SPARQL Queries documentation

1 – What is the role of the player who scored the most goals?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

#PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
select ?Player ?Role  ?playerLabel ?roleLabel where {
      ?Player se:hasRole ?Role ;
                  se:goals     ?MostGoals .
    filter(?MostGoals = ?MaxGoals) .
    ?Player rdfs:label ?playerLabel .
    ?Role rdfs:label ?roleLabel .
    {
select * where {
    ?BestPlayer se:hasRole ?Role .
    ?BestPlayer se:goals  ?MostGoals .
    filter(xsd:int(?MostGoals) = ?MaxGoals)
    {
    select (max(xsd:int(?Goals)) as ?MaxGoals) where
        {
            ?Player se:goals ?Goals .
        }
    }
}
    }
}
limit 1
```

In this query we try to first get the Player and the corresponding goals scored by that player. We do this using the hasRole goals properties that helps us get the Goals and the Class Role for our dbo:SoccerPlayer. We then filter for the players with the max goals using the filter function in sparql. We then use those dbo:SoccerPlayers to get the their Class Role and Labels that is then returned.

2 – What were the points of the opposition team of the team that faced Russia in the first round ?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX dbo:<http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
```

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

select ?pointsTeamFaceRussia where {
    ?Team4 se:points ?pointsTeamFaceRussia .
    {
        select  Distinct (?Team3 as ?Team4)  where {
            ?Team3 ?p ?o .
            {
                select ?Team3 where {
                    ?Team2 se:isOpponentOf ?Team3 .
                    ?Match2 se:hasTeam ?Team2 .
                    ?Match2 se:round "1"^^rdfs:Literal .
                    {
                        select * where {
                            ?Team1 se:isOpponentOf ?Team2 .
                            ?Match se:hasHomeTeam ?Team1 .
                            ?Team1 rdfs:label "Russia" .
                            ?Match se:round  "1"^^rdfs:Literal .
                        }
                    }
                }
            }
      }
      }
}
```

In this query we first try to get the Class Team with label Russia and then use the symmetric property isOpponentof to get the opponent of Russia.
`?Team1 se:isOpponentOf ?Team2 .`
`?Match se:hasHomeTeam ?Team1 .`
`?Team1 rdfs:label "Russia" .`
We filter it to get the opponents only i.e. se:Team in the first round. With these se:Team results we use the symmetric property again to get the opponents of these teams that faced Russia in the first round and then fetch the points for these opponents after running them through a distinct filter.
`?Team2 se:isOpponentOf ?Team3 .`
`?Match2 se:hasTeam ?Team2 .`

3 – Which teams have the highest score in the first 3 group stages of the world cup ?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

select  ?won (COUNT(?won) as ?numberOfTime) ?wonLabel  where {
```

```
    ?match se:round ?round .
    ?match se:hasAwayTeam ?awayTeam .
    ?match se:hasHomeTeam ?homeTeam .
    ?match se:homeTeamScore ?homeScore .
    ?match se:awayTeamScore ?awayScore .
     BIND(
        IF(?homeScore > ?awayScore, ?homeTeam,
            IF(?homeScore < ?awayScore, ?awayTeam,"Draw")) AS ?won) .

       ?won rdfs:label ?wonLabel .

    filter(?won != "Draw" && ( ?round = "1"^^rdfs:Literal || ?round =
"2"^^rdfs:Literal  || ?round ="3"^^rdfs:Literal) && lang(?wonLabel) = "en") .
}
group by ?won ?wonLabel
order by desc(?numberOfTime) limit 3
```

In this query we first use the Bind command to create a new Column won using the Score of the Home Team and the Away Team which are both instances of the Class Team.

```
BIND(
    IF(?homeScore > ?awayScore, ?homeTeam,
       IF(?homeScore < ?awayScore, ?awayTeam,"Draw")) AS ?won) .
```

We get the scores by first finding the home team and away team using the properties hasAwayTeam and hasHomeTeam and then using the homeTeamScore and awayTeamScore we get the score which we use in our Bind command. We then use the Filter Function to keep the results only the match that took place in round 1, 2 and 3 and that wasn't a draw. We use a group by to check the team who has the most wins and we give back that team: `group by ?won ?wonLabel`.

4 – Who are the goalkeepers of the Semi-Finals teams?

```
PREFIX se:<http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xml:<http://www.w3.org/XML/1998/namespace>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

select ?players ?playersLabel where {
    ?matches rdf:type dbo:FootballMatch ;
          se:round "Semi Finals"^^rdfs:Literal ;
          se:hasTeam ?teams .

    ?players rdf:type dbo:SoccerPlayer ;
             se:playsIn ?teams;
             rdfs:label ?playersLabel;
             se:hasRole
<http://www.semanticweb.org/kde/ontologies/sport-events/Goalkeeper> .
}
```

To retrieve the goalkeepers of teams playing in semi-final matches, we first have to select all teams which played semi-final matches. We can then fetch the players of these teams and select only the ones having the Goalkeeper role.

5 – How many different clubs are the Spain team players in?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT (COUNT(DISTINCT(?club)) AS ?count)
WHERE {
    ?team rdf:type se:Team ;
    rdfs:label "Spain" .
    ?player se:playsIn ?team .
    ?player se:playsIn ?club .
    ?club rdf:type dbo:SoccerClub .
}
```

To answer this question, we can start by getting the Spain team by its label. It is then possible to fetch all players from this team and bind them to their respective clubs with the *se:playsIn* property. Finally, the number of clubs can be retrieved using a *COUNT(DISTINCT(?club))* aggregate clause.

6 – Which team has the lowest total number of goals in the season?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?team ?teamLabel ?goals  WHERE {

    ?team rdfs:label ?teamLabel .
    filter(lang(?teamLabel) = "en") .

    {
        SELECT ?team (SUM(xsd:nonNegativeInteger(?goal)) as ?goals) WHERE {
            ?player se:hasNationality ?team .
            ?player se:goals ?goal .
        }

        GROUP BY ?team
        ORDER BY ASC((SUM(xsd:nonNegativeInteger(?goal))))
    }
}
LIMIT 1
```

The idea of this query is to sum the total number of goals of all players and group the results by team. This can be done using the following clauses: *GROUP BY ?team ORDER BY ASC((SUM(xsd:nonNegativeInteger(?goal))))*. Thanks to the ordering, taking the first team with *LIMIT 1* allows us to know which team has the lowest total number of goals.

7 – What are the nationalities of the top 10 players with the most number of assists in their clubs?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

select ?nationality ?nationalityLabel where {
    ?player se:hasNationality ?nationality .
    ?nationality rdfs:label ?nationalityLabel .
    filter(lang(?nationalityLabel) = "en") .

    {
        select * where {
            ?player se:assists ?numberAssists
        }
             ORDER BY DESC(?numberAssists) LIMIT 10
      }
```

```
}
```

This question can be answered by retrieving the number of assists of each player (i.e. *?player se:assists ?number*) and limiting results to those of the top 10 with *ORDER BY DESC(?numberAssists) LIMIT 10*. It is then straightforward to fetch the nationality of the retrieved players with *se:hasNationality* property.

8 – Which players played as the home team on 23/06/2018 during the Football World Cup?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?player ?playerLabel
WHERE {
    ?match se:scheduledAt ?date.
    FILTER (xsd:dateTime(?date) >= "2018-06-23T00:00:00"^^xsd:dateTime &&
xsd:dateTime(?date) < "2018-06-24T00:00:00"^^xsd:dateTime).
    ?match se:hasHomeTeam ?HomeTeam.
    ?player se:playsIn ?HomeTeam.
    ?player rdfs:label ?playerLabel.
}
```

To answer this question, we can get all players which played as home team from any match using *?match se:hasHomeTeam ?HomeTeam.* and then *?player se:playsIn ?HomeTeam.* triples. It is then possible to retrieve the date of each match with *?match se:scheduledAt ?date.* and finally filtering out the results to keep only matches that happened the 23th of June 2018: *FILTER (xsd:dateTime(?date) >= "2018-06-23T00:00:00"^^xsd:dateTime && xsd:dateTime(?date) < "2018-06-24T00:00:00"^^xsd:dateTime).*

9 – Which players are composing the teams engaged in semi-final matches?

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

select * where {
    ?matches rdf:type dbo:FootballMatch ;
          se:round "Semi Finals"^^rdfs:Literal ;
          se:hasTeam ?teams .
```

```
    ?players rdf:type dbo:SoccerPlayer ;
             se:playsIn ?teams ;
             rdfs:label ?playersLabel .
}
```

The first part of this query is to retrieve all matches having “Semi Finals” as round property with *?matches se:round "Semi Finals"^^rdfs:Literal.* After retrieving the teams that played in these matches with *se:hasTeam* we just have to get all players that plays in these teams: *?players se:playsIn ?teams .*

10 – Which players won a match between 16/06/2018 and 22/06/2018 during the Football World Cup 2018?

```
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?player ?playerLabel
WHERE {
    ?player se:playsIn ?team .
    ?team rdf:type se:Team .
    ?player rdfs:label ?playerLabel
    {
        ?match se:awayTeamScore ?away_team_score.
        ?match se:homeTeamScore ?home_team_score.
        ?match se:hasAwayTeam ?away_team.
        ?match se:hasHomeTeam ?home_team.
        ?match se:scheduledAt ?time.
        BIND(
                IF(?away_team_score > ?home_team_score, ?away_team,
                IF(?away_team_score < ?home_team_score, ?away_team, "No one"))
            AS ?team
        ).
        FILTER(xsd:dateTime(?time) > "2018-06-16T00:00:00"^^xsd:dateTime &&
xsd:dateTime(?time) < "2018-06-22T00:00:00"^^xsd:dateTime).
    }
}
```

This question can be answered by comparing the awayTimeScore and homeTeamScore of each match happened between 16/06/2018 and 22/06/2018. We use *BIND(*
IF(?away_team_score > ?home_team_score, ?away_team,
IF(?away_team_score < ?home_team_score, ?away_team, "No one"))
AS ?team
) to combine two results of the if condition filters.
And then list the winners of every match.

11 – In which country did the first match occur? (transitive)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX dbo: <http://dbpedia.org/ontology/>

select ?firstMatch ?city ?country where {
    ?firstMatch rdf:type dbo:FootballMatch ;
                se:scheduledAt ?scheduledAt ;
                se:tookPlaceIn ?stadium .

    ?stadium se:isIn ?city .
    ?city se:isIn ?country .
} order by ?scheduledAt limit 1
```

We can take advantage of the transitive property *se:isIn* to answer this question. First of all, to retrieve only the first match, we can use *?firstMatch rdf:type dbo:FootballMatch ; se:scheduledAt ?scheduledAt .* and order by that latter *?scheduledAt* variable and limiting the results to only one. Then, it is possible to get the Stadium that match took place in with *?firstMatch se:tookPlaceIn ?stadium.* From that point, we can fetch the country by transitivity: *?stadium se:isIn ?city . ?city se:isIn ?country .*

12 – What is the population of each opponent's country of the last match? (using dbpedia data)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX se: <http://www.semanticweb.org/kde/ontologies/sport-events#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

select ?lastMatch ?teams ?country ?population where {
    ?lastMatch se:hasTeam ?teams .
    ?teams rdfs:label ?teamsLabel .

    filter(lang(?teamsLabel) = "en") .

    service <http://dbpedia.org/sparql> {
            ?country rdf:type dbo:Country ;
            rdfs:label ?teamsLabel ;
            dbo:populationTotal ?population .

    }

    {
        select ?lastMatch where {
            service <http://192.168.1.47:7200/repositories/task-7> {
                ?lastMatch rdf:type dbo:FootballMatch ;
                           se:scheduledAt ?scheduledAt .
```

```
            }
        } order by desc(?scheduledAt) limit 1
    }
}
```

Retrieving the last match can be done by querying all matches (*?lastMatch rdf:type dbo:FootballMatch ; se:scheduledAt ?scheduledAt* .), ordering the results by descending date and limiting the results to a single one.
The particularity of that SPARQL query is that it takes advantage of the *service* clause to query data from both our local triplestore and dbpedia SPARQL endpoint (http://dbpedia.org/sparql):

service <http://dbpedia.org/sparql> {
?country rdf:type dbo:Country ;
rdfs:label ?teamsLabel ;
dbo:populationTotal ?population .
}

As teams are national teams in our ontology, we can bind the label of a team to a *dbo:Country* and fetch its population with *dbo:populationTotal* property.