Report on Week 5

Lei Liu, 9669373

Some exploration had been done by querying the ddbpedia endpoint with 30000ms execution time for each query.

1. Explore predicates that can be used to find country atoms.

Time spent: one hour on Monday, and two hours on Tuesday.

- First query:

```
SELECT distinct ?c
WHERE { ?c a dbo:Country }
```

This query returned 1694 URIs, including valid atoms about countries and invalid results. Both results were obtained according to one of its rdf:type: dbo:Country. The invalid atoms, such as Finland national cricket team and Indiana Democratic Party also have this predicate that depict its relation with the country they belong to.

- Second query:

By comparing this kind of invalid results with the correct country atom, a second predicate, capital, were employed.

```
SELECT distinct ?c
WHERE {
   ?c a dbo:Country;
dbo:capital ?cap.}
```

858 atoms were returned from this query. Indeed, some invalid atoms were removed successfully since we assume that country should have a capital while parties and teams do not have the relation with type dbo:capital. However, some countries that have not capital might also be filtered out. Besides, some organization or regions were included in this result.

- Third query

In order to remove some regions, like England, another predicate was added into the query.

```
SELECT distinct ?c
WHERE {
   ?c a dbo:Country;
dbo:capital ?capi;
dbo:leader ?l}
```

This query got 242 atoms, which is very close to the number of results queried in relational database-mondial. However, some atom of regions, like Scotland, Wales, still could not be removed from the querying country list, and as well as to some organizations.

- Fourth query

```
SELECT distinct ?c
WHERE {
   ?c a dbo:Country;
```

dbo:capital ?capi;

dbo:leader ?l;

dbp:cctld ?cct}

This query employed the predicate dbp:cctld that could be used to remove a lot of organizations and produce a more accurate result of country list. It returned 207 atoms, from which most of the atoms are valid information about countries.

These are the exploration of querying about countries so far. The endpoint in this period were relatively stable and reliable.

2. Explore information about City

Time spent: 3 hours on Wednesday.

- First query

select distinct ?c

where {

?c a dbo:City.}

This query took obviously long time for execution and returned part of atoms from the database. It returned 19381 atoms.

- Second query:

select distinct ?nm

where {

?c a dbo:City.

?c foaf:name ?nm.

}

This query returned the names of all cities, which had 21720 results. It also took a long time for execution, and the result were in different languages. Thus, a city would return several times according to the number of different language in the predicate foaf:name.

- Third query

After looking at the predicate foaf:name, it says that the name of a city is marked in multiple language, English, and native language. By filtering the language, it should return all value in English.

select distinct ?c ?nm

where {

?c a dbo:City.

?c foaf:name ?nm.

FILTER (langMatches(lang(?nm), "EN"))

}

Actually, the results were the same as the previous query and still cost a long time.

- Fourth query

select count distinct ?c ?nm

where {

?c a dbo:City.

?c dbp:name ?nm.

FILTER (langMatches(lang(?nm), "EN"))

}

By choosing another predicate, dbp:name, which contains only one string of name in English, all returned name of all cities were in English, but only 8440 records were returned. This means some cities might not have the predicate dbp:name.

- Five query

Another predicate is related to the name of a city.

select distinct ?c ?nm

where {

?c a dbo:City.

?c rdfs:label ?nm.

}

This query executed for a quite long time, and returned 133726 results. For each city, its label was displayed in multiple languages.

- Sixth query

select distinct ?c ?nm

where {

?c a dbo:City.

?c rdfs:label ?nm.

FILTER (langMatches(lang(?nm), "EN"))

}

By filtering language, the names of all city were shown in English, and each city has only one English label. The results were 19379 atoms, as similar as the number of cities in the first query (19381 atoms).

In this exploration, the execution time was significantly longer than the previous exploration, since much more atoms were queried and returned.

3. Exploration information with specific conditions.

Time spent: one hour on Thursday

- First query

select distinct ?c

where {

?c a dbo:City;

foaf:name "Manchester"@en.

}

This query returned the correct information as same as the relational database mondial within a relatively short executing time. Two atoms were returned.

- Second query

```
select distinct ?c
where {
?c a dbo:City;
 dbp:label "Manchester"@en.
}
```

It returned nothing. Although the city atoms have a property dbp:label, as the country atoms, this predicate does not relate to the name of a city.

- Third query

```
select distinct ?c
where {
?c a dbo:City;
 dbp:name "Manchester"@en.
}
```

Only one atom were returned from this query. Compared with the first query, another valid atoms did not return since it does not contain a dbp:name predicate.

- Fourth query

```
select distinct ?c ?nm
where {
?c a dbo:City;
 foaf:name ?nm.
filter( regex( ?nm, '^Man', 'i') )
}
```

Atoms of city with the name start by ‘Man’ were returned successfully and quickly. The results were similar with the relational database.

From this exploration, it is clear that query with the precise condition was much more accurate. However, different prefixes with the same predicate would generate different results.

In conclusion, the dbpedia endpoint is stable and reliable. However, for querying a range of data, this is not efficient and effective as a data structured with respect of a schema. A lot of time and human resource are needed to explore the relationship and predicates between objects and subjects. While, a simple SPARQL query could generate the much greater amount of data than querying in a traditional relational database which needs the combination of several tables and complex query. For example, querying all information of a city, SPARQL query would be one or two conditions, whereas an SQL query would have several conditions that used to join tables and make constraints.