# Hybrid Search with Multiple Full-text Indexes over RDF Graphs

Antoine Willerval[1,2] Dennis Diefenbach[1] Angela Bonifati[2]

[1]The QA Company

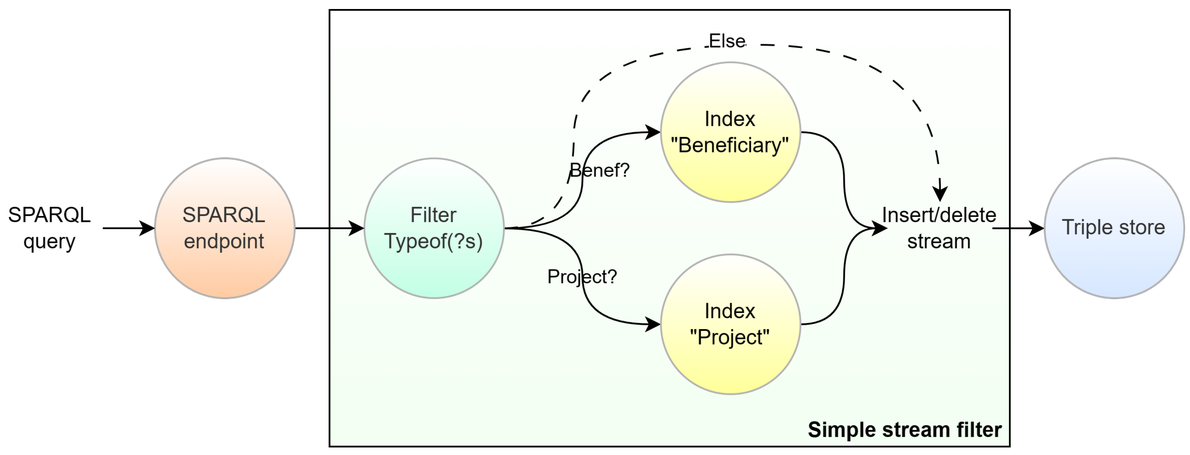
[2]Lyon 1 University, LIRIS

## Motivation and Goals

**How to use multiple full-text indexes over an RDF graph?**

- Indexing and querying RDF literals from a graph.
- Using multiple indexes tuned for different languages.
- Reduce index **size and query time** using specific indexes.

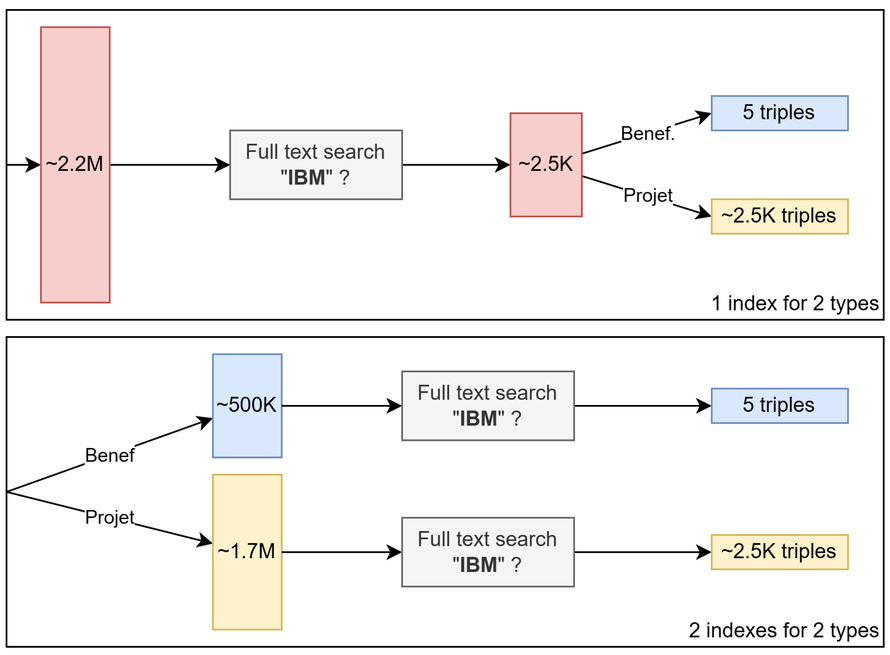
## Architecture




Integrate filtering to the RDF4J API to combine multiple indexes.

- Allows to easily configure and join multiple filters.
- Filter by rdf:type, literal language or type.
- Keep same interface as RDF4J for a fast replacement.

## Full-text queries differences




Example with the query "IBM" in a 2.2M literals dataset.

**Case 1**

```
SELECT ?s WHERE {
    ?s s:matches [
      s:query "IBM"
    ] .
    ?s a ex:Benef .

    # Anything on ?s ...
}
```

**Case 2**

```
SELECT ?s WHERE {
    ?s s:matches [
      s:query "IBM" ;
      s:indexid ex:IdxBenef
    ]

    # Anything on ?s ...
}
```

**1 - Using one index** The search is done on the index and then the intermediate values are filtered by the SPARQL engine.

**2 - Using multiple indexes** The right index is selected and then we do the search, reducing the intermediate computations.

## Full-text store capabilities comparison

| Store | full-text index | analyser | insert/update | multi-index |
|---|---|---|---|---|
| **qEndpoint** | Lucene | **yes** | **yes** | **yes** |
| RDF4J | Multiple[1] | yes | yes | no |
| Virtuoso | Native | no | yes | no |
| qLever | Native | no | read-only | no |
| Stardog | Lucene | yes | yes | no |
| Jena | Lucene | yes | yes | no |
| Blazegraph | Internal | no | yes | no |
| Blazegraph | Solr | yes | no | yes |

Only our system is able to provide a way to use multiple indexes in a update scenario. It also provides the ability to pick an analyser per index.

[1]Lucene, Solr or Elastic Search.

## Demo overview

Demo source with the configs and the queries.[1]

[1]https://github.com/ate47/RDF-MultiIndex-Demo

Example using the cocktail dataset to demonstrate the different aspect of the queries in different steps.

1. Show how the indexes and the filters are configured to create an architecture between the triple store and the SPARQL endpoint. This step will also show the overhead between a single-index based system and our multi-indexes based system.
2. Explain how to convert queries from a single-index based system to our newly created architecture. Its goal is to show the better performances of the joins and the complexity of the queries.
3. Showcase the hybrid search method with tasks within the two projects of the European Commission described bellow using the example of Kohesio.

## European Commission deployments

The system is currently deployed in the production of two European Commission projects.
Kohesio[1], a website tracing the projects and the beneficiaries of the EU funds.
The Europe Direct Contact Centre, answering the questions of about the European Union.

Both of these projects support the 24 official European languages. Using respectively 48 and 72 independent indexes. (2 or 3 types $\times$ 24 languages)

[1]https://kohesio.ec.europa.eu/

## Conclusion

- Provide system for **full-text indexing**.
- **Customisable** indexes with text analyser, language filtering or RDF types.
- Support **updates**.
- Built on top of the RDF4J API for an easy integration in current existing systems.

qEndpoint library[1]

[1]https://github.com/the-qa-company/qEndpoint