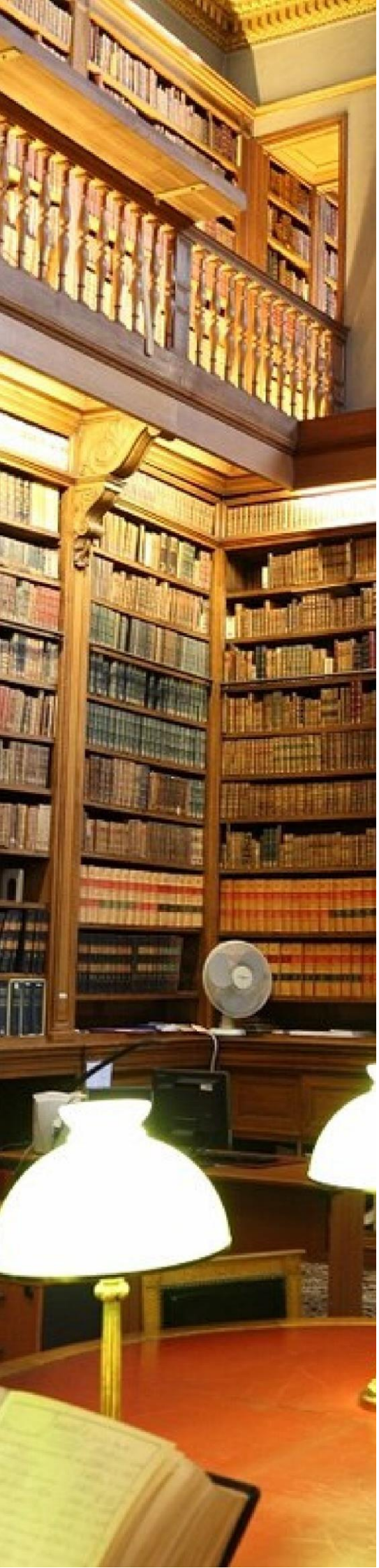


ESILV LÉONARD DE VINCI | ÉCOLE D'INGÉNIEURS PARIS-LA DÉFENSE

RAPPORT PROJET

Webdataming & Semantics

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

2022

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

Table des matières

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

1. Présentation du sujet

Ce projet est un exercice consistant à intégrer toutes ou une partie des notions que nous avons vues durant nos cours de Web Datamining & Semantics, dans la construction d'une application. En effet, nous avons pour objectif de faire une application, Web ou autre, intégrant des données géospatiales depuis 2 sources, ou plus. Nous avons alors pour but de concevoir et mettre en place une application avec des outils vus en classe (ontologie, JSONLD, SPARQL…), et ainsi permettre à un utilisateur de pouvoir interagir avec nos données.

De ce fait, nous avons choisi de réaliser une application permettant à un utilisateur, comme un étudiant, de pouvoir localiser où et à quelle distance se trouve les bibliothèques aux alentours de son établissement scolaire.

2. Démarches et Codes

1. Importation des sources

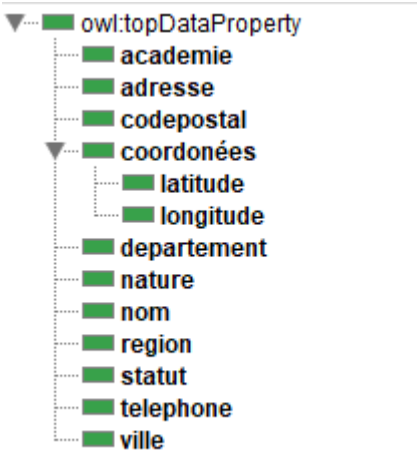
Tout d'abord, nous avons commencé par choisir sur quels datasets nous nous sommes appuyés pour réaliser le projet. Nous donc choisi les 3 datasets « Universities and higher education establishments », « Primary and secondary schools » et « Public libraries ». De ce fait, nous avions eu à les importer, les nettoyer, autrement dit, effectuer un preprocessing, tout cela sous Python soit :

- Régler les problèmes d'encodage
- Changer le format des départements pour uniformiser nos 3 sources
- Séparer les coordonnées latitude et longitude, afin de faciliter la manipulation de la localisation
- Ajouter un champs « classe » à chaque objet JSON de nos sources pour expliciter de quel type de donnée correspond tel objet d'une source

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

2. Création de l'ontologie

Ensuite, nous nous sommes tournés vers la création de notre ontologie sous Protege. Ci-dessous est la structure de notre ontologie



3. Création du JSONLD et conversion des sources

Puis, nous avons pu nous baser sur notre ontologie pour créer notre JSONLD qui nous a permis de :

- Uniformiser le noms des champs entre chaque dataset
- Retirer les champs non-utilisés dans notre ontologie
- Convertir nos sources JSON en format Turtle avec RDFLIB sous Python (Manipulable avec nos requêtes SPARQL)

Ainsi, nous avons pu stocker ces données dans Fuseki, étant un Triplestore. Cependant, nous n'avons pas pu importer nos données sous Python, de ce fait nous les avons importées manuellement. Ci-dessous, en est une illustration :

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

/projet

query | add data | edit | info

Available Services

Graph Store Protocol (Read)	/projet/
Graph Store Protocol (Read)	/projet/get
Graph Store Protocol	/projet/
Graph Store Protocol	/projet/data
SPARQL Query	/projet/query
SPARQL Query	/projet/
SPARQL Query	/projet/sparql
SPARQL Update	/projet/
SPARQL Update	/projet/update

Dataset size

count triples in all graphs

graph name	triples
default graph	1185025

Statistics

Endpoint	Requests	Good	Bad
Graph Store Protocol (Read)	0	0	0
Graph Store Protocol (Read) (get)	0	0	0
Graph Store Protocol	0	0	0
Graph Store Protocol (data)	3	3	0
SPARQL Query (query)	14	14	0
SPARQL Query	0	0	0
SPARQL Query (sparql)	2	2	0
SPARQL Update	0	0	0
SPARQL Update (update)	0	0	0
Overall	**19**	**19**	**0**

4. Création des requêtes SPARQL

Nous avons donc dû nous connecter à Fuseki pour pouvoir requêter sur nos données. Ces requêtes ont eu pour but de récupérer les informations qui nous semblaient fondamentales pour assurer le bon fonctionnement des services proposées par notre application. Les requêtes sont les suivantes :

- Récupérer toutes les régions pour lesquelles nous avons des établissements associés [Annexe 2]

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

- Récupérer les départements appartenant à la région précédemment sélectionnée [Annexe 3]
- Récupérer les villes appartenant au département précédemment sélectionné [Annexe 4]
- Récupérer les types d'écoles se situant dans la ville sélectionnée [Annexe 5]
- Récupérer le nom des écoles qui correspondent aux filtres précédemment sélectionnés [Annexe 6]
- Récupérer les coordonnées de cette même-école [Annexe 7]
- Récupérer les coordonnées de chaque bibliothèque [Annexe 8]
- Récupérer les informations des bibliothèques [Annexe 9]

5. Création de l'interface

Concernant l'interface, nous nous sommes reposés sur la libraire Streamlit qui nous a permis de coder notre application Web. Cette librairie permet de découper l'interface en plusieurs containers. Puis, pour l'affichage de la carte, nous avons utilisé la librairie Folium.

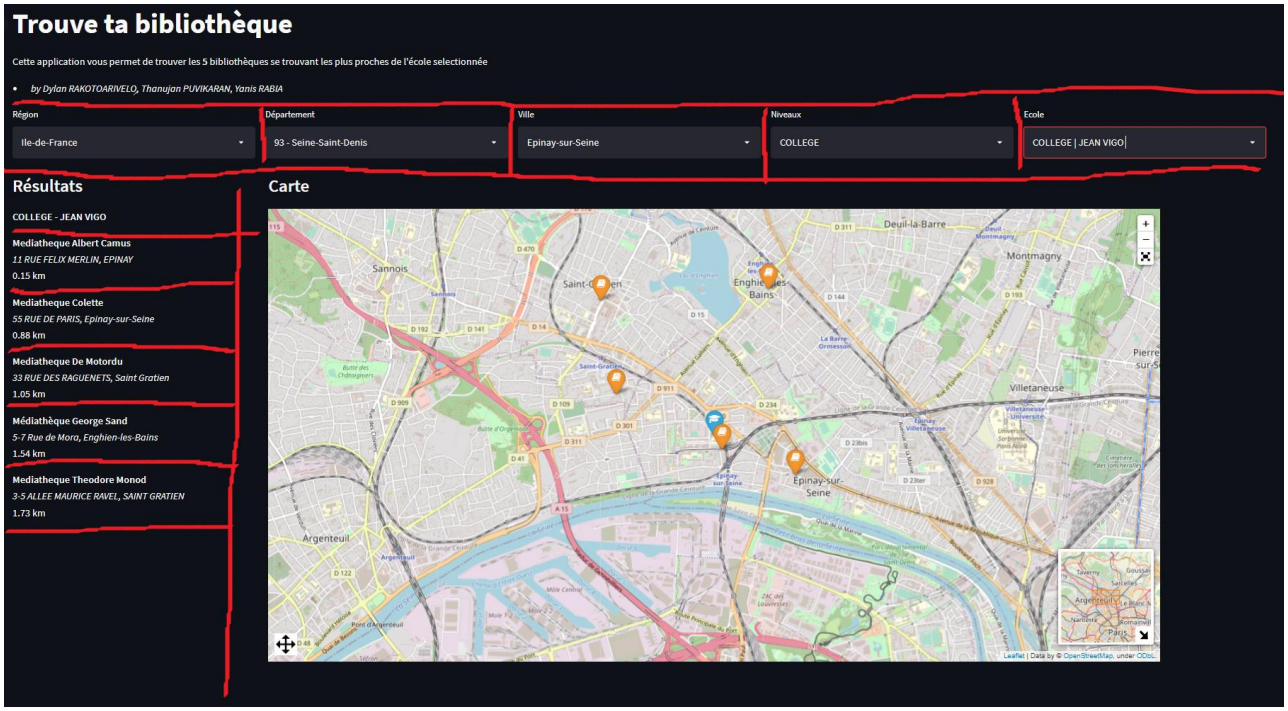
6. Autres fonctionnalités

Par ailleurs, nous avons mis en place une fonctionnalité qui affiche les 5 bibliothèques les plus proches de l'établissement scolaire, classées du plus proche au plus loin. Pour le calcul des distances, nous avons utilisé la librairie Haversine.

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

3. L'Application

Notre application demandera dans un premier temps à l'utilisateur d'indiquer séquentiellement la région, le département, la ville et le type (lycée, collège, université…) jusqu'à lui permettre de sélectionner son école. Ainsi, à partir des données qu'il aura sélectionnées, notre application vous affichera une carte dans laquelle la position des 5 bibliothèques les plus proches seront indiquées. Par ailleurs, sur la partie gauche de l'interface sera spécifiées les bibliothèques les plus proches de l'école sélectionnée, avec leur adresse ainsi que leur distance qui les séparent de l'école. Ci-dessous en est une illustration :



Afin de lancer l'application, vous pouvez suivre les instructions suivantes : README

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

4. Auto-évaluation

Parties	Résultats obtenus	Résultats en cours / non obtenus
Part 1	• 5 classes et 13 propriétés • 1 restriction (Disjonction retirée entre les différents établissements) • Réutilisation de l'ontologie	• Ajout de plus de restrictions sur les classes

Part 2	• 3 sources :
	  • 16 220 écoles supérieures
	  • 65 475 écoles secondaires
	  • 15 750 bibliothèques
	• 1 JSONLD pour chaque source/classe	

Part 3	• 14 requêtes SPARQL	• Aucune règle SWRL

Part 4	• Recherche d'une entité donnée par l'utilisateur
	• Fonctionnalités avancées :
	  • Calcul de distances entre 2 points
	  • Filtrage avec suggestions
	• Affichage des résultats dans l'interface	

Bonus	• Triple store (Fuseki)
	• GUI :
	  • Design global (Streamlit)
	  • Carte dynamique (Folium)
		• RDFA
		• Données mises à jour en temps réel

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

5. Conclusion

Ce projet nous a permis de nous entraîner à utiliser les outils que nous avons vu en cours tout au long de ce semestre tel que ontologie… De ce fait, dans le but de répondre à notre problématique, nous avons construit notre propre ontologie à partir de plusieurs datasets et réalisé pas à pas une application mettant en œuvre notre travail.

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

6. 5. Annexes

Annexes 1 : Lien github : [DATA_MINING_A4]

```python
def getRegions():

    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
    SELECT DISTINCT ?region
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:region ?region] .
    }
    """
    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))

    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?region
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:region ?region.
    }
    """
    qresUni = getValueFromJson(fuseki_query.run_sparql(querUni))

    regions = set(qresSecond+qresUni)
    return list(regions)
```

Annexe 2 : Récupération des régions

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

```python
def getDepartements(region):

    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
    SELECT DISTINCT ?departement
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:departement ?departement] .
        ?a projet:fields [projet:region "%s"^^<xsd:string> ] .
    }
    """%region
    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))


    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
    SELECT DISTINCT ?departement
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:departement ?departement .
        ?a projet:region "%s"^^<xsd:string> .
    }
    """%region
    qresUni = getValueFromJson(fuseki_query.run_sparql(querUni))

    departements = set(qresSecond+qresUni)

    return list(departements)
```

Annexe 3 : Récupération des départements

```python
def getVille(departement):

    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?ville
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:departement "%s"^^<xsd:string>] .
        ?a projet:fields [projet:ville ?ville  ] .
    }
    """%departement
    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))

    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?ville
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:departement "%s"^^<xsd:string> .
        ?a projet:ville ?ville .
    }
    """%departement
    qresUni =  getValueFromJson(fuseki_query.run_sparql(querUni))

    villes = set(qresSecond+qresUni)

    return list(villes)
```

Annexe 4 : Récupération des villes

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

```python
def getNatureEcole(ville):
    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?nature
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:ville "%s"^^<xsd:string>] .
        ?a projet:fields [projet:nature ?nature  ] .
    }
    """%ville
    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))

    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?nature
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:ville "%s"^^<xsd:string> .
        ?a projet:nature ?nature .
    }
    """%ville

    qresUni = getValueFromJson(fuseki_query.run_sparql(querUni))

    villes = set(qresSecond+qresUni)

    return list(villes)
```

Annexe 5 : Récupération des types d'école

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

```python
def getEcoles(ville,nature):
    ecoles = []

    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?nom
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:nature "%s"^^<xsd:string>] .
        ?a projet:fields [projet:ville "%s"^^<xsd:string> ] .
        ?a projet:fields [projet:nom ?nom ] .
    }
    """%(nature,ville)
    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))

    for x in qresSecond :
        ecoles.append(nature + " | " + x )
    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?nom
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:nature "%s"^^<xsd:string> .
        ?a projet:ville "%s"^^<xsd:string>  .
        ?a projet:nom ?nom  .
    }
    """%(nature,ville)

    qresUni = getValueFromJson(fuseki_query.run_sparql(querUni))

    for x in qresUni :
        ecoles.append(nature + " | " + x )

    return ecoles
```

Annexe 6 : Récupération des écoles

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

```python
def getCoordonneesEcole(ecole,ville):

    nom = ecole.split(" | ")[1]

    querSecond = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?latitude ?longitude
    WHERE {
        ?a rdf:type projet:Secondaire .
        ?a projet:fields [projet:nom "%s"^^<xsd:string> ] .
        ?a projet:fields [projet:ville "%s"^^<xsd:string> ] .
        ?a projet:fields [projet:longitude ?longitude] .
        ?a projet:fields [projet:latitude ?latitude] .
    }
    """%(nom,ville)

    qresSecond = getValueFromJson(fuseki_query.run_sparql(querSecond))
    querUni = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?latitude ?longitude
    WHERE {
        ?a rdf:type projet:Superieur .
        ?a projet:longitude ?longitude .
        ?a projet:latitude ?latitude .
        ?a projet:nom "%s"^^<xsd:string>  .
        ?a projet:ville "%s"^^<xsd:string>  .
    }
    """%(nom,ville)

    qresUni = getValueFromJson(fuseki_query.run_sparql(querUni))

    if len(qresSecond) != 0 :
        return qresSecond[0]
    else :
        return qresUni[0]
```

Annexe 7 : Récupération des coordonnées de l'école

```python
def getCordonneesBiblio():

    querBib = """
    PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
    PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

    SELECT DISTINCT ?latitude ?longitude
    WHERE {
        ?a rdf:type projet:Bibliotheque .
        ?a projet:longitude ?longitude .
        ?a projet:latitude ?latitude .
    }
    """

    coords = []
    qresBib = getValueFromJson(fuseki_query.run_sparql(querBib))
    return qresBib
```

Annexe 8 : Récupération des coordonnées de chaque bibliothèque

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6

```python
def getBibInfo(coords):
    biblios = []
    for coord in coords:
        querBib = """
        PREFIX projet: <http://www.semanticweb.org/ontologies/projet_webdataming#>
        PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

        SELECT DISTINCT ?nom ?adresse ?ville
        WHERE {
            ?a rdf:type projet:Bibliotheque .
            ?a projet:latitude "%s"^^<xsd:float> .
            ?a projet:longitude "%s"^^<xsd:float> .
            ?a projet:fields [projet:nom ?nom] .
            ?a projet:fields [projet:adresse ?adresse] .
            ?a projet:fields [projet:ville ?ville] .
            ?a projet:fields [projet:codepostal ?codepostal] .
        }
        """%(coord[0],coord[1])

        qresBib = getValueFromJson(fuseki_query.run_sparql(querBib))

        for x in qresBib:
            x.append(coord[0])
            x.append(coord[1])
            x.append(coord[2])

        biblios.append(qresBib[0])

    return biblios
```

Annexe 9 : Récupération des informations des bibliothèques

Dylan RAKOTOARIVELO
Thanujan PUVIKARAN
Yanis RABIA
Groupe : DIA6