Données Semi-Structurées:

RDF

Enseignant: *Dario COLAZZO*
Chargée de TD/TP: *Beatrice NAPOLITANO*

08 Février 2021

1 Avant de commencer

Un vocabulaire RDF est un modèle de données comportant des classes, propriétés et relations qui peut être utilisé pour décrire vos données et métadonnées. C'est-à-dire, un vocabulaire RDF est un ensembles de termes utilisés pour décrire des choses. Un terme est soit une **classe**, soit une propriété.

— Propriétés de type d'objet (les **relations**) ;
— Propriétés de type de données (les **attributs**).

Classe. Une construction qui représente des choses dans le monde réel et/ou des informations, par exemple une personne, une organisation, un concept tel que "santé" ou "liberté".
Relation. Un lien entre deux classes ; par exemple le lien entre un document et l'organisation qui l'a publié (c-à-d organisation *publie* document). En RDF, les relations sont codées comme des propriétés de type d'objet.
Propriété. Une caractéristique d'une classe dans une dimension particulière, comme le nom légal d'une organisation ou la date et l'heure où une observation a été faite.

Les schémas et les vocabulaires RDF comprennent souvent des termes qui sont très génériques. En créant des sous-classes et des relations de sous-propriété, les systèmes qui comprennent la super propriété ou super classe peuvent être capables d'interpréter les données, même si les termes plus spécifiques sont inconnus.

1.0.1 Vocabulaires RDF existants

Une ressource utile pour trouver les vocabulaires RDF existants est le Linked Open Vocabularies repository [LOV], mais plusieurs autres services pour le même usage existent.

Nous utiliserons les vocabulaires de base :

— `RDF URI http://www.w3.org/1999/02/22-rdf-syntax-ns#`
RDF a ses propres propriétés de base et les termes de ce vocabulaire apparaissent dans presque toutes les documents RDF. Les principales propriétés comprennent rdf :type (qui peut être simplement écrit comme "a" dans Turtle). La classe principale est la classe Resource - la super classe pour toutes les classes.
— `RDFS URI http://www.w3.org/2000/01/rdf-schema#`
Le schéma du W3C pour la description des schémas. Les termes de ce vocabulaire sont utilisés pour définir des classes, des propriétés, des sous-classes, des sous-propriétés, etc.
— `XML Schema URI http://www.w3.org/2001/XMLSchema#`
Plutôt que d'inventer ses propres types de données, RDF réutilise ceux définis dans XML Schema. Les principaux types sont les suivants : date, dateTime, anyURI, boolean, integer, float.

Lorsque de nouveaux termes sont nécessaires, vous pouvez les créer suivant les meilleures pratiques communément admises :

— Les Classes commencent avec une majuscule et sont toujours au singulier.
— Les propriétés commencent par une lettre en minuscule.
— Les propriétés des objets devraient être des verbes.
— Les propriétés de type de données doivent être des noms.
— Utilisez le« Camel Case » si un terme a plus d'un mot.

1.1 Schema RDF en XML

Exemple 1 *Schema RDF (en XML) :*

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE rdf [
        <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema">
]>
<rdf:RDF
xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdfs:Class rdf:ID="Animal">
<rdfs:label xml:lang="en">animal</rdfs:label>
<rdfs:comment xml:lang="en">a living organism characterized by voluntary
        movement.</rdfs:comment>
<rdfs:label xml:lang="fr">animal</rdfs:label>
<rdfs:comment xml:lang="fr">etre vivant doue de sensibilite de
        mobilite.</rdfs:comment>
</rdfs:Class>

</rdf:RDF>
```

Pour alléger l'écriture des documents XML en évitant de mettre un préfixe aux éléments les plus fréquents, on peut définir un espace de noms par défaut.

```
xmlns = "http://www.w3.org/2000/01/rdf-schema#">
```

Voir plus : `http://www-sop.inria.fr/members/Philippe.Poulard/pdf/01c-xml-namespaces.pdf`.

On peut aussi résoudre les ID RDF en en définissant xml :base :

```
xml:base = "http://humans/">
```

Les URI sont purement déclaratives, il n'y a pas nécessairement quelque chose à l'adresse indiquée.

Exemple 2 *Schema RDF (en XML) :*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf [
<!ENTITY humans   "http://humans/">
<!ENTITY rdfs     "http://www.w3.org/2000/01/rdf-schema">
]>

<rdf:RDF
xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="&rdfs;#"
xmlns     ="&rdfs;#"
xml:base  ="&humans;" >
```

```
<Class rdf:ID="Animal">
<label xml:lang="en">animal</label>
<comment xml:lang="en">a living organism characterized by voluntary movement.</comment
<label xml:lang="fr">animal</label>
<comment xml:lang="fr">etre vivant doue de sensibilite de mobilite.</comment>
</Class>

</rdf:RDF>
```

Le validateur RDF en ligne fourni par W3C (`https://www.w3.org/RDF/Validator/`) analyse votre document RDF, vérifie votre syntaxe et génère des vues tabulaires et graphiques de votre document RDF.

Testez les deux exemples en utilisant le validateur RDF.

1.2 Schema RDF en Turtle

Turtle est une syntaxe d'un langage qui permet une sérialisation non-XML des modèles RDF. Documentation : `https://www.w3.org/2007/02/turtle/primer/`.

Voici les principales règles pour lire ou écrire des triplets en Turtle :

- Un triplet simple est une séquence de termes (sujet, prédicat, objet) séparés par des espaces et terminée par un point (.).
- Il y a trois types de termes :
  - URI écrit entre < et > ; un URI peut aussi s'exprimer par un nom qualifié (QNAME), i.e. précédé par un préfixe défini en utilisant les définitions @prefix, suivi d'un " :" et du reste de l'URI. L'URI complet s'obtient en remplaçant le préfixe par le contenu de sa définition. Un préfixe peut être vide (mais on garde alors le " :") ;
  - littéral entre guillemets ou accolades, possiblement suivie d'une définition de type suivi d'un nom de type souvent de la forme xsd: ... pour les types standards de XML Schema ;
  - noeud vide indiqué par _:nodeId ou un groupe entre crochets.
- Un préfixe p est défini par @prefix p : URI. La virgule répète le sujet et le prédicat de triplets qui ne varient que par l'objet. p.e. $ex : a \quad ex : b \quad ex : c, \quad ex : d. \Rightarrow ex : a \quad ex : b \quad ex : c. \quad ex : a \quad ex : b \quad ex : d.$
- Le point-virgule répète le sujet de triplets qui ne varient que par le prédicat et l'objet. p.e. $ex : a \quad ex : b \quad ex : c; \quad ex : d \quad ex : e. \Rightarrow ex : a \quad ex : b \quad ex : c. \quad ex : a \quad ex : d \quad ex : e.$
- Les parenthèses regroupent les éléments d'une collection.

Le schéma en notation Turtle correspondant au schéma de l'Exemple 2 est le suivant :

Exemple 3 *Schema Turtle :*

```
@prefix : <http://www.w3.org/2000/01/rdf-schema#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

:Animal  a  :Class;
:label "animal"@en,"animal"@fr;
:comment "a␣living␣organism␣characterized␣by␣voluntary␣movement."@en;
:comment "etre␣vivant␣doue␣de␣sensibilite␣de␣mobilite."@fr.
```

Le datatype *xsd :integer* est identifié par son URIref (l'URIref complet étant

http ://www.w3.org/2001/XMLSchema#integer). Cet URIref peut être utilisé sans indiquer explicitement dans le schéma qu'il identifie un datatype. Cependant, il est utile d'indiquer explicitement qu'un URIref identifie un datatype. Cela peut être fait en utilisant la classe RDF Schema *rdfs :Datatype* :

```
xsd:integer rdf:type rdfs:Datatype .
```

Ici un validateur de syntaxe pour Turtle `http://ttl.summerofcode.be/`.

Le validateur en ligne `https://www.w3.org/2015/03/ShExValidata/` analyse votre document RDF par rapport à le schéma.

1.3 Exemples de documents RDF

Exemple 4 *Document RDF en XML :*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
<!ENTITY humans   "http://humans/">
<!ENTITY xsd      "http://www.w3.org/2001/XMLSchema#"> ]>

<rdf:RDF
xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd ="&xsd;"
xmlns     ="&humans;#"
xml:base  ="&humans;-instances" >

<Man rdf:ID="Harry">
<name>Harry</name>
<hasChild rdf:resource="#John"/>
<hasSpouse rdf:resource="#Sophie"/>
</Man>
```

1.4 Document RDF en Turtle

Exemple 5 *Document RDF en Turtle :*

```
@prefix : <http://www.w3.org/2000/01/rdf-schema#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@base <http://humans/>.

<Harry>
        a :Man;
        :hasChild <John>;
        :hasSpouse <Sophie>.
```

2 Exercices

Exercice 1 *Ecrire le schéma RDF en XML (.rdfs) pour représenter les faits suivants :*

- *un animal peut être mâle ou femelle ;*
- *une personne est un animal ;*
- *une personne mâle est un homme, une personne femelle est une femme ;*
- *un animal a des ancêtres (y compris les parents, le père (mâle), la mère (femelle)), des frères et sœurs, des enfants ;*
- *une personne a aussi des partenaires et des amis ;*
- *une personne peut être un professeur ou un chercheur ;*
- *une personne a un nom et un âge.*

Exercice 2 *Ecrire le document RDF en XML (.rdf) pour représenter les faits suivants :*

- *Harry est un homme marié à Sophie avec un enfant John ;*

— *John a 37 ans ;*
— *Mark a 14 ans et est le fils de John ;*
— *Eve est une professeur, marié à David et amie avec Alice ;*
— *Alice est amie avec John ;*
— *David est un chercheur, ami avec Gaston ;*
— *Flora a 95 ans, est la femme de Gaston et la mère de Pierre ;*
— *Jennifer est mariée à John ;*
— *Gaston a 102 ans et est le père de Pierre et de Jack ;*
— *Pierre a 62 ans ;*
— *Jack est le père de Harry.*

Exercice 3 *Refaire l'Exercice 1 en notation Turtle (.ttl).*

Exercice 4 *Refaire l'Exercice 2 en notation Turtle (.ttl).*