Tutoriel - RDF et SPARQL

Fabien Duchateau (Université Claude Bernard Lyon 1) - 2025

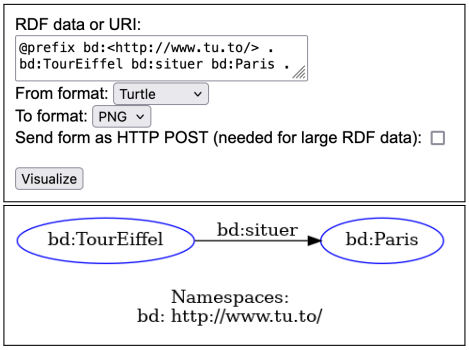
Resource Description Framework (RDF) est un modèle standard (du W3C) pour décrire des graphes au moyen de triplets.

1. Un **triplet** définit un prédicat (ou relation) entre un sujet et un objet (similaire à une phrase simple). Les éléments d'un triplet sont des ressources, qui possèdent une identifiant sous forme d'IRI (identifiant de ressource, e.g., une URL).
Ci-contre un triplet indiquant que la Tour Eiffel est située à Paris.

```
<http://www.tu.to/TourEiffel>
<http://www.tu.to/situer>
<http://www.tu.to/Paris> .
```

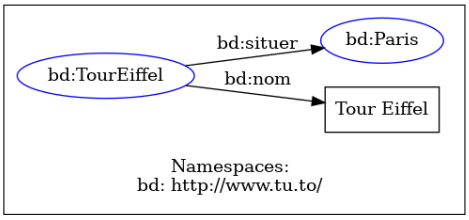
2. Pour simplifier l'écriture des IRI, on peut utiliser des **préfixes**, qui remplacent une partie de l'IRI. Ici notre préfixe s'appelle `bd`. Notez qu'un triplet se termine par un point.
Pour visualiser nos données RDF, copiez-collez le code sur RDFgrapher.

```
@prefix bd:<http://www.tu.to/> .
bd:TourEiffel bd:situer bd:Paris .
```



3. Ajoutons un nom à notre monument. Les guillemets définissent un type particulier de ressource, les **littéraux** (ou valeurs). Un littéral ne peut pas être sujet (donc aucune flèche ne peut en partir).
Nous avons réutilisé notre ressource `bd:TourEiffel` (encore comme sujet), et complété le reste du triplet.

```
@prefix bd:<http://www.tu.to/> .
bd:TourEiffel bd:situer bd:Paris .
bd:TourEiffel bd:nom "Tour Eiffel".
```

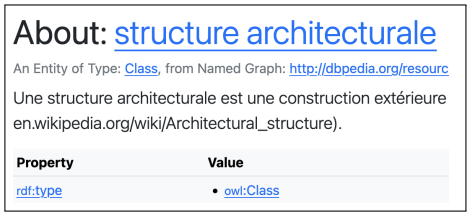


4. DBpedia est un immense **graphe de connaissances** qui contient des millions d'entités (personnes, lieux, événements, etc.). Une entité est représentée par une ressource RDF.
Utilisez le moteur de recherche de DBpedia pour trouver la ressource qui représente la Tour Eiffel et visualiser ses propriétés.
Mais où sont les données en RDF ?!

```
Saisissez "tour eiffel" dans le
formulaire, et cliquez sur la
première entité proposée dans les
résultats.

La nouvelle page décrit les
métadonnées sur cette ressource,
cliquez sur About Tour Eiffel. Et
on arrive sur la ressource :
https://dbpedia.org/page/Eiffel_Tower

Pour les données en RDF, cliquez
sur Formats et choisissez Turtle.
Cherchez quelques unes des classes
qui typent notre monument !
```

5. Ci-contre quelques triplets de typage pour la Tour Eiffel.
RDF favorise la **réutilisation des ressources**. Aussi des vocabulaires contrôlés définissent des classes, des prédicats, etc. qui peuvent être réutilisés ailleurs.
DBpedia spécifie un vocabulaire pour ses classes (préfixe `dbo`). Mais elle utilise aussi le vocabulaire RDF (`rdf:type`) et des ressources définies ailleurs, comme Wikidata (un autre graphe).

```
@prefix dbo:<http://dbpedia.org/ontology/> .
@prefix dbr:<http://dbpedia.org/resource/> .
dbr:Eiffel_Tower rdf:type dbo:ArchitecturalStructure,
dbo:Building .
# la classe ArchitecturalStructure possède une page
descriptive (elle ausssi en RDF, voir ci-contre)

@prefix wikidata:<http://www.wikidata.org/entity/>
dbr:Eiffel_Tower rdf:type wikidata:Q41176 .

# à quoi correspond ce wikidata:Q41176 ?
```



Réponse

6. DBpedia offre différents moyens de chercher les données, et notamment un point d'accès utilisant le **langage d'interrogation SPARQL** pour des données RDF.
Rendez-vous sur l'interface du point d'accès SPARQL de DBpedia. La requête ci-contre demande tous les sujets (représentés par une variable `?s`) d'un triplet contenant un prédicat `rdf:type` et un objet `dbo:ArchitecturalStructure`. Dans le résultat, une (longue) liste de ressources...

```
select distinct ?s
where {?s rdf:type
dbo:ArchitecturalStructure .}
```

Le motif de la requête correspond à :

s
http://dbpedia.org/resource/Ca'_Granda_(Milan_Metro)
http://dbpedia.org/resource/Ca'_Loredan
http://dbpedia.org/resource/Ca'_Pesaro
http://dbpedia.org/resource/Ca'_Rezzonico
...

7. L'exécution d'une requête SPARQL cherche à apparier le motif de la requête avec les données, en remplaçant chaque variable par une ressource.
Voyageons un peu en s'intéressant à un lieu retourné par la requête précédente : le Musée canadien des langues. Nous voulons **récupérer toutes les informations** sur cette entité en tant que sujet. Les variables `?p` et `?o` représentent le prédicat et l'objet des triplets recherchés.

```
select *
where {dbr:Canadian_Language_Museum
?p ?o .}

# comment modifier la requête pour
compter les informations quand le
musée est en objet ?
```

p	o
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#Thing
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.wikidata.org/entity/Q33506
...	...

```
select (count(*) as ?nb)
where {?s ?p
dbr:Canadian_Language_Museum .}
```

nb
8

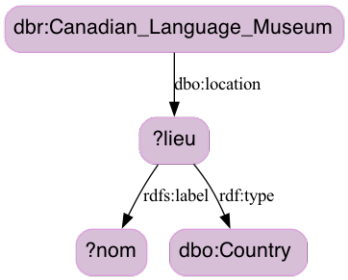
8. Utilisons **un motif avec plusieurs triplets** (correspondant à un "et") pour obtenir le libellé du pays où se trouve le musée.
Le prédicat `dbo:location` spécifie le lieu d'une entité. Mais ce lieu est aussi une ressource (e.g., `dbr:Canada`) et le second triplet permet donc de retrouver son nom grâce au prédicat `rdfs:label`. Notez que le nom des entités est souvent disponible en plusieurs langues, d'où le filtre sur la variable `?nom` en anglais uniquement.
Dans le résultat, on se rend compte que le prédicat `dbo:location` mentionne le pays, mais aussi l'état, la ville, et la rue. Comment faire pour ne garder que le pays ?

```
select *
where {dbr:Canadian_Language_Museum
dbo:location ?lieu .
?lieu rdfs:label ?nom .
filter(lang(?nom) = "en")
}
```

lieu	nom
http://dbpedia.org/resource/Canada	"Canada"@en
http://dbpedia.org/resource/Bayview_Avenue	"Bayview Avenue"@en
http://dbpedia.org/resource/Toronto	"Toronto"@en
http://dbpedia.org/resource/Ontario	"Ontario"@en

lieu	nom
http://dbpedia.org/resource/Canada	"Canada"@en

Une aide pour ne garder que le pays :



Réponse

9. Dans un motif, les triplets sont reliés implicitement par un "et". L'opérateur `union` permet de réaliser un **"ou" entre motifs** (i.e., de chercher à apparier différents motifs).
Ici nous voulons le nom du pays et de la ville, toujours pour notre musée canadien. On récupère les lieux du musée, puis l'union des 2 triplets suivants spécifie que ce lieu est une ville ou un pays.

```
select *
where {dbr:Canadian_Language_Museum
dbo:location ?lieu .
{
  {?lieu rdf:type dbo:City .}
union
  {?lieu rdf:type dbo:Country .}
}
?lieu rdfs:label ?nom .
filter(lang(?nom) = "en")
}
```

lieu	nom
http://dbpedia.org/resource/Toronto	"Toronto"@en
http://dbpedia.org/resource/Canada	"Canada"@en

10. Le langage SPARQL dispose de **nombreuses fonctions** pour manipuler les chaînes de caractères, les types, les dates, etc (liste des fonctions).
L'exemple ci-contre s'intéresse aux termes (`dcterms:subject`) qui décrivent notre musée. On récupère la valeur de ces termes, que l'on filtre pour ne garder que ceux contenant *museum* (la fonction `lcase` transforme en minuscules). Dans la clause `select`, on concatène des variables en une seule.

```
select (concat(?lmusee, " - ",
?lterm) as ?nom_term)
where {
dbr:Canadian_Language_Museum
dcterms:subject ?term ;
foaf:name ?lmusee .
?term rdfs:label ?lterm .
filter(contains(lcase(?lterm),
"museum"))
}
```

nom_term
Canadian Language Museum – University museums in Canada
Le Musée canadien des langues – University museums in Canada
Le Musée canadien des langues – Museums in Toronto
Canadian Language Museum – Museums in Toronto
Le Musée canadien des langues – Language museums
Canadian Language Museum – Language museums

11. Enfin, le langage dispose de clauses pour **le regroupement et le tri**.
Dans cet exemple, le motif indique de trouver tous les types de notre musée (variable `?t`), et de trouver toutes les instances (variable `?s`) qui possèdent ces types. Un regroupement est réalisé sur le type afin de dénombrer le nombre d'instances pour chaque type. Un tri (croissant par défaut) ordonne les solutions.

```
select ?t (count(?s) as ?nb)
where {dbr:Canadian_Language_Museum
rdf:type ?t .
?s rdf:type ?t .
}
group by ?t
order by ?nb
```

t	nb
http://www.wikidata.org/entity/Q33506	9548
http://dbpedia.org/ontology/Museum	13845
http://www.wikidata.org/entity/Q41176	123823
http://dbpedia.org/ontology/Building	165493
http://dbpedia.org/ontology/ArchitecturalStructure	270716
http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing	1270302
http://www.w3.org/2002/07/owl#Thing	5465151

Conclusion

Ce tutoriel est volontairement limité pour en faciliter la prise en main. Des détails sur RDF et SPARQL sont donnés dans les diapositives de cours (https://perso.liris.cnrs.fr/fabien.duchateau/) ou dans les standards RDF et SPARQL.