Nom :
Prénom :

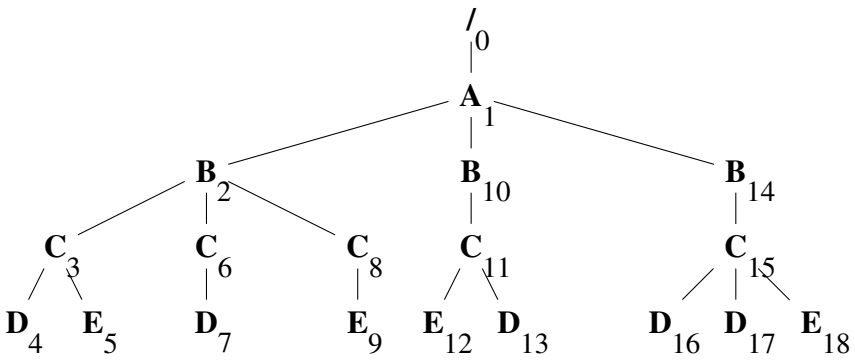
ASWS – Apprentissage Symbolique et Web Sémantique

Examen répartie du 16 novembre 2016
partie Web Sémantique
CORRIGÉ
Documents autorisés

Les téléphones mobiles doivent être éteints et rangés dans les sacs. Le barème sur 20 points (10 questions) n'a qu'une valeur indicative.

XML (3 pts)

Soit donné l'arbre XML suivant où chaque noeud est identifié par un attribut `@id` qui correspond à sa position dans l'ordre préfixe de l'arbre. L'identifiant de la racine est égal à 0, son élément fils de type `A` a l'identifiant 1 etc.:



Le résultat d'une expression XPath est une liste de noeuds DOM triés dans l'ordre du document et sans doublons. Par exemple, l'expression XPath `/descendant::B/@id` retourne les identifiants $2, 10, 14$.

Question 1 (1 point)

Donnez pour chaque expression XPath ci-dessous la liste des identifiants des noeuds retournés.

Réponse :

- $/descendant :: */child :: C[1]/@id$:
- $/descendant :: D/following - sibling :: *[1]/@id$:
- $/descendant :: C[child :: E[following - sibling :: D]]/@id$:
- $/descendant :: *[not(following :: *)]/@id$:

Solution:

- 3, 11, 15: $/descendant :: */child :: C[1]/@id$
- 5, 17, 18: $/descendant :: D/following - sibling :: *[1]/@id$
- 11: $/descendant :: C[child :: E[following - sibling :: D]]/@id$
- 1, 14, 15, 18: $/descendant :: *[not(following :: *)]/@id$

Question 2 (2 points)

Donnez une expression XQuery qui retourne pour chaque noeud avec au moins un enfant, son identifiant et le nombre d'enfants (vous pouvez utiliser la fonction `count`):

```
<A id="1" nb_enfants="3"/>
<B id="2" nb_enfants="3"/>
<C id="3" nb_enfants="2"/>
<C id="6" nb_enfants="1"/>
...
```

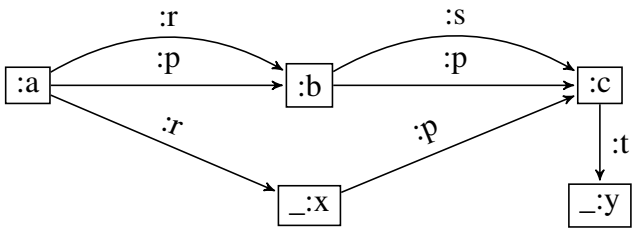
Réponse :

Solution:

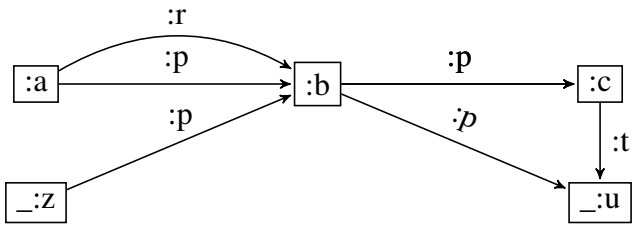
```
for $e in doc("test.xml")/descendant::*[child::*]
return element { name($e) } {
       $e/@id,
       attribute nb_enfants { count($e/child::*) }
}
```

RDF (12 pts)
Soit les deux graphes RDF suivants:

G1:



G2:



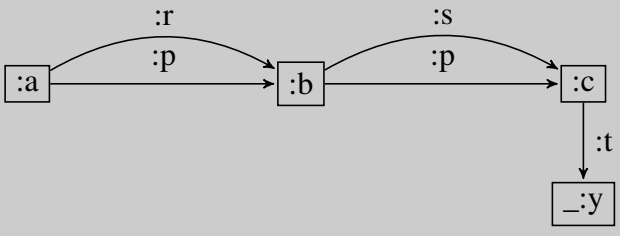
On suppose que les noeuds et les propriétés appartiennent à l'espace de nom `http://exemple.org/`. Chaque noeud est étiqueté par son identifiant ou par une lettre précédée par '_:' s'il s'agit d'un noeud blanc. Les arcs sont étiquetés par les types des propriétés.

Question 3 (2 points)
Donnez les formes normales de $G1$ et de $G2$ en format turtle.

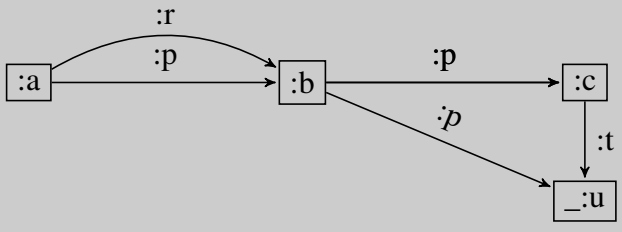
Réponse :

Solution:

G1:



G2:



Question 4 (2 points)
Est-ce que $G1 \models G2$ et/ou $G2 \models G1$? Justifiez votre réponse formellement.

Réponse :

- $G1 \models G2$?

- $G2 \models G1$?

Solution:

- $G1 \not\models G2$: Il n'existe pas de mapping m tel que $m(G2)$ est un sous-graphe de $G1$
  ou
  le motif (:b :p _:u), (:c :t _:u) dans G2 ne peut pas être mappé sur un sous-graphe de G1
  ou
  il n'existe pas de noeud dans G1 connecté a :b bar :p et à :c par :t.
- $G2 \not\models G1$: Il n'existe pas de mapping m tel que $m(G1) \subseteq G2$ est un sous-graphe de $G2$
  ou
  le triplet (:b :s :c) dans G1 n'existe pas dans G2.

Schémas/règles RDFS: Voici un sous-ensemble des règles RDF/S vues en cours:

- rdfD2: a p b . ⇒ p rdf:type rdf:Property .
- rdfs2: p rdfs:domain x . a p b . ⇒ a rdf:type x .
- rdfs3: p rdfs:range x . a p b . ⇒ b rdf:type x .
- rdfs4b: a p b . ⇒ b rdf:type rdf:Resource .
- rdfs5 : p rdfs:subPropertyOf q . q rdfs:subPropertyOf r . ⇒ p rdfs:subPropertyOf r .
- rdfs7: p rdfs:subPropertyOf q . a p b . ⇒ a q b .
- rdfs9: c rdfs:subClassOf d . a rdf:type c . ⇒ a rdf:type d .
- rdfs11: c rdfs:subClassOf d . d rdfs:subClassOf e . ⇒ c rdfs:subClassOf e .

Soit le schéma RDFS *schema.rdfs* suivant :

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

:B rdfs:subClassOf :A .
:C rdfs:subClassOf :A .
:E rdfs:subClassOf :D .
:F rdfs:subClassOf :C .
:D rdfs:subClassOf :C .

:p rdfs:domain :B ; rdfs:range :E .
:q rdfs:domain :D ; rdfs:range :A .
:r rdfs:domain :E ; rdfs:range :C .
:s rdfs:domain :F ; rdfs:range :E .
:s rdfs:subPropertyOf :r .
```

Question 5 (2 points)

Donnez pour chaque expression RDF suivant *les types des ressources inférés* par les définitions des propriétés (domain, range) et de sous-classes (subClassOf) dans le schéma RDFS précédent. On suppose que chaque graphe inclut les trois définitions de préfixes du schéma RDFS.

Réponse :

1.
```
:x :r :y .
```

- :x :
- :y :

2.
```
:x :r :x .
```

- :x :
- :y :

3.
```
:x :q :y ; :s :y, :z.
```

- :x :
- :y :
- :z :

Solution:

1.

```
:x :r :y .
```

```
<x>
    <r> <y> ;
    a <A>, <C>, <D>, <E> .

<y>
    a <A>, <C> .
```

2.

```
:x :r :x .
```

```
<x>
    <r> <x> ;
    a <A>, <C>, <D>, <E> .
```

3.

```
:x :q :y ; :s :y, :z.
```

```
<x>
    <q> <y> ;
    <r> <y>, <z> ;
    <s> <y>, <z> ;
    a <A>, <C>, <D>, <F> .

<y>
    a <A>, <C>, <D>, <E> .

<z>
    a <A>, <C>, <D>, <E> .
```

SPARQL: Soit un graphe RDF/S *graphe.ttl* qui contient un schéma RDFS avec les instances. On suppose que le graphe *graphe.ttl* est saturé (par l'application des règles RDF/S).

Question 6 (6 points)

Exprimez les requêtes suivantes en SPARQL sur *graphe.ttl*. Vous n'êtes pas obligé de spécifier les espaces de nom utilisés (led préfixes `rdf:` et `rdfs:` désignent les espaces de noms RDF et RDFS et le préfixe `:` désigne l'espace de nom du graphe RDF).

Réponse :

Toutes les sous-classes de la classe `:A`.

Toutes les ressources (instances) de la classe `:A` qui ne sont pas des instances de la classe `:B`.

Toutes les propriétés qui ont comme domaine une sous-classe de `:C` et pas la classe `:A` comme co-domaine.

Toutes les propriétés définis sur des instances (ressources) de la classe `:A`.

Toutes les classes avec avec leurs instances propres (qui ne sont pas des instances d'une sous-classe).

On affiche aussi les classes sans instances.

Toutes les ressources qui sont des instances de la classe `:A` et de toutes ses sous-classes.

Solution:

Toutes les sous-classes de la classe `:A`.

```
select distinct ?a
from <graphesat.ttl>
where { ?a rdfs:subClassOf :A  . }
```

Toutes les ressources (instances) de la classe `:A` qui ne sont pas des instances de la classe `:B`.

```
select distinct ?x
from <graphesat.ttl>
where { { ?x rdf:type :A } minus { ?x rdf:type :B } }
```

Toutes les propriétés qui ont comme domaine une sous-classe de `:C` et pas la classe `:A` comme co-domaine.

```
select distinct ?p
from <graphesat.ttl>
where {{ ?p rdfs:domain [ rdfs:subClassOf :C ] }
       minus { ?p rdfs:range  :A  }}
```

Solution:

Toutes les propriétés définis sur des instances (ressources) de la classe `:A`.

```
select distinct ?p
from <graphesat.ttl>
where { ?x rdf:type :A ; ?p [] }
```

Toutes les classes avec avec leurs instances propres (qui ne sont pas des instances d'une sous-classe). On affiche aussi les classes sans instances.

```
select distinct ?t ?x
from <graphesat.ttl>
where {  ?t rdf:type rdfs:Class .
      optional { { ?x rdf:type ?t . }
                 minus { ?x rdf:type ?s .
                         ?s rdfs:subClassOf ?t .
                         filter (?s != ?t) } } }
order by ?t
```

Toutes les ressources qui sont des instances de la classe `:A` et de toutes ses sous-classes.

```
select distinct ?x
from <graphesat.ttl>
where { { ?x rdf:type :A . }
        minus { { ?x rdf:type :A .
                  ?f rdfs:subClassOf :A . }
                minus { ?x rdf:type ?f . } } }
```

Evaluation de requêtes et optimisation (5 pts)

On suppose qu'on a chargé un graphe RDF dans une base TDB Jena (tdbloader). Voici un extrait des statistiques de la base:

```
(stats
  (meta
    (timestamp "2016-11-09T16:07:25.636+01:00")
    (run@ "2016/11/09 16:07:25 CET")
    (count 40311970))
 ((VAR <rdf:type> <http://xmlns.com/foaf/0.1/Person>)
       677212)
 ((VAR <rdf:type> <http://rdvocab.info/uri/schema/FRBRentitiesRDA/Work>)
       107213)
 (<http://xmlns.com/foaf/0.1/name> 647278)
 (<http://purl.org/dc/terms/creator> 90962)
 (<http://www.openarchives.org/ore/terms/isAggregatedBy> 2940)
 (<http://rdvocab.info/ElementsGr2/dateOfDeath> 75748)
 (<http://rdvocab.info/ElementsGr2/dateOfBirth> 74069)
 (other 0))
```

Question 7 (½ point)

Combien d'instances il y a pour chaque classe ou propriété ci-dessous:

Réponse :

- <http://xmlns.com/foaf/0.1/Person>:
- <http://xmlns.com/foaf/0.1/name>:
- <http://rdvocab.info/ElementsGr2/dateOfDeath>:

Solution:

- <http://xmlns.com/foaf/0.1/Person>: 677 212
- <http://xmlns.com/foaf/0.1/name>: 647 278
- <http://rdvocab.info/ElementsGr2/dateOfDeath>: 75 748

On veut exécuter la requête SPARQL suivante:

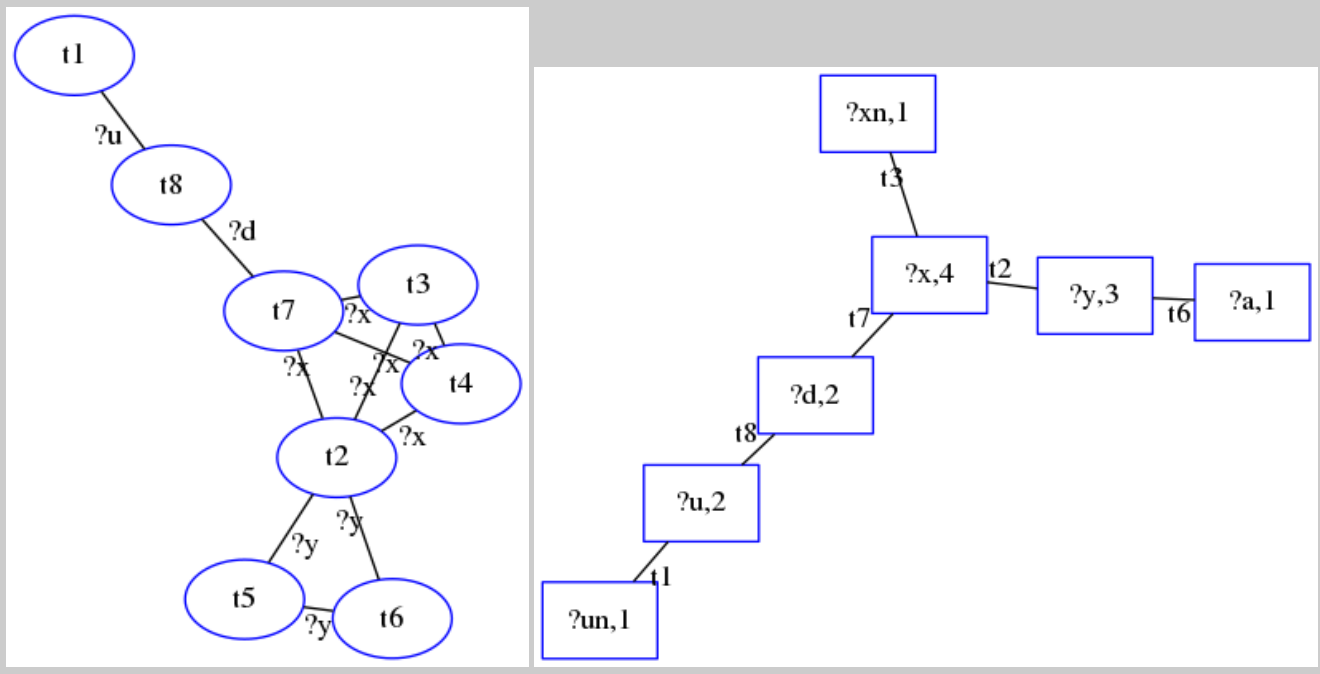
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema>

SELECT *
WHERE {
        ?u <http://xmlns.com/foaf/0.1/name> ?un .
        ?y <http://purl.org/dc/terms/creator> ?x .
        ?x <http://xmlns.com/foaf/0.1/name> ?xn .
        ?x a <http://xmlns.com/foaf/0.1/Person> .
        ?y a <http://rdvocab.info/uri/schema/FRBRentitiesRDA/Work> .
        ?y <http://www.openarchives.org/ore/terms/isAggregatedBy> ?a .
        ?x <http://rdvocab.info/ElementsGr2/dateOfBirth> ?d .
        ?u <http://rdvocab.info/ElementsGr2/dateOfDeath> ?d .
}
limit 10
```

Question 8 ($1\frac{1}{2}$ points)
Dessinez le graphe de jointure et le graphe de variables de la requête précédente. Vous pouvez numéroter les triplets dans l'ordre de leur apparition dans la requête ($t1$, $t2$, ..., $t8$).

Réponse :

Solution:

```
t1: ?u <http://xmlns.com/foaf/0.1/name> ?un .
t2: ?y <http://purl.org/dc/terms/creator> ?x .
t3: ?x <http://xmlns.com/foaf/0.1/name> ?xn .
t4: ?x a <http://xmlns.com/foaf/0.1/Person> .
t5: ?y a <http://rdvocab.info/uri/schema/FRBRentitiesRDA/Work> .
t6: ?y <http://www.openarchives.org/ore/terms/isAggregatedBy> ?a .
t7: ?x <http://rdvocab.info/ElementsGr2/dateOfBirth> ?d .
t8: ?u <http://rdvocab.info/ElementsGr2/dateOfDeath> ?d .
```



Question 9 (1 point)

Expliquez pourquoi l'évaluation de la requête prend beaucoup de temps si on n'applique les filtres dans l'ordre d'apparition dans la clause WHERE (pas d'optimisation, **none.opt**).

Réponse :

Solution: En regardant le graphe de variables, on voit que l'ordre d'évaluation consiste à calculer plusieurs produits cartesiens ce qui est très inefficace.

Voici l'ordre d'évaluation de jointures qu'on obtient en utilisant la stratégie **fixed.opt** (heuristiques sans statistiques):

```
t4 : (?x rdf:type <http://xmlns.com/foaf/0.1/Person>)
t3 : (?x <http://xmlns.com/foaf/0.1/name> ?xn)
t7 : (?x <http://rdvocab.info/ElementsGr2/dateOfBirth> ?d)
t2 : (?y <http://purl.org/dc/terms/creator> ?x)
t5 : (?y rdf:type <http://rdvocab.info/uri/schema/FRBRentitiesRDA/Work>)
```

```
t6 : (?y <http://www.openarchives.org/ore/terms/isAggregatedBy> ?a)
t8 : (?u <http://rdvocab.info/ElementsGr2/dateOfDeath> ?d)
t1 : (?u <http://xmlns.com/foaf/0.1/name> ?un)
```

Question 10 (2 points)

Est-ce que ce plan est optimal ? Expliquez comment il pourrait être amélioré en utilisant les statistiques (**stats.opt**).

Réponse :

Solution: On doit commencer avec des triplets de faible cardinalité en les réordonnant (par exemple, t6 a une plus faible cardinalité que t4):

```
t6 : (?y <http://www.openarchives.org/ore/terms/isAggregatedBy> ?a)
t5 : (?y rdf:type <http://rdvocab.info/uri/schema/FRBRentitiesRDA/Work>)
t2 : (?y <http://purl.org/dc/terms/creator> ?x)
t7 : (?x <http://rdvocab.info/ElementsGr2/dateOfBirth> ?d)
t3 : (?x <http://xmlns.com/foaf/0.1/name> ?xn)
t4 : (?x rdf:type <http://xmlns.com/foaf/0.1/Person>)
t8 : (?u <http://rdvocab.info/ElementsGr2/dateOfDeath> ?d)
t1 : (?u <http://xmlns.com/foaf/0.1/name> ?un)
```