

## TD8

### Ontologie : technologies support

#### Cours correspondant

SPARQL/SWRL : technologies support

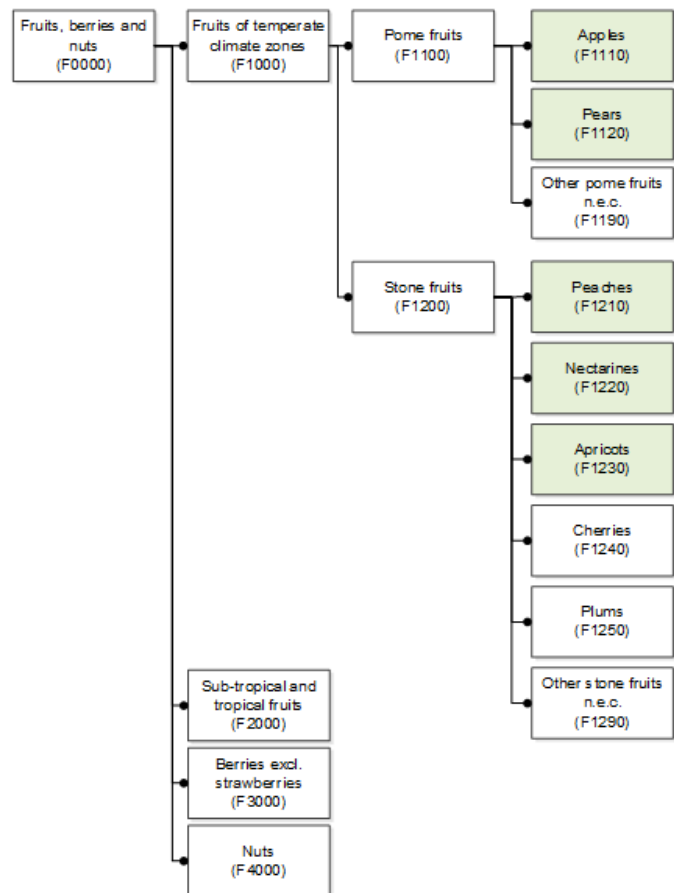
#### Objectifs

- Mettre en œuvre les connaissances acquises en construisant une ontologie à l'aide de technologies support
- Découvrir les technologies disponibles pour gérer les ontologies
- Écrire du code Python pour construire et intégrer des ontologies

### Exercice 1 : Modélisation de la Hiérarchie des Fruits

La figure suivante<sup>1</sup> illustre la hiérarchie des fruits à pépins (pome) et à noyaux (stone) pour le module « vergers » (orchards) qui a été définie dans le manuel des statistiques agricoles intégrées de la commission européenne.

- Utilisez RDFLib pour modéliser la hiérarchie des fruits montrée dans la figure.
- Créez des classes pour les différentes catégories de fruits (par exemple, PomeFruits, StoneFruits, etc.).
- Créez des instances pour chaque type de fruit (par exemple, Apples, Pears, etc.).
- Définissez les relations de sous-classe et d'appartenance entre les différentes catégories et types de fruits.
- Ajoutez des propriétés supplémentaires suivantes : « hasColor », « hasTaste » (doux, acide), « isSeasonal » (oui, non) et « requiresColdStorage » aux fruits.
- Sérialisez ce graphe en différents formats : Turtle, RDF/XML, et JSON-LD.



<sup>1</sup> <https://wikis.ec.europa.eu/display/IFS/3.7+IFS+Orchards>

## Exercice 2 : Interroger la Hiérarchie avec SPARQL

En utilisant le graphe RDF modélisé dans l'exercice précédent, interrogez-le avec des requêtes SPARQL en Python avec les deux bibliothèques « RDFLib » :

- Trouvez une requête SPARQL pour obtenir tous les types de fruits qui appartiennent aux Pome Fruits.
- Trouvez une requête SPARQL pour récupérer les informations sur un ensemble de fruits, y compris leurs types et autres attributs intéressants.
- Trouvez une requête SPARQL pour récupérer tous les fruits de saison (« isSeasonal » est vrai).

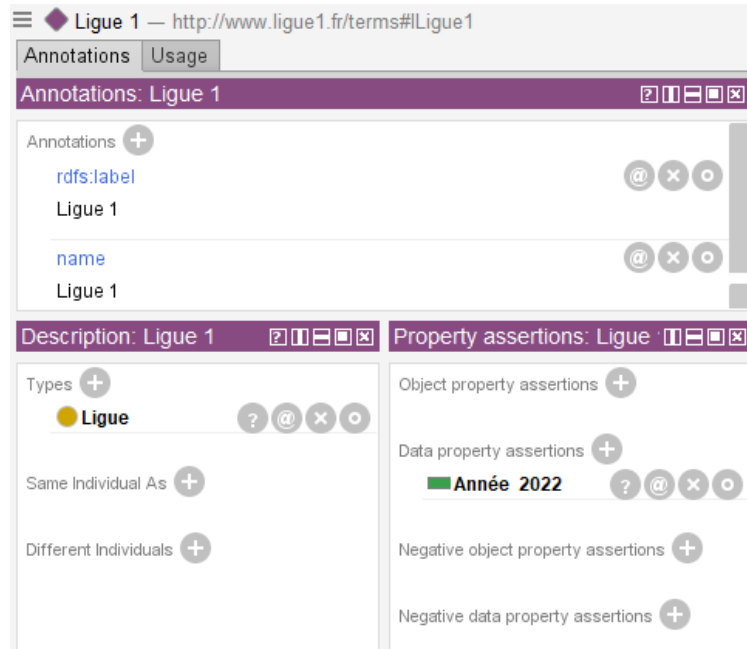
## Exercice 3 : Définir des règles SWRL

En utilisant le graphe RDF modélisé dans l'exercice précédent (exercice 1), définissez et appliquez des règles SWRL en Python avec « owlready2 » pour déduire de nouvelles connaissances :

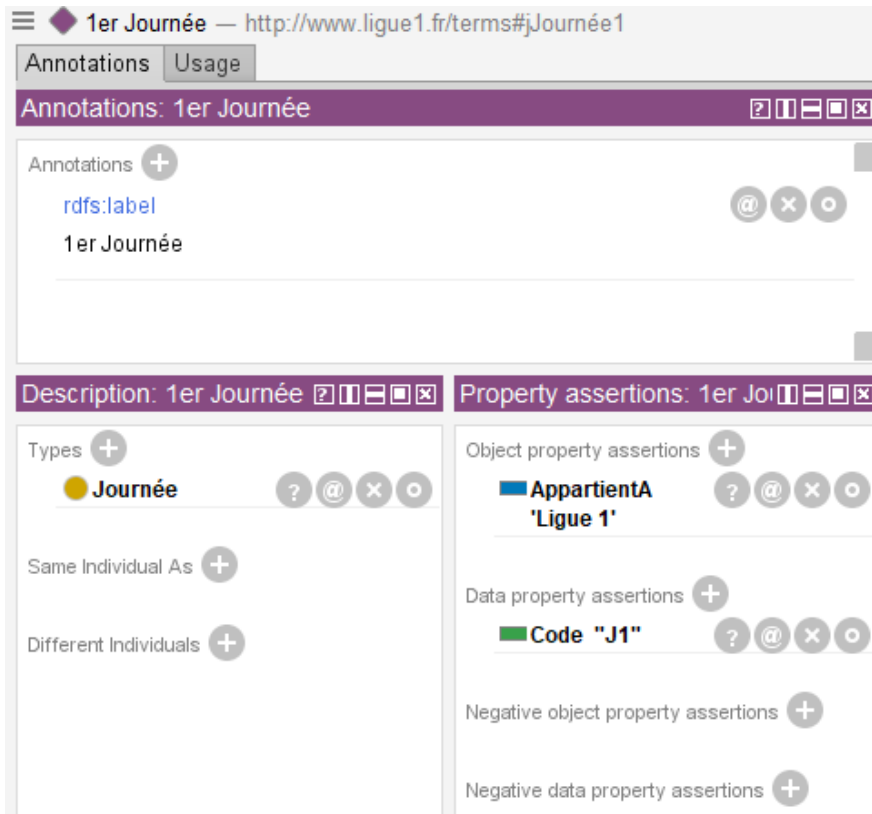
- Créez des règles pour catégoriser automatiquement les fruits en fonction de leurs caractéristiques. (par exemple, un « Apple » est automatiquement un « PomeFruit ».)
- Écrivez une règle SWRL qui spécifie : "Si un fruit est de type « StoneFruit » et est de couleur « orange » et a un goût « doux », alors il est probablement un « Peaches »".
- Créez une règle SWRL pour vérifier : « Si un fruit est un « StoneFruit » et est saisonnier (isSeasonal est vrai), alors il nécessite un stockage à froid « requiresColdStorage ». »

## Exercice 4 :

- En utilisant l'ontologie Ligue précédemment étendue, ajoutez les individus suivants avec la librairie RDFlib
  - Ligue 1



- 1<sup>er</sup> Journée



- 16<sup>e</sup> Journée

16e Journée — <http://www.ligue1.fr/terms#Journée2>

Annotations Usage

Annotations: 16e Journée

Annotations +

`rdfs:label`

16e Journée

Description: 16e Journée

Types +

● Journée

Same Individual As +

Different Individuals +

Property assertions: 16e Journée

Object property assertions +

■ AppartientA "Ligue 1"

Data property assertions +

■ Code "J16"

Negative object property assertions +

Negative data property assertions +

## - Jérémie Stinat

Jérémie Stinat — <http://www.ligue1.fr/terms#arJStinat%20>

Annotations Usage

Annotations: Jérémie Stinat

Annotations +

`rdfs:label`

Jérémie Stinat

Description: Jérémie Stinat

Types +

● Arbitre

Same Individual As +

Different Individuals +

Property assertions: Jérémie Stinat

Object property assertions +

Data property assertions +

■ `familyName` "Stinat"

■ `firstName` "Jérémie"

Negative object property assertions +

Negative data property assertions +

- Clermont

Annotations: Clermont

Annotations	
<code>rdfs:label</code>	@ x o
Clermont	
<code>name</code>	@ x o
Clermont	

Description: Clermont

Types	
● Équipe	? @ x o

Same Individual As +

Different Individuals +

Property assertions: Clermont

Property assertions	
Object property assertions +	
Data property assertions +	
Negative object property assertions +	
Negative data property assertions +	

- Paris Saint-Germain

Paris Saint-Germain — <http://www.ligue1.fr/terms#eqPSG>

Annotations    Usage

Annotations: Paris Saint-Germain

Annotations +

`rdfs:label`    @    ✕    ○

Paris Saint-Germain

`name`    @    ✕    ○

Paris Saint-Germain

Description: Paris Saint-Germain    ?    ||    □    □    ✕

Property assertions: Paris Saint-Germain    ||    □    □    ✕

Types +

● Équipe    ?    @    ✕    ○

Same Individual As +

Different Individuals +

Object property assertions +

Data property assertions +

Negative object property assertions +

Negative data property assertions +

- Strasbourg

The screenshot shows the Protégé interface with the 'Strasbourg' class selected. The 'Annotations' tab is active, displaying two annotations: 'rdfs:label' with the value 'Strasbourg' and 'name' with the value 'Strasbourg'. The 'Description' tab is also visible, showing the class 'Équipe' with the property 'Same Individual As' set to 'Strasbourg'. The 'Property assertions' panel on the right is empty.

## - Match 1.2

Match 1.2 — <http://www.ligue1.fr/terms#mMatch2>

Annotations Usage

Annotations: Match 1.2

Annotations +

`rdfs:label`

Match 1.2

Description: Match 1.2

Types +

**MatchJoué**

Same Individual As +

Different Individuals +

Property assertions: Match 1.2

Object property assertions +

**JouéDans '1er Journée'**

**ArbitréPar 'Jérémy Stinat'**

**ÉquipeLocale Clermont**

**ÉquipeHôte 'Paris Saint-Germain'**

Data property assertions +

**ButsHôte 5**

**ButsDomicile 0**

**Code "J1"**

**JouéLe "2022-08-06"^^xsd:date**

## - Match 16.1

Match 16.1 — <http://www.ligue1.fr/terms#mMatch3>

Annotations Usage

Annotations: Match 16.1

Annotations +

`rdfs:label`

Match 16.1

Description: Match 16.1

Types +

**Match**

Same Individual As +

Different Individuals +

Property assertions: Match 16.1

Object property assertions +

**ÉquipeLocale 'Paris Saint-Germain'**

**ÉquipeHôte Strasbourg**

**JouéDans '16e Journée'**

Data property assertions +

**JouéLe "2022-12-28"^^xsd:date**

**Code "J16"**

- Utilisez les préfixes suivants pour toutes les requêtes

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX ligue: <http://www.ligue1.fr/terms#>
```

Trouvez la requête appropriée pour

1. Affichez toutes les instances qui ont la propriété "foaf:name".
2. Affichez les prénoms des arbitres.
3. Affichez les noms des équipes locales qui ont participé aux matchs joués.
4. Affichez les noms des équipes visiteuses qui ont participé aux matchs à venir.
5. Affichez le nombre de buts marqués par les équipes locales aux matchs joués.

Ajoutez deux nouvelles propriétés de données `scoreFinalDomicile` et `scoreFinalVisiteuse` à la classe `MatchJoué` et de type entier (`xsd:nonNegativeInteger`)

- A l'aide des règles SWRL, ajoutez la signification suivante à l'ontologie : « les buts marqués par les équipes visiteuses aux matchs joués sont doublés dans le score final. »