Summary

This project has been implemented to combine basic Natural Language Processing techniques with SPARQL query processing and answer questions over an RDF corpus using Java programming language. We have been given a corpus of RDF triples from Freebase and facts about some of entities that has been mentioned in documents. And here we have been asked five different questions to write queries with to extract information from this corpus. To carry out these, we have added Apache Jena open source framework to our project library that enables us to perform SPARQL queries over the RDF triples. We call our Main method and as soon as execute this, we get results for each of five requirements. Below are our approaches to answer the requirements:

1. Which stadiums are used by which clubs?

Here, we have used query that selects "stadium" pattern from the documents, then matched obtained stadiums with teams that contains "home stadium of/to", "home ground of/to" patterns. As a result, we get *cmput690a3_q1_Atakishiyev.tsv* output file.

2. Who plays for which team?

We select players and teams from *fb:sports.sports_team_roster.player* and *fb:sports.sports_team_roster.team* that connects to each other with the same blank node and then match these with appropriate Wikipedia IDs and get appropriate player/team pairs. The result has been written to *cmput690a3_q2_Atakishiyev.tsv* file.

3. Who coaches a team with Spanish players?

For this query, we again select players and teams from *fb:sports.sports_team_roster.player* and *fb:sports.sports_team_roster.team* connected to each other with the same blank nodes , manager from *fb:soccer.football_team_management_tenure.manager* and match the last one with *fb:soccer.football_team_management_tenure.team* that connects each other with the same blank nodes. Finally, with these found IDs we have searched for documents that contains *"Spanish footballer"* pattern and grouped obtained results by manager ID. The result is in *cmput690a3_q3_Atakishiyev.tsv* file with manager IDs, teams and the Spanish players of those teams.

4. Which clubs have stadiums named after former presidents?

We have used a query that selects "stadium" pattern from the documents, then matched obtained stadiums with teams that contains "home stadium of/to", "home ground of/to" patterns. After that, we looked through the documents that filtered results with "former president/chairman" pattern and got those club/stadium pairs. The result has been written in *cmput690a3_q4_Atakishiyev.tsv* file.

5. Which teams have the most nationalities amongst their roster?

To answer this question, we search for players and teams from *fb:sports.sports_team_roster.player* and *fb:sports.sports_team_roster.player* then search for "is a/an w{3,20} footballer" pattern from text documents.Here w{3,20} indicates the nationality. Then for each team we count nationalities and assign the number of these nationalities for each of teams. Finally, we have filtered results by showing 120 results and written the output to in *cmput690a3_q5_Atakishiyev.tsv* file.

While finding solutions to each of requirements we encountered some issues regarding approaches to the problems. For the question one, my main problem became to figure out query that extracts list of stadiums using their Wikipedia ID and matches them with team names because there were not blank nodes that connect these pairs. Therefore, I selected “stadium” pattern from the documents, then matched obtained stadiums with teams that contains “home stadium of/to”, “home ground of/to” patterns. In the second question, finding query was a bit easier because there were blank nodes that connected these two classes. Consequently, we were able to perform query that extract team-player pairs easily. The third requirement was also one of cumbersome task that I dealt with in terms of coming up with appropriate query because there weren’t direct connections between these triples that we could write simple query. I firstly decided to select players and teams from appropriate *fb:sports.sports_team_roster.player* and *fb:sports.sports_team_roster.team* and then matched teams with managers using *fb:soccer.football_team_management_tenure.team* identifier and *fb:soccer.football_team_management_tenure.manager* as they have the same blank nodes. The fourth question was almost similar to first requirement and dealing with finding query did not become very difficult, because I had already solved the first question that gives me the list of stadiums. The main difficulty was in text documents, so that finding “named after former president” pattern similarities varied, therefore I looked through text and designated “named after”, “former chairman”, ”former president” patterns and eventually obtained five such kinds of stadiums. Finally, the fifth question was also one of hard task that I coped with. Initially, I did not know how to find different kinds of nationalities from the documents. Then looking carefully, I saw that great majority of documents that describe footballers have similar contexts: Those documents mainly start with this template: ”(Somebody) is a/an (Nationality) footballer who plays for…”. Thus, I took all of different teams, counted all nationalities that belongs to these teams programmatically and assigned the number of different nationality counts for each of teams. These gave me the teams and appropriate nationality number that those teams have.

After completing answering to all of the questions, I witnessed that majority of accuracies for each requirement was fairly good. However, there were either missing results or repetitions of obtained results that caused overall performance of programs to degrade. For the first question, we obtained 44 stadium/team pairs,however our program failed to display 13 teams out of this number. I reanalyzed the RDF graph and saw that I have not used fb:type.object.key IDs in my query and these missing teams can be retrieved by assigning their fb:type.object.key ID to document ID and get their names from those documents. So, this method would select the missing teams.For the second question, I got very long list of players and their teams, and saw that player and team names get repeated several times.In order to prevent this case, we can use player and team pattern so that, if the patterns contain the same or very similar content, filter and display most frequently appearing format. This method is very efficient to avoid repetitions. Regarding the third question, we extract managers that they coach teams with the Spanish players. Majority of results are correct for this question, however better accuracy can be obtained if we detect the same managers from text and see if there are more than two dates that their id contains, and we could extract the latest date so that in fact other dates belonging to these managers remained in past and for now these managers do not coach those teams with those old dates. We

should only take the latest date belonging to that manager and write only this case as a fact for current time. For the fourth question, we extracted teams and stadiums that were named after former chairmen. In the document, the number of these cases were so few, we only obtained 5 such states programmatically, however looking meticulously, we saw that “Vicente Calderon” which belongs to Atletico Madrid and “Santiago Bernabeu” belonging to Real Madrid are also named stadiums after former presidents. The reason they could not be detected because sentence structure was a bit different and complicated from our standard approach. But probably to extract them, we could use special methods for each: taking pattern “changed to the? after the famous club president” for Atletico Madrid and “renamed in honor of their former chairman ” pattern for Real Madrid and adding these to our code as special conditions. Finally for the fifth question, tendency is just like as in the second question: there are repetitions. We could again take most appearing form of football teams as best indication and assign the appropriate nationality count for them. These would shorten our redundant list and illustrate one team/ one count outcome.

In conclusion, we described our algorithm to answer the required questions, the main difficulties we encountered while designating our approach, and new methodologies that can improve the output performance. Our results for requirements were generally correct, but can be improved with emphasized methods shown above.