INFO-H509: XML & Web Technologies
Semantic Web Exercises
SOLUTIONS

Professeur: Stijn Vansummeren
Assistant: Michael Waumans

2014–2015

---

1 RDF

Exercice 1.1

1. Voir les fichiers inclus pour les tracés de graphes.
2. Voir les fichiers inclus pour les tracés de graphes.
3. Voir les fichiers inclus pour les tracés de graphes.
4. Les triplets, écrit en syntaxe Turtle pour des raisons de lisibilité, sont les suivants :

```
@prefix ulb: <http://code.ulb.ac.be/example/terms/> .
@prefix infoh509: <http://code.ulb.ac.be/example/courses/infoh509/> .
@prefix catalogue: <http://code.ulb.ac.be/example/catalogue/> .
@prefix staff: <http://code.ulb.ac.be/example/staff#> .
@prefix infoh509media: <http://cs.ulb.ac.be/public/_media/teaching/infoh509/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ev: <http://purl.org/rss/1.0/modules/event/>

infoh509:lecture2           ev:location             "UA4.218"                 .
infoh509:lecture2           ev:startDate            "2009-02-12T10:00:00"^^xs:dateTime  .
infoh509:lecture2           ev:endDate              "2009-02-12T12:00:00"^^xs:dateTime  .
infoh509:lecture2           rdf:type                ulb:Lecture               .
infoh509:lecture2           ulb:teaching_material   infoh509media:2-xpath.pdf .
infoh509media:2-xpath.pdf   dc:title                "Cours 2: XPath"          .
infoh509media:2-xpath.pdf   dc:creator              staff:svansumm            .
```

Exercice 1.2

Le fichier `catalog.rdf` est modifié comme suit. Le cours INFO-H-200 est ajouté:

```
  <rdf:Description rdf:about="infoh200">
        <dc:title>Object Oriented Programming</dc:title>
        <ulb:lecturer rdf:resource="&staff;ezimanyi" />
        <ulb:assistant rdf:resource="&staff;boverhae" />
        <ulb:prerequisite rdf:resoucre="&ulb;infoh100"/>
    </rdf:Description>
```

Et le pré-requis est ajouté au cours INFO-H-415:

```
<rdf:Description rdf:about="infoh415">
  ...
  <ulb:prerequisite rdf:resoucre="&ulb;infoh303"/>
</rdf:Description>
```

Exercice 1.3

```
@prefix myconcepts: <http://www.example.org/johndoe/>              .
@prefix foaf:       <http://xmlns.com/foaf/0.1/>                   .
@prefix catalogue:  <http://code.ulb.ac.be/example/catalogue/>     .

myconcepts:me       a                          foaf:Person         .
myconcepts:me       foaf:name                  "John Doe"          .
myconcepts:me       foaf:mbox                  "jd@gmail.com       .
myconcepts:me       myconcepts:has_course      catalogue:infoh509  .
```

2 RDF Schema

Exercice 2.1

Les besoins suivants sont ajouté au fichier `inference.ttl` afin de spécifier que tous les membres du personnel (`ulb:Faculty`) sont des personnes (`foaf:Person`).

```
ulb:Faculty         rdfs:subClassOf     foaf:Person  .
```

Exercice 2.2

```
ulb:prerequisite    rdfs:range          ulb:Course   .
ulb:prerequisite    rdfs:domain         ulb:Course   .
```

Supplementary exercise: De la même manière, en continuant d'utiliser `rdfs:range` et `rdfs:domain`, ajouter les règles qui décrivent les propriétés `lecturer` et `assistant` de manière plus détaillée.

```
ulb:lecturer        rdfs:range          ulb:Faculty  .
ulb:lecturer        rdfs:domain         ulb:Course   .
ulb:assistant       rdfs:range          ulb:Faculty  .
ulb:assistant       rdfs:domain         ulb:Course   .
```

Exercice 2.3

```
ulb:workHomepage    rdfs:subPropertyOf  foaf:homepage  .
```

Notez que cette propriété est du type `rdf:Property`, mais aussi du type `rdfs:Resource`. Souvenez vous que RDFS (de base) vient avec un certain nombre d'axiomes standard qui spécifient, parmis d'autres, que le domaine et la portée de `subPropertyOf` sont `rdf:Property` et que, `rdf:Property` est une sous-classe de `rdfs:Resource`.

Dans le fichier `staff.rdf`, la propriété 'homepage' peut donc être remplacée par `workHomePage` de la manière suivante.

```
<ulb:Professor rdf:ID="svansumm">
    ...
    <ulb:workHomepage
        rdf:resource="http://code.ulb.ac.be/code.people.php?id=992"/>
</ulb:Professor>
<ulb:Professor rdf:ID="ezimanyi">
    ...
    <ulb:workHomepage rdf:resource="http://cs.ulb.ac.be/members/esteban/" />
</ulb:Professor>
```

3 OWL

Exercice 3.1

Souvenez vous que OWL DL définit les propriétés caractéristiques suivantes.

1. `owl:TransitiveProperty`
2. `owl:SymmetricProperty`
3. `owl:FunctionalProperty`
4. `owl:InverseFunctionalProperty`

Pour chacune des propriété suivantes, listez les propriétés caractéristiques qui pourraient s'appliquer.

1. Le prérequis d'un cours (`ulb:prerequisite`)
2. Le numéro d'étudiant d'un étudiant
3. La date de naissance
4. owl:sameAs
5. owl:inverseOf

Supplementary exercise: Complétez la description de `ulb:prerequisite` dans le fichier `inference.ttl`. Vérifiez ensuite l'effet de cette modification en utilisant l'utilitaire `inferencetool.jar`.

	`Transitive`	`Symmetric`	`Functional`	`InverseFunctional`
Prérequis d'un cours	O			
Numéro d'un étudiant			O	O
Date de naissance			O	
owl:sameAs	O	O		
owl:inverseOf		O		

Transitive Si l'on a besoin d'avoir suivi le cours c_1 afin de pouvoir suivre le cours c_2 et, si l'on a besoin de suivre le cours c_2 afin de pouvoir suivre le cours c_3, alors, il est normal de penser que l'on a besoin de suivre c_1 avant de pouvoir suivre c_3. De manière analogue, si a est identique à b et b est identique à c, alors a est identique à c. `prerequisite` et `sameAs` sont donc transitives.

Symétrique Si a est identique à b, alors forcément, b est identique à a. De façon similaire, si a est l'inverse de b, alors b est l'inverse de a. `sameAs` et `inverseOf` sont donc symétriques.

Fonctionelle Une personne a exactement une date de naissance. De plus, une personne à au plus un numéro d'étudiant. Ces propriétés sont donc fonctionelles.

Fonctionelle inverse Un numéro d'étudiant n'est jamais attribué à plus de une personne. Donc, ceci est 'functionally inverse'.

Note Notez néanmoins que, puisque cette propriété lie un individu à un élément de donnée litéral, c'est un `owl:DataObjectProperty`. De ce fait, si nous spécifions que c'est fonctionellement inverse, nous avons une ontologie qui est OWL Full et non OWL DL. Il est donc possible que nous ne soyons pas capable de raisonner autamiquement avec.

Supplementary exercise:

La propriété `prerequisite` est définie comme étant transitive comme suit :

```
ulb:prerequisite    a                   owl:TransitiveProperty  .
```

Exercice 3.2

```
ulb:teaches       owl:inverseOf       ulb:lecturer .
ulb:teaches       rdfs:domain         ulb:Faculty .
ulb:teaches       rdfs:range          ulb:Course .
```

Exercice 3.3

```
staff:fpicalau      owl:sameAs       <http://my.opera.com/fpicalausa/xml/foaf#me> .
```

Exercice 3.4

```
@prefix terms: <http://code.ulb.ac.be/example/terms/> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .

terms:PizzaTopping a owl:Class .

terms:Vegetable a             owl:Class ;
rdfs:subClassOf terms:PizzaTopping .

terms:Pizza     a owl:Class ;
owl:disjointWith terms:PizzaTopping .

terms:aubergine a   terms:Vegetable .

terms:hasTopping a  owl:ObjectProperty ;
 rdfs:domain  terms:Pizza ;
 rdfs:range  terms:PizzaTopping .

terms:noMeatPizza a   owl:Class ;
  rdfs:subClassOf terms:Pizza .

terms:noFishPizza a   owl:Class ;
  rdfs:subClassOf terms:Pizza .

terms:VegPizza    a   owl:Class ;
  owl:intersectionOf (terms:noMeatPizza terms:noFishPizza) .

terms:hasTopping rdfs:subPropertyOf terms:hasIngredient .
```

```
terms:boring a terms:noMeatPizza .
terms:boring a terms:noFishPizza .
```

- **Réponse** Puisque nous avons déclaré `PizzaTopping` et `Pizza` comme étant disjoints, le document est devenu inconsistant. (i.e., il ne possède pas de modèle). Nous devrions retirer l'individu d'une des deux classes pour que cela redevienne consistant.
- **Réponse:** Puisque nous avons déclaré la classe `VegPizza` comme constituée d'éléments des classes `NoMeatPizza` et `NoFishPizza`, l'outil infère convenablement que l'individu appartient à la classe `VegPizza`.

Exercice 3.5

1. Chaque pizza a `tomato` comme garniture.

```
terms:Pizza rdfs:subClassOf [
   a               owl:Restriction;
   owl:onProperty  terms:hasTopping;
   owl:hasValue    terms:tomato
] .
```

2. Toutes les pizzas de la classe `PizzaMargarita` ont exactement `tomato` et `cheese` comme garniture. Nous avons besoin de deux déclarations :

```
terms:PizzaMargarita rdfs:subClassOf [
   a               owl:Restriction;
   owl:onProperty  terms:hasTopping;
   owl:allValuesFrom [
                       a owl:Class;
                       owl:oneOf (terms:tomato terms:cheese)
                      ]
] .

terms:PizzaMargarita rdfs:subClassOf [
   a                    owl:Class ;
   owl:intersectionOf
     ( [a owl:Restriction; owl:onProperty terms:hasTopping; owl:hasValue terms:tomato]
       [a owl:Restriction; owl:onProperty terms:hasTopping; owl:hasValue terms:cheese] )
] .
```

La première spécifie que toutes les valeurs dans la portée de `hasTopping` sont soit `terms:tomato` soit `terms:cheese`. La seconde spécifie que `hasTopping` devrait exister avec ces valeurs.

Note Notez que le fichier 'infertools.jar' ne raisonne pas avec `owl:OneOf`. De ce fait, l'outil ne sera pas capable de conclure des triplets suivants que l'ontologie est devenue inconsistante. (parce que `hasTopping` a une valuer qui est différente de 'tomato' ou 'cheese'). Il y a des outils plus puissants à disposition (comme, pellet `http://clarkparsia.com/pellet/download/pellet-2.3.1` qui permettent de tirer ce genre de conclusions.

```
terms:pm a terms:PizzaMargarita .
terms:pm terms:hasTopping terms:ex1 .
terms:ex1 owl:differentFrom terms:tomato, terms:cheese .
```

4 SPARQL

Exercice 4.1

1. Récupérez les URIs de tous les cours

```
PREFIX ulb: <http://code.ulb.ac.be/example/terms/>
SELECT ?course
WHERE {
  ?course       a       ulb:Course  .
}
```

2. Récupérez, pour chaque cours, son titre et le nom du professeur.

```
PREFIX ulb: <http://code.ulb.ac.be/example/terms/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?title   ?name
WHERE {
  ?course       a                   ulb:Course      .
  ?course       dc:title            ?title          .
  ?course       ulb:lecturer        ?lecturer       .
  ?lecturer     foaf:name           ?name           .
}
```

3. Le nom de tous les professeurs qui enseignent un cours tel que ce professeur enseigne aussi le prérequis de ce cours.

```
PREFIX ulb: <http://code.ulb.ac.be/example/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  ?course       a                   ulb:Course      .
  ?course       ulb:lecturer        ?lecturer       .
  ?course       ulb:prerequisite    ?pre            .
  ?pre          ulb:lecturer        ?lecturer       .
  ?lecturer     foaf:name           ?name           .
}
```

4. Toutes les personnes (`foaf:Person`) et leur page personnelle, si cette information est disponible (i.e., récupérez seulement la personne si aucune page personnelle n'est disponible.)

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?homepage
WHERE {
  ?person       a                   foaf:Person     .
  ?person       foaf:name           ?name           .
  OPTIONAL {
    ?person     foaf:homepage       ?homepage
  }
}
```

5. Le titre de tous les cours qui ont été organisés en *UA4.218*.

```
PREFIX ulb:     <http://code.ulb.ac.be/example/terms/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
PREFIX ev:      <http://purl.org/rss/1.0/modules/event/>
SELECT DISTINCT ?title
WHERE {
  ?course       a                   ulb:Course      .
  ?course       dc:title            ?title          .
  ?course       ulb:schedule        ?schedule       .
  ?schedule     ?test               ?item           .
  ?item         ev:location         "UA4.218"       .
}
```

6. **Supplementary exercise:** Toutes les personnes qui connaissent quelqu'un qui connait M.Vansummeren.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX staff: <http://code.ulb.ac.be/example/staff#>
SELECT DISTINCT ?name
WHERE {
  ?p2               foaf:knows          ?p1             .
  staff:svansumm    foaf:knows          ?p1             .
  ?p2               foaf:name           ?name           .
  FILTER(!sameTerm(?p2, staff:svansumm))
}
```

Le filtre `!sameTerm` exprime que la personne `p2` (qui connait quelqu'un qui connait prof. Vansummeren) n'est pas prof. Vansummeren lui-même.