

## Projet : Knowledge graphs et raisonnement

Le projet est à faire en binôme. Vous soumettrez une archive ou un lien vers vos fichiers sources (e.g. données RDF/S, code java, requêtes SPARQL, y compris ceux nécessaires à l'extraction / génération des données), ainsi qu'un rapport en anglais ou français au format pdf, décrivant votre travail. La date limite pour soumettre ces documents sur Moodle est le 5 janvier 2026.

### 1. Jeux de données reliés

Récupérez plusieurs jeux de données (en accès libre) qui sont susceptibles d'avoir des ressources en commun. Par exemple des entrées sur les pays et les villes du monde existent en Dbpedia, en Wikidata, ainsi que dans des jeux de données géographiques comme Geonames <https://www.geonames.org>.

Plusieurs jeux de données RDF sur des sujets variés sont rendus disponibles par le W3C <https://www.w3.org/wiki/DataSetRDFDumps>. D'autres jeux de données, majoritairement en format CSV ou Json sont disponibles également sur Kaggle <https://www.kaggle.com/datasets> et sur Datahub <https://datahub.io/>. Vous pouvez également utiliser le moteur de recherche pour datasets de Google <https://datasetsearch.research.google.com/> en se limitant aux jeux de données libres d'accès.

#### 1.1. Extraction des données

Généralement il existe plusieurs façons d'extraire ces données

- téléchargement de fichiers csv,
- dump RDF
- interrogation de SPARQL endpoints ou autres services Web (par interface Web ou par du logiciel)
- bibliothèques clientes (pour python, java etc.) qui facilitent l'extraction automatique (cf. par exemple <https://www.geonames.org/export/client-libraries.html> pour GeoNames).

Vous pouvez également récupérer des données directement sur le web (par exemple en utilisant les bibliothèques de la Python Data Science Stack).

Veillez à ce que les différents jeux de données aient des concepts en commun. Par exemple la ville de Paris (comme n'importe quelle autre ville) est représentée aussi bien en GeoNames par

l'URI <https://sws.geonames.org/2988507/>, qu'en Dbpedia <https://dbpedia.org/resource/Paris>, qu'en Wikidata par <http://www.wikidata.org/entity/Q90>.

## 2. Intégration des données

Quel que soit le format d'origine des données vous les convertirez tous en RDF. Pour cela vous pouvez vous inspirer des techniques apprises dans le TP2b, ou bien vous pouvez écrire votre propre programme de transformation, dans le langage de votre choix. Attention à bien choisir la façon de modéliser les données en RDF. Dans le TP2b vous trouverez un exemple de restructuration de données relationnelles en RDF.

Vous obtiendrez un unique jeu de données intégrées en ajoutant des relations entre les données (comme par exemple des faits `owl:sameAs`, comme dans le TP2b, ou des triplets additionnelles de votre choix pour créer des liens (même fictives) entre les données.

Au besoin n'oubliez pas que vous pouvez vous servir de jena pour lire, manipuler, et écrire du RDF, cf. la documentation Jena et en particulier [https://jena.apache.org/tutorials/rdf\\_api.html#ch-Reading%20RDF](https://jena.apache.org/tutorials/rdf_api.html#ch-Reading%20RDF).

Dans le rapport décrivez bien tout le processus d'importation et de transformation des données.

## 3. Requêtes

Vous proposerez une dizaine de requêtes SPARQL interrogeant votre base RDF. Les requêtes doivent le plus possible intégrer des informations provenant des différents jeux de données que vous avez combinés (par exemple, une même requête pourrait extraire les coordonnées géographiques de Paris depuis GeoNames ainsi que l'histoire de l'architecture de la ville depuis DBpedia). Les aspects suivants sont requis :

- une requête fédérée interrogeant des sources externes à votre jeux de données;
- une requête avec `OPTIONAL`;
- une requête sur des graphes nommés;
- une requête avec agrégation;
- une requête utilisant des expressions de chemin;
- une requête `MINUS` ou `FILTER NOT EXISTS`.

## 4. Raisonnement sur les données RDF

Dans cette section, vous mettrez en œuvre des mécanismes de raisonnement basés sur RDFS. Le but est de structurer vos données RDF en définissant un schéma clair et bien organisé,

puis de réaliser des inférences simples pour enrichir les données. Ces inférences permettront de découvrir de nouvelles informations à partir des relations et classes déjà définies dans vos jeux de données intégrés.

## 4.1. Construction d'une ontologie RDFS

Definissez une ontologie RDFS pour décrire les relations entre les classes et les propriétés au sein de votre modèle RDF. Cela servira de base pour le raisonnement.

Suivez les étapes suivantes :

1. Si vos jeux de données possèdent déjà un schéma (des URIs de classes, de propriétés et des relations RDFS entre celles-ci), vous pouvez les réutiliser. Sinon identifiez les entités principales de vos jeux de données et organisez-les en classes; identifiées également les propriétés (prédicats) utilisées.

Par exemple si un de vos jeux de données n'a pas de schéma et possède les triplets

```
:london :capitalOf :UK
:paris :capitalOf :FR
:paris :surface 105.4
```

Vous pouvez définir un schema RDFS qui contient entre autre des propriétés `:capitalOf`, `:surface`, etc., ainsi que les classes `:City`, `:Country`, etc.

2. Définissez ensuite votre ontologie RDFS pour relier ces classes et propriétés. Les éléments suivants doivent être pris en compte :
  - **Classes et hiérarchie de classes** : Utilisez la contrainte `rdfs:subClassOf` pour créer une hiérarchie entre les classes. Par exemple, vous pourriez avoir une classe `:City` qui est une sous-classe de `:Location`.
  - **Propriétés et hiérarchie de propriétés** : Utilisez `rdfs:subPropertyOf` pour structurer les relations entre propriétés. Par exemple, si vous avez une propriété `capitalOf` pour indiquer qu'une ville est la capitale d'un pays, vous pourriez créer une super-propriété `locatedIn` pour préciser que cette ville se trouve dans ce pays.
  - **Domaines et co-domaines** : Assurez-vous de définir les `rdfs:domain` et `rdfs:range` pour chaque propriété. Par exemple, pour la propriété `locatedIn`, le domaine pourrait être `City` et le co-domaine `Country`.
3. Définissez des contraintes comme au point précédent mais entre classes et propriétés de différents jeux de données, par exemple

```
dbp:region rdfs:subPropertyOf :LocatedIn
geo:inCountry rdfs:subPropertyOf :LocatedIn
```

permet d'unifier deux propriétés, une du vocabulaire Dbpedia et une du vocabulaire GeoNames en une unique propriété. De la même façon il est possible d'unifier des classes de différents jeux de données.

4. Ajoutez ensuite les faits de base pour peupler les nouvelles classes et propriétés, si vous en avez (par exemple `:london rdf:type :City`, `:FR rdf:type Country`, etc.). Notez que des ressources de différents jeux de données peuvent être déclarées appartenante à la même classe, ce qui facilite leur intégration.

## 4.2. Raisonnement avec RDFS

Une fois le schéma défini, le raisonnement peut commencer. Vous pouvez utiliser les modèles RDF préconfigurés de Jena qui appliquent automatiquement les règles de raisonnement RDFS aux données RDF. Voici les étapes de base à suivre :

- Importez vos données RDF dans un modèle Jena.
- Appliquez un modèle de raisonnement RDFS à vos données.
- Exécutez des requêtes SPARQL sur le modèle enrichi pour découvrir les informations déduites à partir des règles RDFS.

Reférez-vous au TP3b et à la documentation Jena indiquée dans ce TP pour les détails de mise en oeuvre de ce point.

À l'aide des requêtes SPARQL, présentez les résultats que vous obtenez après avoir appliqué les règles de raisonnement RDFS. Comparez les données brutes à celles qui ont été enrichies grâce aux inférences. Par exemple, montrez comment certaines entités sont automatiquement classées dans des catégories supérieures grâce à `rdfs:subClassOf`, ou comment des relations sont étendues grâce à `rdfs:subPropertyOf`. Vous devez concevoir des requêtes interrogeant plusieurs jeux de données en même temps, et combinant des informations enrichies par le raisonnement.

## 5. Conclusion

Concluez en expliquant comment ces requêtes SPARQL, combinées au raisonnement RDFS, permettent d'exploiter efficacement vos jeux de données intégrés. Mentionnez également les limites potentielles du raisonnement RDFS et discutez des éventuelles améliorations possibles pour votre projet, comme l'extension à des schémas plus riches ou l'intégration de nouveaux jeux de données.

N'oubliez pas de citer les sources académiques, documentations et tutoriels que vous avez utilisés pour réaliser ce projet.