Semantic Web
Lab Assignment 3

1. Create and test queries using Gruff.

   Use data file "VC-DB-3.rdf". You will need to make a new triple store or overwrite an existing one—**you will not be submitting the store**.

   Create a SPARQL query for each description below. Use **prefixes** for name spaces. Test your queries within Gruff.

Query ID	Description
1	A query that gives the subject IRI where the formatted name is Jane Smith.
2	A query that lists all the formatted names in the data.
3	A query that lists the IRIs, formatted names, and ages of all people for which we have both formatted name and age.
4	A query that lists a person's IRI, formatted name, and age of people whose age is 25 or more.
5	A query to list the IRI and given names of all people for which we have both formatted name and age. (Hint: blank node, reuse #3)

   Store **all queries and results** in one text file named:

   Lab3_1_<YourID>.txt

   where <YourID> should be replaced with your UTD NetID. Insert an empty line between each query. Add a comment line before each query using the form:

   #Query <num>

   where <num> matches the query identifier in the table, above.

See the end of exercise 2 for instructions on submission to the eLearning site.

2. Investigate a data store for Incident information—load a file, query the data, and output results.

   a. Use Jena to load "Monterey.rdf" into an in-memory default (i.e., un-named) Jena triple store. Use code to print to the screen how long the insertion/loading step requires in tenths of seconds (e.g., "Load of Monterey.rdf took 2.5 seconds").
   b. Use the Query and ResultSet classes of Jena to add **a single query** to your class that gives **<u>all information</u> on *monterey incident 1621*** and write the query results to the file:

      ./Lab3_2_<YourID>.xml

      in xml format, where <YourID> should be replaced with your UTD NetID.

HINTS:

- Use Gruff to load and query the information on the incident. Continue modifying the query until all the incident data is exposed. An example top level SPARQL query is:
```
SELECT ?p1 ?o1
WHERE { <http://urn.monterey.org/incidents#incident1621> ?p1 ?o1 }
```
  Notice that the subject is specified, so it is not shown in the results. Extend the query to expose the entire incident and related information. It should include any object resources (`?o1`) found at the top level used as subject resources in the extended query.
- Investigate what needs to be extended by performing the top level query, examining the results and adding any second level query statements. Use a logical variable naming process such as `?p1` and `?o1` for the top level, `?p2` and `?o2` for the second level, etc. You may need to use UNIONs and/or other SPARQL keywords to properly collect your data.
- Watch out for blank nodes!
- Generally, an "organization" is associated with an "incident". Organizations back reference all the incidences in which they are associated. Filter out all other incident information found at the organization level—we want the organization data, but not all of its associated incidents. Otherwise...circular references are bad.
- Once your query is complete in Gruff, use it in the Jena code.
- An example of using Query and ResultSet in Jena can be found at:
  http://www.ibm.com/developerworks/xml/library/j-sparql/
  in the section named "Executing SPARQL queries with the Jena API"
- ResultSetFormatter. outputAsXML() provides an option to output information as an XML String which should provide the desired form. You will need to create an OutputStream for your output file or some other equivalent output process.
- Outputting the ResultSet destroys the ResultSet for some some odd reason (or it did previously). If you write to STDOUT before writing to a file, it may destroy the data before the second write. So, just do STDOUT prints until you have your full output, then change your code to output to your file.
- If the data is too big for your memory, you may need to use a TDB. Do so, if needed.

EXTRA CREDIT:

Push the results to a new in-memory model and output the model into two files (RDF/XML and Turtle). Check this output in Gruff (Gruff reads RDF, not Turtle). Since the results from the query are in table form, you will need to process the results into triples for the new model. Write the query results to the files:

./Lab3_2EC_<YourID>.rdf
./Lab3_2EC_<YourID>.ttl

in RDF and Turtle formats, where <YourID> should be replaced with your UTD NetID.

Prepare to submit your eclipse project: If your eclipse project for Lab 3 is quite large (>30MB) you will not be able to submit it to eLearning. Please add the following steps to preparing your submittal:

1. If present, delete contents of the MyDatabases/ directory (problem is likely MyDatabases/Dataset1 or wherever you stored the TDB if you used one)
2. Confirm that your project still works
3. (Again, if present) Delete contents of the MyDatabases/ directory
4. Make your zip file

   (The important thing is that your project works for the grader AND your zip file is a reasonable size [does not contain any triple store, MyDatabases/ directory]).

Please compress your eclipse project as well as the .txt file from exercise #1 into one zip file, and submit the zip file on e-learning. Name your main java file lab3_2.java.

Grading (100 points):

-100	Nothing submitted
-50	No Part 1 answer file
-5	Particular query doesn't work in Gruff
-50	Jena program not submitted
-10	No Summary Report for part 2
-10	Program Doesn't Run
-1/item	Program has wrong output, missing data
-3	Has Log4j runtime complaint
-5	Incorrect output filenames
-5	Incorrect source filenames
-10	Lacking use of Jena memory model (in-memory or TDB as needed)

EXTRA CREDIT: +10