

Datasets and Task:

1. For this project, I utilized a dataset that consisted of relational data. I merged the datasets together to create a comprehensive database for the task at hand. I followed the instructions provided in the accompanying Jupyter Notebook and included comments within the code to facilitate understanding. Additionally, I included some ttl files for caching the graph and a requirements.txt file to assist in setting up the virtual environment for testing. The task involved constructing an RDF knowledge graph using the provided datasets. I successfully completed most of the sections, except for the last section which required dealing with different graph individuals and possible variations in answers.

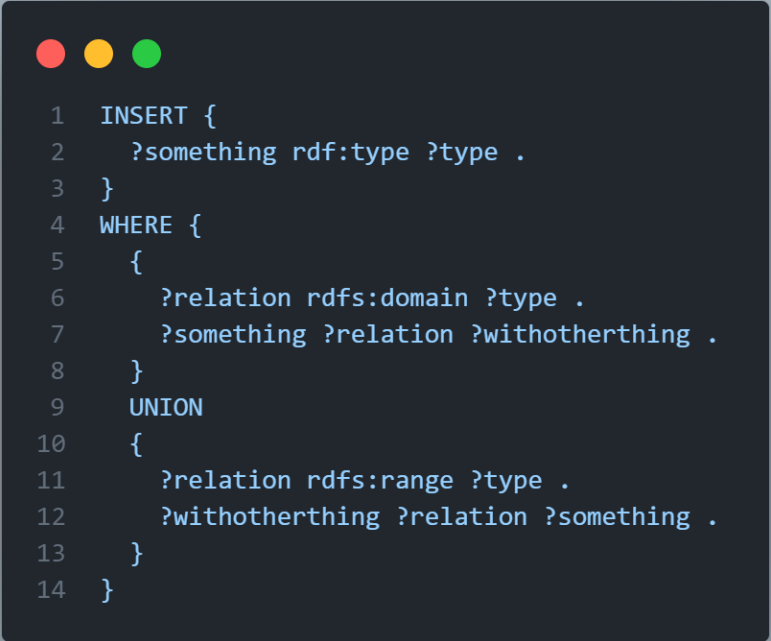
Workflow:

2. To begin, I started by defining the RDF graph without incorporating the RDFS language. As the project progressed, I added classes, domains, and ranges to enhance the entailing of facts within the graph. However, I encountered some challenges in connecting my local naming conventions with the standards used in DBpedia. Consequently, I had to verify my queries using the SPARQL DBpedia endpoint to ensure their accuracy. The Jupyter Notebook contains detailed technical information, including comments and additional files.

RDF Knowledge Graph:

3. The RDF knowledge graph comprises three main classes: person, band, and genre. The classes are represented as dbo:person, dbo:band, and ex:genre, respectively. Additionally, there are extra classes such as

dbo:MusicAlbum and ex:genres (a subclass). Several properties are defined within the graph, including dbp:artist (which establishes connections to both bands and individuals), dbo:hasrole (with a domain of person and a range of ex:role), dbp:birthCountry (a property linking artists to their birth countries), dbp:birthPlace, dbp:founding_country, and dbp:formerBandMember (which indicates the members of a band). Similar properties are also included. Inferences within the graph are made using the domain and range information, as well as subclass relationships. This enables the connection of domains and ranges to their respective classes. Example :



```
1  INSERT {
2    ?something rdf:type ?type .
3  }
4  WHERE {
5    {
6      ?relation rdfs:domain ?type .
7      ?something ?relation ?withotherthing .
8    }
9    UNION
10   {
11     ?relation rdfs:range ?type .
12     ?withotherthing ?relation ?something .
13   }
14 }
```

This is used to infer the domain and range

Conclusion:

4. While completing this project, I encountered certain limitations. One limitation is the reliance on the provided dataset, which may not cover all possible scenarios. Expanding the project would involve incorporating additional datasets to enhance the knowledge graph's coverage and accuracy. Furthermore, improving the alignment between local naming conventions and standard naming conventions would streamline the graph construction process. Additionally, exploring more advanced reasoning techniques and incorporating ontologies could enhance the graph's inferential capabilities and overall effectiveness.