SPARQL

Instructions:

Write the answers in this same file. Specifically, place your answer to each query just below the query statement. Then, upload the file to Moodle on time (check the time and do not miss the deadline!).

Statement:

Solve the following queries with SPARQL. You are suggested to validate your answers with the SPARQL Explorer (http://dbpedia.org/snorql/). If you create any PREFIX include it in your answer. You can also check the DBPedia ontology schema at: http://mappings.dbpedia.org/server/ontology/classes/

1. *Write the SPARQL query to list all the DBPedia classes (i.e., instances of the owl:Class).*

```
SELECT ?x
WHERE {
  ?x rdf:type owl:Class .
}
```

2. *Write the SPARQL query to list all the DBPedia properties (i.e., either instances of the owl:DatatypeProperty or owl:ObjectProperty properties).*

```
SELECT ?x
WHERE {
  {?x rdf:type owl:DatatypeProperty}
  UNION
  {?x rdf:type owl:ObjectProperty}
}
```

3. *Write the SPARQL query to list all the triples in DBPedia using the populationEstimate (i.e., http://dbpedia.org/property/populationEstimate) property:*

```
SELECT ?x ?y
WHERE {
  ?x dbpedia2:populationEstimate ?y
}
```

4. *Write the SPARQL query (i.e., a single query) to list all the pairs <SUBJECT, PROPERTY> where <http://dbpedia.org/ontology/CyclingTeam> is the OBJECT **and** all the pairs <PROPERTY, OBJECT> where <http://dbpedia.org/ontology/CyclingTeam> is the SUBJECT.*

Not sure if you asked for all pairs that fulfill both properties or all pairs that fulfill the first one and all pairs that fulfill the second one (I included both):

```
PREFIX on: <http://dbpedia.org/ontology/>
SELECT ?s ?p ?o
WHERE {
  ?s ?p on:CyclingTeam .
  on:CyclingTeam ?p ?o .
}

or

PREFIX on: <http://dbpedia.org/ontology/>
SELECT ?s ?p ?o
WHERE {
  { ?s ?p on:CyclingTeam}
  UNION
  {on:CyclingTeam ?p ?o}
}
```

5. *Now check the results you obtained from the previous query. Briefly explain the SPARQL query results. What do these pairs mean (you can group the results per similarity and comment on each group)? You can ignore the following properties (for internal usage of DBPedia): wasDerivedFrom and isDefinedBy as well as describedby, defines and describes (mainly used for metadata purposes).*

:Lupus_Racing_Team	rdf:type
:Lviv_Cycling_Team	rdf:type
:Lviv_Cycling_Team_(women's_team)	rdf:type
:Lygie_(cycling_team)	rdf:type
:MEXX-Watersley_International_Women's_Cycling_Team	rdf:type
:MG_Maglificio_(cycling_team)	rdf:type
:Macogep_Tornatech_Girondins_de_Bordeaux	rdf:type
:Madeinox-BRIC-AR_Canelas	rdf:type
:Madison_Genesis	rdf:type
:Madison_Saracen	rdf:type
:Maes_Pils_(cycling_team)	rdf:type

The subjects correspond to **cycling teams' entities** which class is *CyclingTeam*.

6. *Now, let us explore the data. Write the SPARQL query to list the name (i.e., rdfs:label) of all persons (i.e., of type <http://dbpedia.org/ontology/Person>) that were born (i.e., <http://dbpedia.org/property/birthPlace>) in Barcelona. Assume that the range of the birthPlace property is a literal. Also, order the result in ascending order.*

```
PREFIX on: <http://dbpedia.org/ontology/>
PREFIX re: <http://dbpedia.org/resource/>
SELECT ?name
WHERE {
  ?x a on:Person .
  ?x rdfs:label ?name .
  ?x on:birthPlace re:Barcelona .
}
ORDER BY ASC(?name)
```