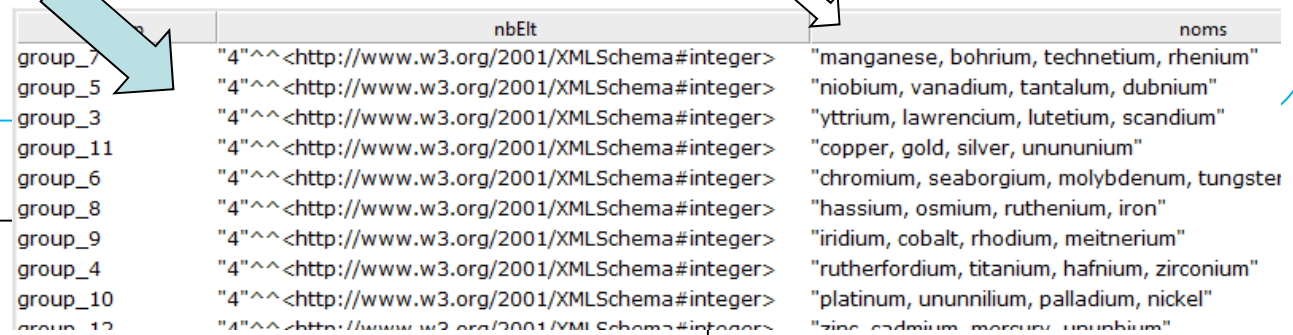
SELECT (?g AS ?group) (**count**(?e) AS ?nbElt) (**GROUP_CONCAT**(?n ; SEPARATOR =", ") AS ?noms)

WHERE { ?e :group ?g ; :name ?n }

GROUP BY (?g)

HAVING (**count**(?e) > 3)

ORDER BY ?nbElt



group	nbElt	noms
group_7	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"manganese, bohrium, technetium, rhenium"
group_5	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"niobium, vanadium, tantalum, dubnium"
group_3	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"yttrium, lawrencium, lutetium, scandium"
group_11	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"copper, gold, silver, ununium"
group_6	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"chromium, seaborgium, molybdenum, tungsten"
group_8	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"hassium, osmium, ruthenium, iron"
group_9	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"iridium, cobalt, rhodium, meitnerium"
group_4	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"rutherfordium, titanium, hafnium, zirconium"
group_10	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"platinum, ununnilium, palladium, nickel"
group_12	"4"^^<http://www.w3.org/2001/XMLSchema#integer>	"zinc, cadmium, mercury, ununbium"

Le dataset interrogé :

:Group_13 a :Group .

:Group_16 a :Group .

:Element a owl:Class.

:Al a :Element ; :symbol "Al" ; :name "aluminium" ; :group :Group_13.

:Se a :Element ; :symbol "Se" ; :name "selenium" ; :group :Group_16.

Les sous-requêtes

Une sous-requête (SPARQL 1.1) est imbriquée dans une requête principale et est exécutée en premier. Son résultat permet de conditionner l'exécution de la requête principale.

Exemple : Quel groupe d'atomes contient l'atome le moins lourd, et quel est le poids de cet atome ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

SELECT ?g ?minWeight

WHERE {

 ?e2 :group ?g ; :atomicWeight ?minWeight.


 { SELECT (MIN(?w) AS ?minWeight)

 WHERE {

 ?e :atomicWeight ?w .

 } }

}



g	minWeight
<http://www.daml.org/2003/01/periodictable/PeriodicTable#group_1>	"1.00794" ^^^<http://www.w3.org/200

Les chemins de propriétés



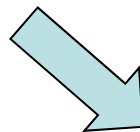
Les chemins de propriétés sont définis dans SPARQL 1.1 ; ils permettent la simplification de l'écriture de patrons de graphes

Les opérateurs pour décrire des chemins de propriétés dans un patron de graphe sont :

- L'alternative : **|** Ex.: `?x :name|rdfs:label ?y.`
- La séquence : **/** Ex.: `?x :group/rdfs:label ?y.`
- Le chemin inverse (objet **→** sujet) : **^** Ex.: `?x ^rdfs:label ?y.`
- Un chemin de longueur arbitraire : **+** Ex.: `?x rdfs:SubClassOf+ ?y`
- La négation de chemin : **!** Ex.: `?x !rdfs:label ?y.`
- Une application :
 - Les ressources et types inférés : **`rdf:type/rdf:subClassOf*`**

Exemple :

```
SELECT ?x WHERE {  
    ?x ^rdfs:label ?y.  
}
```



```
x  
"aluminium"@fr  
"argent"@fr  
"actinium"@fr  
"argon"@fr
```


La requête ASK

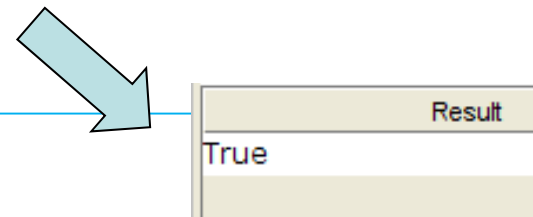
ASK retourne « true » ou « false » selon que le modèle de graphe a ou n'a pas de correspondances dans le dataset RDF interrogé.

- Même corps de requête que le SELECT

Exemple : L'atome de cuivre est-il plus lourd que l'atome d'aluminium ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

```
ASK {  
    :Al :atomicWeight ?w1.  
    :Cu :atomicWeight ?w2.  
    FILTER( ?w2 > ?w1 ).  
}
```



NB : La clause WHERE est optionnelle dans les requêtes SELECT

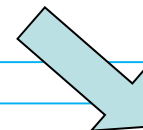
Découverte d'une ressource : DESCRIBE

DESCRIBE retourne toutes les informations RDF sur les ressources demandées, i.e. les triplets dont elles sont le sujet ou l'objet.
Même corps de requête que le SELECT ; le corps de requête est facultatif

Exemple : Informations sur l'atome d'aluminium ?

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

DESCRIBE :Al



Peut être équivalent à :

PREFIX : <http://www.daml.org/2003/01/

SELECT ?s ?p ?o

WHERE {

 {:Al ?p ?o . }

UNION

 { ?s ?p :Al }

}

Subject	Predicate	Object
Al	type	NamedIndividual
Al	period	period_3
Al	type	NamedIndividual
Al	block	p-block
Al	type	NamedIndividual
Al	atomicNumber	"13"^^<http://www
Al	type	NamedIndividual
Al	symbol	"Al"^^<http://ww
Al	type	NamedIndividual
group_13	element	Al
period_3	element	Al

La requête CONSTRUCT

CONSTRUCT retourne le triplet ou graphe RDF spécifié par le patron de graphe décrit dans {}

- Même corps de requête que le SELECT
- Permet de faire des transformations entre vocabulaires

Exemple :

PREFIX monVoc: <http://ex/monVocabulaire#>

PREFIX : <http://www.daml.org/2003/01/periodictable/PeriodicTable#>

CONSTRUCT {

 ?y monVoc:plusGrandQue ?x

}

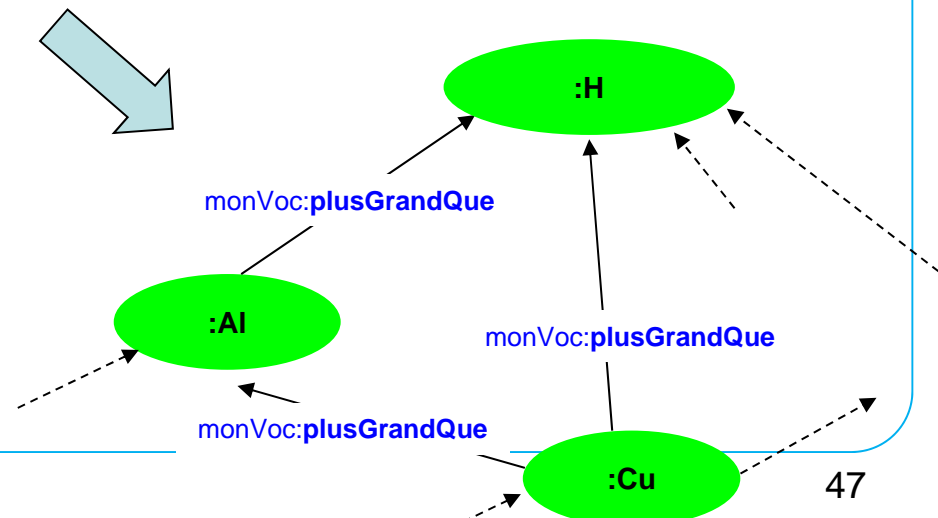
WHERE {

 ?x :atomicNumber ?w1.

 ?y :atomicNumber ?w2.

 filter (?w2 > ?w1)

}



Sérialisation du résultat d'une requête

Différents formats possibles :

- XML. La table résultant de la requête est représentée en XML
- JSON. Le résultat peut être directement exploité dans une application web.
- CSV/TSV. Le résultat peut être directement exploité un tableur
- RDF. Le résultat est constitué de triplets RDF qui peuvent être sérialisés de différentes façons (RDF/XML, N-Triples, Turtle, etc.)
- HTML. Une transformation XSL d'une table en XML (pour la lisibilité).

Exemple : Résultat XML d'une requête SELECT

```
<sparql xmlns='http://www.w3.org/2005/sparql-results#' xmlns:rd='http://www.w3.org/1999/02/22-rdf-syntax-ns#' >
<head> <variable name='number' /> <variable name='symbol' /> </head>
<results ordered='false' distinct='false' >
  <result>
    <binding name='number'><literal datatype='http://www.w3.org/2001/XMLSchema#integer'>34</literal></binding>
    <binding name='symbol'><literal datatype='http://www.w3.org/2001/XMLSchema#string'>Se</literal></binding>
  </result>
  <result>
    <binding name='number'><literal datatype='http://www.w3.org/2001/XMLSchema#integer'>20</literal></binding>
    <binding name='symbol'><literal datatype='http://www.w3.org/2001/XMLSchema#string'>Ca</literal></binding>
  </result> ...
</results>
</sparql>
```

Les variables retournées

Un « result » par ligne retournée

Mise en œuvre pratique

1. **Résoudre une enquête avec SPARQLUEDO** (Développée par Camille PRADEL, ancien doctorant de l'équipe MELODI de l'IRIT) :

<http://swip.univ-tlse2.fr/tpsparql/sparqluedo.html>

2. **Interrogation du endpoint Dbpédia :**

https://www.irit.fr/~Catherine.Comparot/sparql/TP_SPARQL_DBpedia.pdf