

Jeux de données reliés sur le football international

1. Introduction

Ce projet s'inscrit dans le cadre des données liées (Linked Open Data) et du Web sémantique. Il a pour objectif de construire un graphe RDF cohérent à partir de plusieurs sources de données ouvertes portant sur le football international.

Plus précisément, le travail consiste à :

- exploiter des jeux de données CSV décrivant les matchs internationaux et les buteurs,
- modéliser ces données sous forme RDF en définissant un schéma sémantique adapté,
- enrichir les données locales à l'aide de ressources externes issues de DBpedia,
- produire un ensemble de données interrogeable via SPARQL.

Le domaine du football a été choisi car il permet de relier naturellement plusieurs entités (matchs, équipes, joueurs, buts) et d'illustrer les principes fondamentaux de la modélisation sémantique.

2. Jeux de données et transformation en RDF

2.1 Présentation générale des jeux de données

Pour ce projet, deux jeux de données principaux au format CSV ont été utilisés comme sources initiales : l'un décrivant les résultats des matchs internationaux et l'autre détaillant les buts et les buteurs. Ces données ont été complétées par des informations issues de DBpedia afin d'enrichir la description des joueurs et des équipes nationales.

2.1.1 Résultats des matchs internationaux

Le jeu de données International Football Results recense l'ensemble des matchs internationaux officiels depuis 1872.

Chaque enregistrement correspond à un match entre deux équipes nationales et inclut :

- la date du match,
- les équipes jouant à domicile et à l'extérieur,
- le score final,
- le type de tournoi,
- le pays hôte.

Ces informations constituent la base de la modélisation des matchs dans le graphe RDF.

2.1.2 Buteurs et détails des buts

Le jeu de données International Football Results – Goalscorers fournit le détail des buts inscrits lors des matchs internationaux.

Chaque ligne correspond à un but marqué par un joueur à une minute précise d'un match. Les informations principales incluent :

- le nom du joueur ayant marqué,
- la minute du but,
- la date du match,
- les équipes concernées.

Cette structuration permet de modéliser chaque but comme une entité distincte dans le graphe RDF et de relier de manière précise les événements aux joueurs et aux matchs correspondants.

2.1.3 Enrichissement avec DBpedia

À partir des fichiers CSV, un fichier RDF contenant l'ensemble des joueurs (ex:Footballer) a été généré.

Ce jeu de données a ensuite été enrichi avec des informations issues de DBpedia, telles que :

- l'équipe nationale,
- la date et le lieu de naissance,
- la position du joueur,
- les clubs.

Les ressources DBpedia sont identifiées par des URI stables, ce qui facilite leur intégration dans le graphe RDF local.

2.2 Transformation et conversion en RDF

Cette section décrit en détail le processus de transformation des jeux de données CSV en un graphe RDF cohérent.

La génération du graphe a été réalisée à l'aide de requêtes **TARQL**, appliquant des requêtes **SPARQL** de type **CONSTRUCT** directement sur les fichiers CSV, et complétée par des scripts Python pour l'enrichissement des données à partir de DBpedia.

Chaque entité du domaine étudié — matchs, joueurs et buts — est représentée par une URI unique dans l'espace de noms **http://example.org/football/**, garantissant l'identification non ambiguë des ressources. Les vocabulaires standards, en particulier DBpedia Ontology, sont réutilisés lorsque cela est pertinent afin d'assurer l'interopérabilité du graphe avec d'autres jeux de données du Web sémantique.

Les entités locales ainsi créées sont progressivement enrichies par des données externes issues de DBpedia, notamment pour compléter les informations relatives aux joueurs.

Chaque étape de transformation correspond à un fichier de requête distinct, stocké dans le répertoire **queries/creation_du_graphe**, ce qui permet une génération modulaire, reproductible et facilement extensible du graphe RDF.

2.2.1 Génération des matchs

Chaque match est représenté par une ressource RDF unique de type `ex:Match`, avec les propriétés suivantes :

dbo:homeTeam et **dbo:awayTeam** : équipes à domicile et à l'extérieur, reliées à DBpedia pour assurer l'interopérabilité ;

- **dbo:date** : date du match typée en `xsd:date` ;
- **dbo:tournament** : type de tournoi (ex. `Friendly_match`) ;
- **dbo:homeScore** et **dbo:awayScore** : scores finaux typés numériquement ;
- **ex:result** : résultat du match calculé automatiquement (victoire, défaite ou match nul)
- **dbo:country** : pays hôte, relié à DBpedia.

Cette modélisation transforme les données CSV en nœuds RDF exploitables, permettant de relier les matchs aux joueurs, aux buts et aux équipes, et de réaliser des requêtes SPARQL analytiques.

```
schema > data > match.ttl
1  @prefix dbo: <http://dbpedia.org/ontology/> .
2  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
3  @prefix ex: <http://example.org/football/> .
4  @prefix dbr: <http://dbpedia.org/resource/> .
5  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
6
7  ex:Match_1872-11-30_Scotland_England
8      rdf:type                ex:Match ;
9      dbo:date                "1872-11-30"^^xsd:date ;
10     dbo:homeTeam            dbr:Scotland ;
11     dbo:awayTeam            dbr:England ;
12     dbo:tournament          dbr:Friendly_match ;
13     dbo:homeScore            0 ;
14     dbo:awayScore            0 ;
15     ex:result                ex:Draw ;
16     dbo:country              dbr:Scotland .
17
18  ex:Match_1873-03-08_England_Scotland
19      rdf:type                ex:Match ;
20      dbo:date                "1873-03-08"^^xsd:date ;
21      dbo:homeTeam            dbr:England ;
```

Le fichier de transformation correspondant est **extract_match.rq** et le résultat généré se trouve dans **./schema/data/match.ttl**.

2.2.2 Génération des buts (Goals)

Chaque but est représenté par une ressource RDF unique de type `ex:Goal`, avec les propriétés suivantes :

- **ex:scoredBy** : joueur ayant marqué le but, identifié par une URI locale unique ;
- **ex:scoredIn** : match dans lequel le but a été inscrit, relié à `ex:Match` ;
- **ex:goalMinute** : minute du but, typée en `xsd:integer` ;
- **ex:penalty** : indication précisant s'il s'agit d'un penalty.

Cette modélisation transforme les données CSV en nœuds RDF fins, permettant de relier chaque événement de jeu aux joueurs et aux matchs correspondants, et de réaliser des requêtes SPARQL précises.

```
schema > data > ≡ goal.ttl
1  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
2  @prefix ex: <http://example.org/football/> .
3  @prefix dbr: <http://dbpedia.org/resource/> .
4  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
5
6  ex:Goal_José_Piendibene_44
7      rdf:type          ex:Goal ;
8      ex:scoredBy       ex:José_Piendibene ;
9      ex:scoredIn       ex:Match_1916-07-02_Chile_Uruguay ;
10     ex:goalMinute     44 ;
11     ex:penalty        "FALSE" .
12
13  ex:Goal_Isabelino_Gradín_55
14      rdf:type          ex:Goal ;
15      ex:scoredBy       ex:Isabelino_Gradín ;
16      ex:scoredIn       ex:Match_1916-07-02_Chile_Uruguay ;
17      ex:goalMinute     55 ;
18      ex:penalty        "FALSE" .
```

Le fichier de transformation correspondant est **extract_goal.rq** et le résultat généré se trouve dans **./schema/data/goals.ttl**.

2.2.4 Création et enrichissement des joueurs

a) Extraction initiale des joueurs

Chaque joueur est tout d'abord extrait du fichier CSV des buts grâce à une requête TARQL, qui identifie les buteurs et les transforme en ressources RDF de type `ex:Footballeur`. Cette étape permet de créer les entités locales dans le graphe avec des URI uniques, tout en les connectant aux matchs et aux buts correspondants.

b) Exploration des propriétés disponibles sur DBpedia

Avant l'enrichissement, une requête exploratoire sur DBpedia a été réalisée afin d'identifier les propriétés pertinentes pour chaque joueur.

Par exemple, pour Lionel Messi :

```
queries > creation_graphe > ≡ player_properties.rq
1  PREFIX dbr: <http://dbpedia.org/resource/>
2
3  SELECT DISTINCT ?property
4  WHERE {
5      SERVICE <https://dbpedia.org/sparql> {
6          | dbr:Lionel_Messi ?property ?value .
7      }
8  }
9  ORDER BY ?property
10 |
```

Cette étape permet de repérer les propriétés biographiques et sportives disponibles, comme la date et le lieu de naissance, le poste, les clubs, ou la date et le lieu de décès le cas échéant.

c) Enrichissement progressif des joueurs

L'enrichissement effectif des joueurs a été réalisé avec un script Python (`extract.py`), qui :

1. Charge le graphe RDF local contenant les joueurs extraits du CSV.
2. interroge DBpedia via son endpoint SPARQL pour récupérer les propriétés manquantes.
3. Divise les requêtes en lots de 100 joueurs pour respecter les limites imposées par DBpedia.
4. Introduit des délais (`sleep`) entre les requêtes pour éviter d'être bloqué par le serveur.
5. Ajoute progressivement les propriétés récupérées au graphe RDF local.
6. Génère un fichier RDF final enrichi (`player_enriched.ttl`).

7. Propriétés ajoutées aux joueurs

Grâce à cette approche, chaque joueur est enrichi avec :

- **dbo:NationalTeam** : équipe nationale du joueur, reliée à DBpedia ;
- **dbo:birthDate** et **dbo:birthPlace** : date et lieu de naissance ;
- **dbo:position** : poste du joueur ;
- **dbo:team** : clubs fréquentés ;
- **dbo:deathDate** et **dbo:deathPlace** : date et lieu de décès, lorsque disponibles.

Les joueurs sont désormais pleinement intégrés au graphe RDF : ils sont connectés aux matchs, aux buts et aux équipes, et leur description est complétée avec des informations fiables provenant de DBpedia.

```
schema > data > ≡ players_enriched.ttl
1  @prefix dbo: <http://dbpedia.org/ontology/> .
2  @prefix ex: <http://example.org/football/> .
3  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
4
5  ex:Adolfo_Baloncieri a ex:Footballer ;
6      dbo:NationalTeam <http://dbpedia.org/resource/Italy> ;
7      dbo:birthDate "1897-07-27"^^xsd:date ;
8      dbo:birthPlace <http://dbpedia.org/resource/Alessandria>,
9          <http://dbpedia.org/resource/Italy_national_football_team> ;
10     dbo:deathDate "1986-07-23"^^xsd:date ;
11     dbo:deathPlace <http://dbpedia.org/resource/Genoa>,
12         <http://dbpedia.org/resource/Italy_national_football_team> ;
13     dbo:position <http://dbpedia.org/resource/Midfielder> ;
14     dbo:team <http://dbpedia.org/resource/Como_1907>,
15         <http://dbpedia.org/resource/Italy_national_football_team>,
16         <http://dbpedia.org/resource/Torino_FC>,
17         <http://dbpedia.org/resource/US_Alessandria_Calcio_1912> .
```

Le fichier de transformation correspondant est **extract_player.rq** et le résultat généré se trouve dans **./schema/data/player_enriched.ttl**.

2.2.5 Illustration d'un sous-graphe RDF

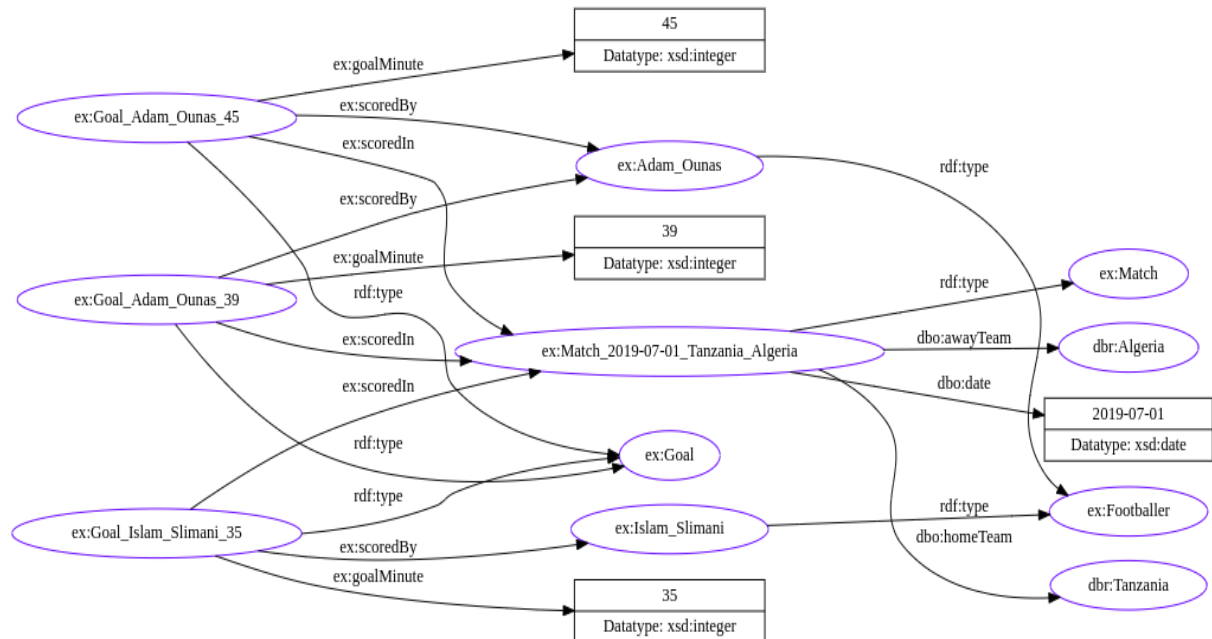
Pour illustrer la structure du graphe RDF et la façon dont les entités sont reliées, un sous-graphe a été extrait pour un match spécifique entre la Tanzanie et l'Algérie (le 1er juillet 2019). Ce sous-graphe inclut :

- le match en tant que ressource **ex:Match_2019-07-01_Tanzania_Algeria** ;
- les buts marqués lors du match (**ex:Goal_Adam_Ounas_39**, **ex:Goal_Islam_Slimani_35**, etc.) ;
- les joueurs ayant marqué (**ex:Adam_Ounas**, **ex:Islam_Slimani**).

Ce sous-graphe a été généré à partir du fichier TTL du projet à l'aide de la commande suivante :

```
$ rapper -i turtle -o dot schema/all.ttl \ | dot -Tsvg > schema/football.svg
```

Pour ne représenter qu'un match et ses buts, une requête SPARQL de type CONSTRUCT a été appliquée, filtrant l'URI du match concerné et incluant les joueurs et buts associés.



3. Requêtes SPARQL sur le graphe RDF

Dans cette section, nous présentons plusieurs requêtes SPARQL permettant d'interroger notre graphe RDF sur le football international. Ces requêtes exploitent les relations entre matchs, buts et joueurs, et montrent la richesse analytique du graphe.

Requête 1 : Statistiques de buts de Lionel Messi depuis 2000

```

queries > requete > requete1.sparql
1  PREFIX ex: <http://example.org/football/>
2  PREFIX dbo: <http://dbpedia.org/ontology/>
3  PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4
5  SELECT (COUNT(?goal) AS ?totalGoals)
6         (SUM(IF(?penalty = "TRUE", 1, 0)) AS ?penaltyGoals)
7         (SUM(IF(?penalty = "FALSE", 1, 0)) AS ?nonPenaltyGoals)
8  WHERE {
9      ?goal a ex:Goal ;
10         ex:scoredBy ?player ;
11         ex:scoredIn ?match ;
12         ex:penalty ?penalty .
13
14      ?match dbo:date ?date .
15
16      FILTER (?date >= "2000-01-01"^^xsd:date)
17      FILTER (?player = ex:Lionel_Messi)
18  }
19

```

Cette requête calcule :

- le nombre total de buts marqués par Lionel Messi depuis l'année 2000,
- le nombre de buts sur penalty,
- le nombre de buts non penalty.

Elle repose sur les relations entre les entités Goal, Player et Match et effectue des agrégations (COUNT, SUM) pour produire des statistiques synthétiques.

totalGoals	penaltyGoals	nonPenaltyGoals
63	18	45

Cette requête illustre l'intérêt des agrégations SPARQL pour obtenir des statistiques précises sur les joueurs.

Requête 2 : Nombre de défenseurs ayant marqué pour l'Uruguay

Cette requête identifie les joueurs défensifs (position = Defender) qui ont marqué au moins un but pour l'équipe nationale de l'Uruguay. Elle combine les données locales sur les joueurs, leurs positions et les buts inscrits.

```
queries > requete > ≡ requete2.sparql
1  PREFIX ex: <http://example.org/football/>
2  PREFIX dbo: <http://dbpedia.org/ontology/>
3  PREFIX dbr: <http://dbpedia.org/resource/>
4
5  SELECT (COUNT(DISTINCT ?player) AS ?nbDefenseursButteursPourUruguay)
6  WHERE {
7    ?player a ex:Footballer ;
8            dbo:position dbr:Defender ;
9            dbo:team dbr:Uruguay_national_football_team .
10
11    ?goal ex:scoredBy ?player .
12  }
13  |
```

nbDefenseursButteursPour
1

Cette requête montre comment combiner plusieurs propriétés RDF pour répondre à des questions ciblées sur les joueurs et les matchs.

Requête 3 : Joueurs ayant marqué en qualifications mais pas en phase finale

Cette requête permet de sélectionner les joueurs français qui ont marqué des buts en phase de qualification pour la Coupe du Monde mais pas lors de la phase finale. C'est utile pour identifier des joueurs ayant performé dans les qualifications mais pas dans le tournoi principal.

1. On sélectionne tous les buts (ex:Goal) marqués par un joueur dans les matchs de qualifications (dbr:FIFA_World_Cup_qualification).
2. On restreint les joueurs à ceux de l'équipe de France.
3. On utilise l'opérateur MINUS pour exclure tous les joueurs qui ont marqué lors de la phase finale (dbr:FIFA_World_Cup).
4. La clause DISTINCT permet de ne garder chaque joueur qu'une seule fois, même s'il a marqué plusieurs buts.

```
queries > requete > requete3.sparql
1  PREFIX ex: <http://example.org/football/>
2  PREFIX dbo: <http://dbpedia.org/ontology/>
3  PREFIX dbr: <http://dbpedia.org/resource/>
4
5  SELECT DISTINCT ?player
6  WHERE {
7      # Buts en qualifications
8      ?goal a ex:Goal ;
9          ex:scoredBy ?player ;
10         ex:scoredIn ?match .
11     ?match dbo:tournament dbr:FIFA_World_Cup_qualification .
12
13     # Joueurs français
14     ?player dbo:NationalTeam dbr:France .
15
16     # MAIS PAS en Coupe du Monde (phase finale)
17     MINUS {
18         ?goal2 a ex:Goal ;
19             ex:scoredBy ?player ;
20             ex:scoredIn ?match2 .
21         ?match2 dbo:tournament dbr:FIFA_World_Cup .
22     }
23 }
```

Résultat

player
ex:Jean-Jacques_Marcel
ex:Lucien_Cossou
ex:Édouard_Kargu
ex:Pierre_Flamion
ex:Raymond_Cicci
ex:Léon_Glovacki
ex:Henri_Baillot
ex:Marius_Walter
ex:Jean_Luciano
ex:Joseph_Ujlaki
ex:Thadée_Cisowski
ex:Ernest_Schultz
ex:Jacques_Faivre
ex:Célestin_Oliver
ex:René_Dereuddre
ex:Said_Brahimi
ex:Armand_Penverne
ex:Jean_Desgranges
ex:Jacques_Foix
ex:Ernest_Liberati
ex:Alfred_Aston

player	birthPlace	birthCountry	team
ex:Aleksandar Tirnanić	dbr:Krnjevo	dbr:Serbia	dbr:Yugoslavia
ex:Bora Kostić	dbr:Obrenovac	dbr:Serbia	dbr:Yugoslavia
ex:Dražan Jerković	dbr:Šibenik	dbr:Croatia	dbr:Yugoslavia
ex:Eddie McMorran	dbr:Larne	dbr:United_Kingdom	dbr:Northern_Ireland
ex:Hans-Peter Friedländer	dbr:Berlin	dbr:Germany	dbr:Switzerland
ex:Heinrich Uukkivi	dbr:Governorate_of_Estonia	dbr:Russian_Empire	dbr:Estonia
ex:Houssein Aouar	dbr:Lyon	dbr:France	dbr:Algeria
ex:Jackie Mudie	dbr:Dundee	dbr:United_Kingdom	dbr:Scotland
ex:John Charles	dbr:Swansea	dbr:United_Kingdom	dbr:Wales
ex:Josef Kadřaba	dbr:Revničov	dbr:Czech_Republic	dbr:Czechoslovakia
ex:Josef Masopust	<http://dbpedia.org/resource/Most_(city)>	dbr:Czech_Republic	dbr:Czechoslovakia
ex:Manuel Piñeiro	dbr:Matanzas	dbr:Cuba	dbr:Chile
ex:Oldřich Nejedlý	dbr:Žebrák	dbr:Czech_Republic	dbr:Czechoslovakia
ex:Ron Flowers	dbr:Doncaster	dbr:United_Kingdom	dbr:England
ex:Ron Flowers	dbr:Edlington	dbr:United_Kingdom	dbr:England
ex:Tadeusz Kraus	dbr:Třinec	dbr:Czech_Republic	dbr:Czechoslovakia
ex:Wilf Mannion	<http://dbpedia.org/resource/South_Bank,_Redcar_and_Cleveland>	dbr:United_Kingdom	dbr:England

Requête sur graphe nommé :

```
queries > requete > ≡ requete_sur_graphs.sparql
1  PREFIX football: <http://example.org/football/>
2
3  SELECT ?player (COUNT(?goal) AS ?nb)
4  WHERE {
5      GRAPH <http://example.org/g/goals> {
6          ?goal a football:Goal ;
7          | | | football:scoredBy ?player .
8      }
9  }
10 GROUP BY ?player
11 ORDER BY DESC(?nb)
12 LIMIT 10
13
```

Résultat :

```
(base) PS C:\Users\lazra\Desktop\M2\BDD_Spe\Projet1\projet_RDF> tdb2_tdbquery --loc tdb --query
sparql
=====
| player | nb |
=====
| football:Cristiano_Ronaldo | 57 |
| football:Robert_Lewandowski | 45 |
| football:Romelu_Lukaku | 43 |
| football:Ali_Daei | 42 |
| football:Lionel_Messi | 41 |
| football:Harry_Kane | 38 |
| football:Miroslav_Klose | 38 |
| football:Edin_Džeko | 37 |
| football:Zlatan_Ibrahimović | 37 |
| football:Aleksandar_Mitrović | 36 |
=====
(base) PS C:\Users\lazra\Desktop\M2\BDD_Spe\Projet1\projet_RDF> |
```

Requête multi graphe :

```
1  PREFIX football: <http://example.org/football/>
2
3  SELECT ?match (COUNT(?goal) AS ?nbGoals)
4  WHERE {
5      GRAPH <http://example.org/g/goals> {
6          ?goal a football:Goal ;
7          | | | football:scoredIn ?match .
8      }
9      GRAPH <http://example.org/g/matches> {
10         ?match ?p ?o .
11     }
12 }
13 GROUP BY ?match
14 ORDER BY DESC(?nbGoals)
15 LIMIT 10
```

Réponse :

```
(base) PS C:\Users\lazra\Desktop\M2\BDD_Spe\Projet1\projet_RDF> tdb2_tdbquery --loc tdb
e.sparql
=====
| match | nbGoals |
=====
| football:Match_2001-04-11_Australia_American_Samoa | 279 |
| football:Match_2001-04-09_Australia_Tonga | 198 |
| football:Match_2000-11-24_Iran_Guam | 171 |
| football:Match_1997-06-02_Maldives_Iran | 153 |
| football:Match_2000-06-19_Australia_Cook_Islands | 153 |
| football:Match_1998-09-28_Australia_Cook_Islands | 144 |
| football:Match_2000-11-26_Tajikistan_Guam | 144 |
| football:Match_2015-09-03_Qatar_Bhutan | 135 |
| football:Match_1928-06-09_Italy_Egypt | 126 |
| football:Match_1996-11-17_Honduras_Saint_Vincent_and_the_Grenadines | 126 |
=====
(base) PS C:\Users\lazra\Desktop\M2\BDD_Spe\Projet1\projet_RDF> |
```

Requête optional:

```
1 PREFIX ex: <http://example.org/football/>
2 PREFIX dbo: <http://dbpedia.org/ontology/>
3 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4
5 SELECT ?player
6         (SAMPLE(?nt) AS ?nationalTeam)
7         (SAMPLE(?bp) AS ?birthPlace)
8         (SAMPLE(?bd) AS ?birthDate)
9 WHERE {
10     ?player ?p ?o .
11
12     OPTIONAL {
13         ?player rdf:type ex:Footballer .
14         OPTIONAL { ?player dbo:NationalTeam ?nt . }
15         OPTIONAL { ?player dbo:birthPlace ?bp . }
16         OPTIONAL { ?player dbo:birthDate ?bd . }
17     }
18 }
19 GROUP BY ?player
20 LIMIT 20
```

player	nationalTeam	birthPlace	birthDate
ex:Ibrahima_Tall	<http://dbpedia.org/resource/Burkina_Faso>		
ex:Juste_Brouzes	<http://dbpedia.org/resource/France>	<http://dbpedia.org/resource/France_national_football_team>	*1894-01-28***<http://www.w3.org/2001/XMLSchema#date>
ex:Gaggy_Coulibaly	<http://dbpedia.org/resource/Mauritania>		
ex:Heisl_Kouamé	<http://dbpedia.org/resource/Ivory_Coast>		
ex:Umas_Kirs	<http://dbpedia.org/resource/Estonia>		
ex:Fahad_Al-Bishi	<http://dbpedia.org/resource/Saudi_Arabia>		
ex:El_Hadji_Oumar_Guzye	<http://dbpedia.org/resource/Senegal>		
ex:Johann_Horvath	<http://dbpedia.org/resource/Austria>	<http://dbpedia.org/resource/Austria-Hungary>	*1983-05-28***<http://www.w3.org/2001/XMLSchema#date>
ex:Vasilis_Dimitriadis	<http://dbpedia.org/resource/Greece>		
ex:Wilson_Guapar	<http://dbpedia.org/resource/Angola>		
ex:Antonijs_Sernomordijs	<http://dbpedia.org/resource/Latvia>		
ex:Armando_Cruz	<http://dbpedia.org/resource/Cuba>		
ex:Hao_Haidong	<http://dbpedia.org/resource/China_PR>		
ex:Badar_Al-Maimani	<http://dbpedia.org/resource/Oman>		
ex:Hakan_Kalhanoglu	<http://dbpedia.org/resource/Turkey>		
ex:Sekou_Cissé	<http://dbpedia.org/resource/Ivory_Coast>		
ex:Rubén_Sosa	<http://dbpedia.org/resource/Uruguay>		
ex:Houmane_Jarir	<http://dbpedia.org/resource/Morocco>		
ex:Kara_Mbodji	<http://dbpedia.org/resource/Senegal>		
ex:Tomas_Brolin	<http://dbpedia.org/resource/Sweden>		

Requête sur les chemins :

```

1 PREFIX football: <http://example.org/football/>
2
3 SELECT DISTINCT ?match ?p1 ?p2
4 WHERE {
5     GRAPH <http://example.org/g/goals> {
6         ?p1 ( ^football:scoredBy / football:scoredIn ) ?match .
7         ?p2 ( ^football:scoredBy / football:scoredIn ) ?match .
8     }
9     FILTER(?p1 != ?p2)
10    FILTER(STR(?p1) < STR(?p2))
11 }
12 LIMIT 20

```

4. L'ontologie :

4.1 La mise en place :

Vocabulaire utilisé :

Notre but est de structurer et d'unifier les différents jeux de données. Plutôt que de redéfinir toutes les propriétés, l'ontologie réutilise des propriétés du vocabulaire DBpedia (dbo:) lorsque cela est pertinent, notamment pour les équipes, les dates, les informations biographiques des joueurs...

Les classes:

Afin de centraliser toutes nos classes et qu'elles aient la même racine nous avons posé Entity comme classe mère, toutes les autres classes en sont des sous classes.

L'ontologie définit les classes suivantes dans le namespace ex :

Ces classes correspondent directement aux entités manipulées dans les fichiers RDF.

Classe	SubclassOf
Entity	—
Person	Entity
Footballer	Person
Match	Entity
Goal	Entity
Team	Entity
NationalTeam	Team
ClubTeam	Team
Tournament	Entity
Country	Entity
Place	Entity
Position	Entity

Propriétés :

- Les principales:
Plusieurs propriétés relient ces classes entre elles et aux vocabulaires externes.
ex:scoredBy relie un ex:Goal à un ex:Footballer.
ex:scoredIn relie un ex:Goal à un ex:Match.
dbo:homeTeam et dbo:awayTeam relient un ex:Match à une ex:Team.
dbo:NationalTeam, dbo:birthDate, dbo:position permettent l'enrichissement des joueurs à partir de DBpedia.
- Les propriétés de données:
ex:goalMinute
ex:penalty
dbo:date

Les sous propriétés:

Nous avons mis en place quatre sous-propriétés dans notre ontologie afin de factoriser certaines relations tout en conservant une sémantique correcte du domaine.

La sous-propriété `dbo:team` \sqsubseteq `ex:playsFor` permet de généraliser la relation entre un joueur et une équipe de club, et d'interroger de manière unifiée l'ensemble des équipes pour lesquelles un joueur a évolué.

La sous-propriété `dbo:NationalTeam` \sqsubseteq `ex:playsFor` étend cette généralisation aux équipes nationales, ce qui permet de regrouper clubs et sélections nationales sous une même relation abstraite sans perte de sens.

La sous-propriété `dbo:homeTeam` \sqsubseteq `ex:involvesTeam` permet de représenter l'implication d'une équipe dans un match indépendamment de son rôle spécifique d'équipe à domicile.

La sous-propriété `dbo:awayTeam` \sqsubseteq `ex:involvesTeam` complète cette modélisation en représentant l'implication d'une équipe à l'extérieur, tout en restant cohérente avec la relation générique définie pour les matchs.

Nous sommes conscients des limites de notre modélisation, notamment du fait que l'ontologie ne capture pas l'ensemble des relations temporelles ou contractuelles (durée d'un transfert, période de sélection, etc.).

Nous avons envisagé d'introduire d'autres sous-propriétés, comme une relation dérivée du lieu de naissance (`dbo:birthPlace`) pour inférer une appartenance nationale potentielle, mais cette approche a été volontairement écartée car elle introduirait une sémantique incorrecte : le lieu de naissance d'un joueur ne détermine pas nécessairement l'équipe nationale pour laquelle il évolue.

Les contraintes:

Enfin nous avons utilisé range/domain pour préciser le sens des relations:

L'ontologie utilise des axiomes RDFS simples pour préciser le sens des relations :

ex:scoredBy domaine : ex:Goal,range : ex:Footballer

ex:scoredIn domaine : ex:Goal range : ex:Match

dbo:homeTeam, dbo:awayTeam domaine : ex:Match range : ex:Team

4.2 Raisonnement RDFS:

Requête simple:

Joueurs qui ont marqué un but et qui sont reconnus comme **ex:Person**

```
PREFIX ex: <http://example.org/football/>
```

```
PREFIX dbo: <http://dbpedia.org/ontology/>
```

```
SELECT DISTINCT ?player
```

```
WHERE {
```

```
    ?goal ex:scoredBy ?player .
```

```
    ?player a ex:Person .
```

```
}
```

```
LIMIT 20
```

SANS RDFS

Aucun Player n'est reconnu comme Person

AVEC RDFS

```
(base) \M2\BDD_Spe\Projet1\projet_RDF> python scripts/rdfs_sparql.py `
```

```
>> --schema schema/ontology/ontology.ttl `
```

```
>> --data schema/data/player.ttl `
```

```
>> --data schema/data/match.ttl `
```

```
>> --data schema/data/goal.ttl `
```

```
>> --query queries/requete/requete_exploration_rdfs.sparql
```

```
player
```

```
http://example.org/football/Saoud_AI_Gassem_Mohammed_Bo_Saeed
```

```
http://example.org/football/Rodrigo_Cordero
```

```
http://example.org/football/Ryang_Yong-gi
```

```
http://example.org/football/Raúl_Servín
```

```
http://example.org/football/Zanzan_Atte-Oudeyi
```


http://example.org/football/Luc_Holtz
http://example.org/football/Erkan_Zengin
http://example.org/football/José_Luis_Sierra

Requête sur les subpropriétés :

Requête simple afin de récupérer les couples joueur–équipe en utilisant uniquement la propriété générique

```
1 PREFIX ex: <http://example.org/football/>
2 PREFIX dbo: <http://dbpedia.org/ontology/>
3
4 SELECT DISTINCT ?player ?team
5 WHERE {
6   ?player ex:playsFor ?team .
7 }
8 LIMIT 30
9
```

Sans RDFS, aucun résultat avec RDFS :

(base) PS C:\Users\lazra\Desktop\M2\BDD_Spe\Projet1\projet_RDF> python
scripts/rdfs_sparql.py`

```
>> --schema schema/ontology/ontology.ttl `
>> --data schema/data/player.ttl `
>> --data schema/data/match.ttl `
>> --data schema/data/goal.ttl `
>> --query .\queries\requete\requete_rdfs_subprop.sparql
```

player team

http://example.org/football/Gustavo_Gómez <http://dbpedia.org/resource/Peru>
http://example.org/football/César_Socarraz <http://dbpedia.org/resource/Peru>
http://example.org/football/Sergio_Peña <http://dbpedia.org/resource/Peru>
http://example.org/football/Juan_Pajuelo <http://dbpedia.org/resource/Peru>
http://example.org/football/Jefferson_Farfán <http://dbpedia.org/resource/Peru>
http://example.org/football/Manuel_Drago <http://dbpedia.org/resource/Peru>
http://example.org/football/Juan_Fuenmayor <http://dbpedia.org/resource/Peru>
http://example.org/football/Miguel_Ángel_Loayza <http://dbpedia.org/resource/Peru>
http://example.org/football/Darío_Muchotrigo <http://dbpedia.org/resource/Peru>
http://example.org/football/Manuel_Marengo <http://dbpedia.org/resource/Peru>
http://example.org/football/Adolfo_Magallanes <http://dbpedia.org/resource/Peru>
http://example.org/football/Robert_Cavallo <http://dbpedia.org/resource/Peru>
http://example.org/football/Héctor_Bailetti <http://dbpedia.org/resource/Peru>
http://example.org/football/Juan_Emilio_Salinas <http://dbpedia.org/resource/Peru>
http://example.org/football/Luis_Guzmán <http://dbpedia.org/resource/Peru>

http://example.org/football/Eduardo_Malásquez <http://dbpedia.org/resource/Peru>
http://example.org/football/Isaac_Andrade <http://dbpedia.org/resource/Peru>
http://example.org/football/Pedro_Alejandro_García <http://dbpedia.org/resource/Peru>
http://example.org/football/Luis_Guzmán_Gonzales <http://dbpedia.org/resource/Peru>

Requête sur les contraintes :

Cette requête récupère des couples (personne, lieu) et vérifie que la personne est reconnue comme une ex:Person et le lieu comme une ex:Place.

```

1  PREFIX ex: <http://example.org/football/>
2
3  SELECT DISTINCT ?player ?team ?match
4  WHERE {
5      #lien Goal->Player+Goal->Match (datasets goal+match)
6      ?goal ex:scoredBy ?player ;
7      ex:scoredIn ?match .
8
9      ?player ex:playsFor ?team .
10
11     #raisonnement (subclass+domain/range)
12     ?player a ex:Footballer .
13     ?team a ex:Team .
14     ?match a ex:Match .
15 }
16 LIMIT 10

```

AVEC RDFS :

player	team	match
ex:Ģirts_Karlsons	dbo:Latvia	ex:Match_2007-11-17_Latvia_Liechtenstein
ex:Ģirts_Karlsons	dbo:Latvia	ex:Match_2008-09-06_Moldova_Latvia
ex:Ģirts_Karlsons	dbo:Latvia	ex:Match_2009-03-28_Luxembourg_Latvia
ex:Ģirts_Karlsons	dbo:Latvia	ex:Match_2006-10-07_Latvia_Iceland
ex:Riyadh_Nouri	dbo:Iraq	ex:Match_1973-03-21_Iraq_Indonesia
ex:Riyadh_Nouri	dbo:Iraq	ex:Match_1973-03-11_Australia_Iraq
ex:Maksim_Zhalmagambetov	dbo:Kazakhstan	ex:Match_2005-09-07_Kazakhstan_Greece
ex:Bent_Christensen_Arensøe	dbo:Denmark	ex:Match_1991-06-05_Denmark_Austria
ex:Bent_Christensen_Arensøe	dbo:Denmark	ex:Match_1991-09-25_Faroe_Islands_Denmark
ex:Bent_Christensen_Arensøe	dbo:Denmark	ex:Match_1991-05-01_Yugoslavia_Denmark

5 Conclusion :

Ce travail nous a permis de mieux comprendre les apports, mais aussi les limites, du raisonnement RDFS appliqué à des jeux de données RDF. Les données utilisées autour des entités player, match et goal sont volontairement réduites et incomplètes. Il s'agit avant tout d'un jeu de test, conçu pour valider la structuration du schéma, l'ontologie associée et les mécanismes de raisonnement, et non d'un jeu de données entièrement représentatif.

L'utilisation du raisonnement RDFS a clairement enrichi l'exploitation des données. Les hiérarchies de classes et de propriétés définies dans l'ontologie permettent d'unifier des relations exprimées différemment dans les fichiers RDF et de faire apparaître des informations qui ne sont pas explicitement présentes dans les données brutes. Les requêtes menées illustrent bien cet apport, notamment dans l'analyse des relations entre joueurs, équipes et matchs.

Ce projet a également mis en évidence plusieurs difficultés pratiques. Certaines données textuelles, notamment des noms en alphabet cyrillique ou turc, ne sont pas correctement interprétées en UTF-8, ce qui provoque des incohérences à l'affichage. Ces problèmes n'ont pas été corrigés afin de préserver la correspondance avec les ressources d'origine de DBpedia, un choix nécessaire pour maintenir la cohérence sémantique.

Enfin, malgré son intérêt, le raisonnement RDFS reste limité par sa simplicité. Il permet principalement de traiter des inférences et des requêtes simples, et ne couvre pas des raisonnements plus complexes, d'autant plus que le jeu de données utilisé reste minimal. Des améliorations sont possibles, notamment par l'enrichissement des données et l'extension de l'ontologie vers OWL, afin d'explorer des mécanismes de raisonnement plus avancés.

Pour la réalisation de ce projet, nous nous sommes appuyés sur plusieurs supports, en particulier le cours, ainsi que la chaîne YouTube [Stardog](#) et sa playlist dédiée à RDF. Nous avons également utilisé des outils de type LLM pour corriger certaines requêtes SPARQL et pour nous aider à mettre en place des éléments techniques, comme le script `rdfs_sparql.py`.