Validating RDF data

1 From the lecture

a) Why do we need a validation language for RDF?

b) Can you mention some approaches proposed for validation of RDF?

c) How is SHACL different from OWL?

d) What two main types of shapes used in SHACL and what do they describe?

2 Exercises: OWL and constraints

Consider this OWL statement $\mathsf{Student} \sqsubseteq \exists \mathsf{enrolledIn}.\mathsf{Course}$. It seems to express the same thing as this SHACL constraint:

```
:StudentShape a sh:NodeShape ;
  sh:targetClass :Student ;
  sh:property [
    sh:path :enrolledIn ;
    sh:minCount 1 ;
    sh:class :Course
              ] .
```

They do, however, express two quite different things.

2.1 Exercise

Give an interpretation $\mathcal{I}_1$ and a set of triples $\mathcal{A}_1$ such that:

1. $\mathcal{I}_1 \vDash \mathsf{Student} \sqsubseteq \exists \mathsf{enrolledIn}.\mathsf{Course}$
2. $\mathcal{I}_1 \vDash \mathcal{A}_1$
3. $\mathcal{A}_1$ does not satisfy the SHACL constraint.

2.2 Exercises

Give an interpretation $\mathcal{I}_2$ and a set of triples $\mathcal{A}_2$ such that:

1. $\mathcal{I}_1 \not\models \mathsf{Student} \sqsubseteq \exists \mathsf{enrolledIn}.\mathsf{Course}$
2. $\mathcal{I}_1 \models \mathcal{A}_2$
3. $\mathcal{A}_2$ satisifies the SHACL constraint.

3 SHACL constraints for the Simpsons family

Write the SHACL constaints in a turtle file. You can check the simpsons.ttl-file from oblig1 against these constraints using, for instance Shacl playground.

3.1 Exercises: Family shape

1. Create a shape `FamilyShape` that ensures that all instances of `fam:Family` have at least 2 members and the members are of type `foaf:Person`.
2. Run the test and check that the data does not violate the restriction.
3. Add a new instance to the family that is not of type `foaf:Person` and check that you get a violation (remove it afterwords)

3.2 Exercises: name

1. Create a shape, `PersonShape` that ensures that all `foaf:Person`s have exactly one `foaf:name` and that it is of type `xsd:string`.
2. Run the test. What do you find?
3. Add the missing names:
   - Mona Simpson
   - Herbert Powell (Herb)
   - Abraham Simpson (Abraham)
   - Patricia Maleficent (Patty)
   - Selma Bouvier (Selma)
4. What do you find now?
5. Remove the blank-nodes with missing names from the graph and check that there are no violations.

3.3 Exercises: age

1. Extend the shape, `PersonShape` with add a property that checks that all `foaf:Persons` have exactly one `foaf:age` that is of type `xsd:int` and is a value between 0 and 120.
2. Run the test. What do you find?
3. Add missing age-values:
   - Abraham Simpson: 83
   - Mona Simpson: 66
   - Herb: 39
   - Patty: 41
   - Selma: 41
4. Test again an check that the violations are gone.

3.4 Exercises: different father and mother

In SHACL, create a property constraint, `DifferentFatherAndMother` checking that a person cannot have the same person as mother and father. Extend the `:PersonShape` with `DifferentFatherAndMother` and check if the simpsons-file violates this restriction.