Report For Assignment 3, Course: CMPUT 690
Nawshad Farruque

I have used Stanford Core NLP and Apache JENA API for answering the queries. However, alternatively to answer query 1 and 2 we can use simply SPARQL queries, which are as follows: (see q1_dbpedia.txt, q2_dbpedia.txt and q2.rq inside zipped project file under sparql_queries/ folder). To run this you need to download Apache JENA and issue following command:
bin/SPARQL --data= <dataFile.ttl> --query=<queryFile.rq> > output_file_name.tsv, Also all the result files are included in the zipped file under outputs folder with proper naming.

1) ***Which stadiums are used by which clubs?***

This can be obtained from a SPARQL query and running it against dbpedia, as following:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX db: <http://dbpedia.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?stadium_name ?club_name
WHERE {
  ?tenant_resource dbp:clubname ?club_name .
  ?tenant_resource rdf:type dbo:SoccerClub .
  ?stadium dbo:tenant ?tenant_resource .
  ?stadium dbp:stadiumName ?stadium_name .
  ?stadium rdf:type dbo:Stadium .
}
```

Although I wrote a program where I find out all the stadium description. Then from the text of description I generated textual triples. From those I filtered the triple with subject as stadium name, predicate such as home of/home ground/is owned by/is used and the following object which contains the team name. To further refine the objects which contains team name (with some noise), I saved all the team names in a list by running a SPARQL query on the RDF graph. Then I have checked if the object contains the basic part of the team name in the list, if so, I return the team name from that list.

2) ***Who plays for which team?***

This can be obtained from running the following query against the RDF graph provided. The query is as follows:

```
PREFIX fb: <http://rdf.freebase.com/ns/>
PREFIX fbk: <http://rdf.freebase.com/key/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX c690: <cmput690>

SELECT ?player_name ?player_id ?team_name ?team_id
WHERE
{
   ?player_id fbk:wikipedia.en ?player_name .
   ?team_id fbk:wikipedia.en ?team_name .
   ?resource fb:sports.sports_team_roster.team ?team_id .
   ?resource fb:sports.sports_team_roster.player ?player_id .
}
ORDER BY (?team_id)
```

or

The dbpedia query as following:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX db: <http://dbpedia.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?players_name ?team_name
WHERE {
  ?players_resource dbp:fullname ?players_name .
  ?resource dbp:name ?players_resource .
  ?resource dbp:clubname ?team_name .
  ?resource rdf:type dbo:SoccerClub .
}
```

3)***Who coaches a team of Spanish players?***

To get the answer of this, I first find out all the manager and the players currently playing or played under them(grouped by manager). I saved the manager ids and player ids in a Map, which contains manager ids as key and the players under them as the values(which is a list). Then for each players under each manager I fetched the description, find out the triples and checked if those has Spanish keyword in their object description. If so, then I saved the managers ID. Since, all the description contains the nationality of player at the very first line, it is sufficient to check the first line and triples generated from it having Spanish keyword in their object part. The query actually finds out the managers who had coached/coach at least one Spanish player. The output file contains only one column containing all the qualified manager ids.

4)***Which clubs have stadiums named after former presidents?***

Here I find out all the stadiums which has triples containing predicate or object: renamed after, named in honor of, renamed in honor of keywords and subject as stadium names. Again from those stadium triples, I find out the club names/team names for which they are the home ground(same logic as question 1) and output the stadium names tab separated by club/team names.

5) ***Which teams have the most nationalities amongst their roster?***

I find out all the teams and the players(currently playing or played). I find out each of the players Nationality from their corresponding description and analyzing the triples (same as question 4). I save the nationalities in a hashset so that for each team I can get the total count of its players unique nationalities just from the hashsets length. I am saving the team id and number of nationalities in the file as output. This can be sorted based on second column in tsv to see which teams have highest number of nationalities. This query takes a lot of time to output.

Findings:

I have used some heuristics which has very good implication. The consecutive personal pronouns (PRPs) in the description of any organizations can be replaced by its name. Also, we can replace "and the" or WH-Pronouns with the organizations name. A text blob about a particular organization discusses mainly about it, so from that intuition we can use this heuristic. Organizations name can be found out by querying on the RDF and that can be used for replacement, this way we will have more accurate textual triples or facts for particular organization from Stanford Open IE. In future I would try to use Stanford Coref Resolution system, I am not sure how to use the system to replace coreferences by the actual entities at this point.

For former chairman query, I was able to find out most of the stadiums and some of the club/team names. Its happening due to the inaccuracy of Open IE System and my algorithm for filtering triples with limited number of useful patterns.

Miscellaneous:

Since the Stanford Core NLP uses a pipeline for information extraction which consists of number of NLP task as parsing, stemming, pos tagging, ner, etc, it uses three NER classifiers, see output,

```
Loading classifier from edu/stanford/nlp/models/ner/english.all.3class.distsim.crf.ser.gz ... done [1.0 sec].
Loading classifier from edu/stanford/nlp/models/ner/english.muc.7class.distsim.crf.ser.gz ... done [1.9 sec].
Loading classifier from edu/stanford/nlp/models/ner/english.conll.4class.distsim.crf.ser.gz ... done [0.8 sec].
```

Although the classification was not perfect and it identifies nationalities as MISC . So, to make it streamlined, I used a text file that contains all the nationalities of the world(see nationalities.txt). Then I filtered out only the MISC elements from the output of Core NLP NER pipeline (see file allMISC.txt). I joined the two files based on the first column and saved only the common elements that means the nationalities used only in the description field of the RDF graph(see file nationalities_in_file.txt). Since common elements have duplicates, I sorted them and found out the uniq ones. In my program I loaded those nationalities in memory and used wherever it is needed. I used following unix command line tools and commands for this:

```
awk '$2=="MISC" {print $1}' allNERs.txt > allMISC.txt
awk 'NR==FNR{c[$1]++;next};c[$1] > 0' nationalities.txt allMISC.txt | sort | uniq > nationalities_in_file.txt
```