INF6410 - Ontologies et web sémantique
Contrôle périodique - Automne 2017

Professeur: Michel Gagnon
École Polytechnique de Montréal

20 novembre 2017

1 RDF et SPARQL (5 points)

Soit la situation suivante :

> *La cathédrale d'Amos se situe en Abitibi et a été conçue par l'architecte Aristide Beaugrand-Champagne (1876-1950). La construction débute en 1922 et est complétée en 1925. En 1954, elle est pourvue d'un orgue de Casavant et Frères.*

a) (2 points) Représentez cette situation en RDF, en notation Turtle. Assurez-vous que votre représentation soit valide et utilisez adéquatement les URI, les noeuds vides et les littéraux. Assurez-vous aussi de représenter de manière exhaustive tout ce qui est exprimé dans le texte et, de manière raisonnable, toute information supplémentaire qu'il faudrait aussi retrouver pour lier le contenu au Web sémantique. Vous pouvez supposer que certains URI existent déjà dans certaines bases de connaissances, comme DBpedia. Pour vous aider, vous pouvez aussi considérer que votre base de connaissances sera utilisée pour répondre aux questions de l'exercice qui suit.

```
@prefix : <http:www.exemple.org/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dbpedia: <http://dbpedia.org/resource/> .

:Cathedrale rdfs:subClassOf :Eglise .
:Eglise rdfs:subClassOf :LieuDeCulte .
:cathedraleAmos a :Cathedrale ;
                rdfs:label "Cathédrale d'Amos" ;
                :localisation :Abitibi ;
                :débutConstruction  "1922"^^xsd:gYear ;
                :finConstruction "1923"^^xsd:gYear ;
                :mobilier [ a :Orgue ;
                            :arrivee "1954"^^gYear ;
                            :constructeur dbpedia:Casavant ] .
 :ABC :annéeNaissance "1875"^^xsd:gYear ;
     foaf:name  "Aristide Beaugrand-Champagne" ;
     :annéeDécès "1950"^^xsd:gYear ;
     :concepteurDe :cathedraleAmos .
```

b) (2 points) En supposant une base de connaissances RDF selon le modèle que vous avez utilisé à la questions a), représentez en SPARQL chacune des requêtes suivantes (supposez que votre engin SPARQL ne permet de faire aucune inférence) :

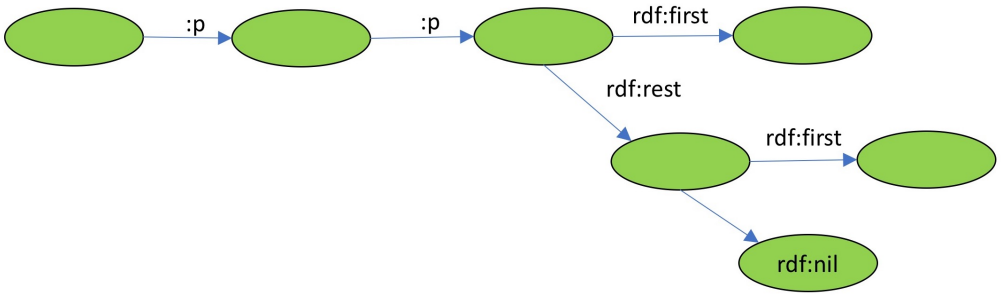
1. À quels endroits en Abitibi (pas nécessairement des églises) des orgues de Casavant et Frères ont-ils été installés depuis 1950 ?
2. Parmi les bâtiments conçus par Aristide Beaugrand-Champagne, combien sont des lieux de culte ?
3. Combien de temps a-t-il fallu pour constuire la cathédrale d'Amos ?
4. Parmi les bâtiments conçus par Aristide Beaugrand-Champagne, combien y en a-t-il pour chaque type de bâtiment, c'est-à-dire combien d'églises, combien d'écoles, etc ?

```
PREFIX : <http:www.exemple.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbpedia: <http://dbpedia.org/resource/>

SELECT ?x
WHERE
{
  ?x :mobilier [ a :Orgue ; :constructeur dbpedia:Casavant ]
}

SELECT  (COUNT(?z) as ?NombreLieuxDeCulte)
WHERE
{
   :ABC :concepteurDe ?z .
   ?z a ?t  .
   ?t rdfs:subClassOf+ :LieuDeCulte
}

SELECT  ?duree
WHERE
{
   :cathedraleAmos
    :débutConstruction ?d ;
    :finConstruction ?f .
```

```
 BIND(YEAR(?f) - YEAR(?d) AS ?duree)
}

SELECT (COUNT(?b) AS ?Bâtiment) ?type
WHERE
{
   :ABC :concepteurDe ?b .
   ?b rdf:type ?type .
}
GROUP BY ?type
```

c) (1 point) Dessinez le graphe RDF suivant :

```
@prefix : <http://www.exemple.org/> .
[ :p [ :p ([] []) ] ] .
```



2 Sémantique de RDF (2 points)

a) Soit les deux graphes RDF suivants :

G1 : `:a :p (:b :c :d) .`

G2 : `:a :p (:b :c) .`

Peut-on dire que G1 implique logiquement G2 ? Justifiez votre réponse.

Non. Une liste en RDF est fermée. Dans G1 la liste DOIT contenir exatement trois élément, alors que la liste dans G2 doit en contenir exactement. Ainsi, si dans l'interprétation de G1 la liste contient les item X, Y et Z, la liste de G2 ne contiendre que X et Y.

b) Proposez une interpréation sémantique pour le graphe RDF suivant :

```
@prefix : <www.exemple.org/> .

:x :p [ a :C ; a :D ] .
```

IR = { X, Y, P, ClasseC, ClasseD, TYPE, . . . }
IS(:x) = X
IS(:p) = P
IS(rdf :type) = TYPE
IS(:C) = ClasseC
IS(:D) = ClasseD
IEXT(P) = { ⟨X,Y⟩, . . . }
IEXT(TYPE) = { ⟨Y,ClasseC⟩,⟨Y,ClasseD⟩, . . . }
IC(ClasseC) = IC(ClasseD) = { Y }

3 Ontologie en OWL (7 points)

a) (2 points) Écrivez en OWL l'ontologie suivante, qui sert à représenter les cours enseignés dans une université :

1. Les étudiants et les professeurs sont des personnes.
2. Les seuls personnes qui existent dans notre base de connaisances sont des professeurs et des étudiants.
3. Toute personne a un nom et une adresse courriel.
4. Les étudiants suivent des cours et les professeurs enseignent des cours.
5. Un cours a un code d'identification et un titre.
6. Un étudiant à temps plein doit suivre au moins 3 cours, sinon il est un étudiant à temps partiel.
7. Un étudiant ne peut pas suivre plus de 6 cours.

Assurez-vous de maximiser la capacité d'inférence avec votre ontologie.

```
Ontology(<http ://www.exemple.org/academique>
SubClassOf( :Cours ObjectIntersectionOf(
                        DataSomeValuesFrom( :code xsd :string)
                        DataSomeValuesFrom( :titre xsd :string)))
SubClassOf( :Etudiant ObjectIntersectionOf(
                         :Personne
                        ObjectSomeValuesFrom( :suit :Cours)))
DisjointClasses( :Etudiant :Professeur)
EquivalentClasses( :EtudiantTempsComplet ObjectIntersectionOf(
                         :Etudiant
                        ObjectMinCardinality(3 :suit :Cours)))
EquivalentClasses( :EtudiantTempsPartiel ObjectIntersectionOf(
                        ObjectComplementOf( :EtudiantTempsComplet))
SubClassOf( :Personne ObjectIntersectionOf(
                        DataSomeValuesFrom( :courriel xsd :string)
                        DataSomeValuesFrom( :nom xsd :string)))
DisjointUnion( :Personne :Etudiant :Professeur)
SubClassOf( :Professeur ObjectIntersectionOf(
                         :Personne
                        ObjectSomeValuesFrom( :enseigne :Cours))))
```

b) (2 points) Ajoutez à votre ontologie le concept de trimestre, qui respecte les deux dernières contraintes (entre 3 et 6 cours pour un étudiant à temps plein). Évidemment, vous devez représenter le fait que l'étudiant s'inscrit à plusieurs trimestres dans son parcours académique.

```
Ontology(<http ://www.exemple.org/michelgagnon/academique>
SubClassOf( :Cours ObjectIntersectionOf(
                    DataSomeValuesFrom( :code xsd :string)
                    DataSomeValuesFrom( :titre xsd :string)))
EquivalentClasses( :CoursTrimestre ObjectIntersectionOf(
                    ObjectMinCardinality(1 :contientCours :Cours)
                    ObjectMaxCardinality(6 :contientCours :Cours)
                    ObjectExactCardinality(1 :trimestre :Trimestre)))
SubClassOf( :Etudiant ObjectIntersectionOf(
                     :Personne
                    ObjectSomeValuesFrom( :cheminement :CoursTrimestre)
                    ObjectAllValuesFrom( :cheminement :CoursTrimestre)))
DisjointClasses( :Etudiant :Professeur)
EquivalentClasses( :CoursTrimestreTempsComplet ObjectIntersectionOf(
                     :CoursTrimestre
                    ObjectMinCardinality(3 :contientCours :Cours)))
EquivalentClasses( :CoursTrimestreTempsPartiel ObjectIntersectionOf(
                     :CoursTrimestre
                    ObjectComplementOf( :CoursTrimestreTempsComplet)))
SubClassOf( :Personne ObjectIntersectionOf(
                    DataSomeValuesFrom( :courriel xsd :string)
                    DataSomeValuesFrom( :nom xsd :string)))
DisjointUnion( :Personne :Etudiant :Professeur)
SubClassOf( :Professeur ObjectIntersectionOf(
                     :Professeur
                    ObjectSomeValuesFrom( :enseigne :Cours)))
```

c) (0,5 point) Représentez dans votre base de connaissances le fait que Pedro Silveira est inscrit aux cours INF1000, INF2000 et INF3000 à l'hiver 2017 et aux cours INF4000 et INF5000 à l'automne 2017. Notez qu'un raisonneur devrait automatiquement pouvoir déduire que Pedro était un étudiant à temps plein au

trimestre d'hiver 2017.

ClassAssertion(:Etudiant :PEDRO)
ClassAssertion(:Trimestre :A17)
ClassAssertion(:Trimestre :H17)
ClassAssertion(:Cours :INF1000)
ClassAssertion(:Cours :INF2000)
ClassAssertion(:Cours :INF3000)
ClassAssertion(:Cours :INF4000)
ClassAssertion(:Cours :INF5000)

ObjectPropertyAssertion(:cheminement :PEDRO :PEDRO_A17)
ObjectPropertyAssertion(:cheminement :PEDRO :PEDRO_H17)

ClassAssertion(:CoursTrimestre :COURS_PEDRO_A17)
ObjectPropertyAssertion(:contientCours :COURS_PEDRO_A17 :INF4000)
ObjectPropertyAssertion(:contientCours :COURS_PEDRO_A17 :INF5000)
ObjectPropertyAssertion(:trimestre :COURS_PEDRO_A17 :A17)

ClassAssertion(:CoursTrimestre :COURS_PEDRO_H17)
ObjectPropertyAssertion(:contientCours :COURS_PEDRO_H17 :INF1000)
ObjectPropertyAssertion(:contientCours :COURS_PEDRO_H17 :INF2000)
ObjectPropertyAssertion(:contientCours :COURS_PEDRO_H17 :INF3000)
ObjectPropertyAssertion(:trimestre :COURS_PEDRO_H17 :H17)

d) (1 point) Expliquez ce que devra nécessairement contenir la base de connaissances pour déduire explicitement qu'un étudiant est inscrit à temps partiel.

DifferentIndividuals(:INF1000 :INF2000 :INF3000 :INF4000 :INF5000)

e) (1,5 point) Identifiez une situation, dans le domaine de l'ontologie que vous venez de définir, qui nécessiterait l'utilisation de règles, et fournissez cette représentation.

Il faudrait utiliser une règle RDF-BLD pour déterminer si un professeur a déjà enseigné à un étudiant :

```
Forall ?prof ?etudiant (
:aEnseigné(?prof ?etudiant) :-
     And(:cheminement(?etudiant ?courstrimestre)
       :contientCours(?coursTrimestre ?cours)
       :enseigne(?prof ?cours)))
```

On pourrait aussi utiliser une règle RDF-PRD pour déduire qu'un étudiant est à temps partiel à un certain trimestre :

```
Forall ?Etudiant ?Trimestre (
If   And(:cheminement(?etudiant ?courstrimestre)
         Not(:CoursTrimestreTempsComplet(?courstrimestre)))
Then :tempsPartiel(?Etudiant ?Trimestre))
```

4 Inférence en logique descriptive (2 points)

Soit une base de connaissances utilisant l'ontologie suivante :

PersonneUnilingue $\equiv$ Personne $\sqcap \leq 1$ parleLangue
Polyglotte $\equiv$ Personne $\sqcap \neg$PersonneUnilingue
Professeur $\equiv$ Personne $\sqcap$ $\exists$enseigne.Universite $\sqcap$ parleLangue : ANGLAIS
Francophone $\equiv$ parleLangue : FRANCAIS
ANGLAIS $\neq$ FRANCAIS

Supposons maintenant que nous ajoutons ceci à la base de connaissances :

Francophone(MICHEL)
Professeur(MICHEL)
parleLangue(JOHN,ANGLAIS)

Dites ce qu'on peut déduire de cette ABox, et expliquez comment on peut faire ces déductions.

On peut déduire ceci :

a) Personne(MICHEL)
b) parleLangue(MICHEL, FRANCAIS)
c) parleLangue(MICHEL, ANGLAIS)
d) Polyglotte(MICHEL)

5 Relations non binaires en RDF (4 points)

Nous voulons représenter la situation suivante en RDF : *Jean a contacté Marie par téléphone alors qu'il a contacté Louis par courriel*. Nous voulons utiliser la même relation *contacter* pour les deux cas cités dans cet exemple.

a) (2 points) Montrez comment on peut faire cette représentation en utilisant chacune des quatre méhodes considérées pour la sérialisation en RDF de Wikidata.

Réification RDF :

```
[] a rdf:Statement ;
   rdf:subject :JEAN ;
   rdf:object  :MARIE ;
   rdf:predicate :contacter ;
   :moyen  :téléphone .

[] a rdf:Statement ;
   rdf:subject :JEAN ;
   rdf:object  :LOUIS ;
   rdf:predicate :contacter ;
   :moyen  :courriel .
```

Relation n-aire :

```
:JEAN :contacterS
   [  :contacterV  :MARIE ;
      :moyen  :téléphone ] ;
   [  :contacterV  :LOUIS ;
      :moyen  :courriel ] .
```

Singleton :

```
:contacter1 a :ContacterProp .
:contacter2 a :ContacterProp .

:JEAN :contacter1 :MARIE ;
      :contacter2 :LOUIS .

:contacter1 :moyen :téléphone .
:contacter2 :moyen :courriel.
```

Graphe nommé :

```
:g1 { :JEAN :contacter :MARIE } .
:g2 { :JEAN :contacter :LOUIS } .

:g1 :moyen :téléphone .
:g2 :moyen :courriel.
```

b) (1 point) Proposez une représentation utilisant un événement de "contact" plutôt qu'une relation.

```
:ContactEvent rdfs:subClassOf :Event .
:contact1 a :ContactEvent ;
          :agent :JEAN ;
          :bénéficiaire :MARIE ;
          :moyen :téléphone .

:contact2 a ContactEvent ;
          :agent :JEAN ;
          :bénéficiaire :LOUIS ;
          :moyen :courriel .
```

c) (1 point) Comparez ces deux approches (avantages, situations où elles sont plus adéquates, simplicité des requêtes SPARQL, etc.)