Introduction à SHACL

Konstantinos Lambrou-Latreille

Mise à jour: 3 octobre 2020

C’est quoi SHACL ?

- **SHA**pes **C**onstraint **L**anguage du W3C
- Définition du schéma d’un graphe RDF
- Validation de graphes RDF à partir de graphes SHACL

Pourquoi pas RDF(S) et OWL ?

- Non adapté à la validation de graphe RDF
  - Axé sur l’inférence que sur la validation
- Hypothèse du monde ouvert
- Ne permettent pas la définition d’un schéma similaire à une base de données relationnelles

Et SHACL?

- Conçu principalement pour valider le schéma d’un graphe RDF
- Hypothèse du monde fermé

```
# SHACL
ex:PersonShape
      a sh:NodeShape ;
      sh:targetClass ex:Person ;
      sh:path ex:parent ;
      sh:minCount 1 .
```

vs

```
# OWL
ex:Person owl:equivalentClass
      [ rdf:type owl:Restriction ;
        owl:onProperty ex:parent ;
        owl:minCardinality “1”
      ] .
```

Rapport de validation SHACL

- La spécification de SHACL inclut le format attendu du rapport de validation

Rapport de validation SHACL

```
# Graphe SHACL
@prefix ex: <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

ex:PersonShape a sh:NodeShape ;
  sh:targetClass ex:Person ;
  sh:path ex:parent ;
  sh:minCount 1 .
```

```
# Graphe RDF
@prefix ex: <http://example.com/> .

ex:michel a ex:Person .
ex:denis ex:parent ex:clement .
ex:clement a ex:Person ;
           ex:parent ex:jopseh .
```

```
# Rapport de validation SHACL
[
      a sh:ValidationResult ;
      sh:resultSeverity sh:Violation ;
      sh:sourceConstraintComponent sh:MinCountConstraintComponent ;
      sh:sourceShape ex:PersonShape ;
      sh:focusNode ex:michel ;
      sh:resultPath ex:parent ;
      sh:resultMessage "Less than 1 values" ;
] .
```

Les « Shapes »

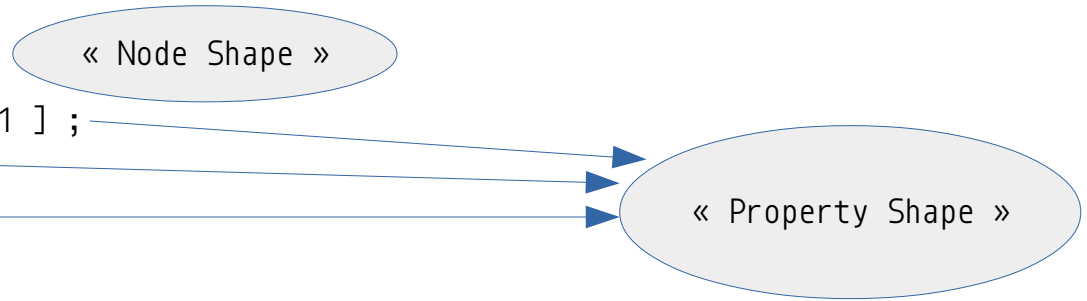
- Les « Node shapes »
  - déclarent des contraintes sur des noeuds visés
  - regroupent des « Property shapes »
- Les « Property shapes » spécifient des contraintes sur des chemins

Les « Shapes »

```
# Graphe SHACL
@prefix ex: <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

ex:PersonShape a sh:NodeShape ;
  sh:targetClass ex:Person ;
  sh:property [ sh:path ex:firstname; sh:minCount 1 ] ;
  sh:property ex:PersonLastnameShape .

ex:PersonLastnameShape a sh:PropertyShape ;
  sh:path ex:lastname ;
  sh:minCount 1 .
```



Ciblé des noeuds avec « sh:target* »

Cible	Description
sh:targetClass ?c	Tous les noeuds qui sont une instance (rdf:type) de ?c
sh:targetNode ?n	Tous les noeuds ?n
sh:targetSubjectsOf ?p	Tous les noeuds qui sont le sujet d’une propriété ?p
sh:targetObjectsOf ?p	Tous les noeuds qui sont l’object d’une propriété ?p

« sh:path »

- Un « Property shape » contient obligatoirement un « sh:path »
- L’objet d’un « sh:path » est un « property path »

```
sh:path ex:email # :alice

sh:path (ex:knows ex:email)

sh:path [ sh:inversePath ex:son ]

# Chemins de propriétés suivants peu supportés:
# zeroOrMorePath, alternativePath, oneOrMorePath, ...
```

Exercice

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

:PersonShape a sh:NodeShape ;
  sh:targetNode :alice, :bob ;
  sh:property [
    sh:path (:knows :name) ;
    sh:minCount 1
  ] .
```

```
@prefix : <http://example.com/> .

:alice
  :knows :bob ;
  :name "Alice" .

:bob
  :knows :alice .
```

Quel(s) noeuds produis(ent) une erreur ?

A. :alice
B. :bob
C. :alice et :bob
D. Aucun

Exercice

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

:PersonShape a sh:NodeShape ;
  sh:targetClass :Person ;
  sh:property [
    sh:path :name ;
    sh:minCount 1
  ] .
```

```
@prefix : <http://example.com/> .

:alice
  :knows :bob ;
  :name "Alice" .

:bob
  :knows :alice .
```

Quel(s) noeuds produis(ent) une erreur ?

A. :alice
B. :bob
C. :alice et :bob
D. Aucun

« targetClass » implicite

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :name ;
    sh:minCount 1
  ] .
```

```
@prefix : <http://example.com/> .

:alice
  :knows :bob .

:bob a :Person ;
  :knows :alice ;
  :name "Bob" .
```

Quel(s) noeuds produis(ent) une erreur ?

A. :alice
B. :bob
C. :alice et :bob
D. Aucun

Contraintes principales

Type de contraintes	Contraintes
Cardinalité	`sh:minCount, sh:maxCount`
Types de valeurs	`sh:class, sh:datatype, sh:nodeKind`
Valeurs	`sh:node, sh:in, sh:hasValue`
Interval de valeurs	`sh:minInclusive, sh:maxInclusive sh:minExclusive, sh:maxExclusive`
Basé sur les chaînes de caractères	`sh:minLength, sh:maxLength, sh:pattern, sh:stem, sh:uniqueLang`
Contraintes logiques	`sh:not, sh:and, sh:or, sh:xone`
« Shapes » fermés	`sh:closed, sh:ignoredProperties`
Contriantes entre deux propriétés	`sh:equals, sh:disjoint, sh:lessThan, sh:lessThanOrEquals`
Contraintes pas utilisés pour la validation	`sh:name, sh:value, sh:defaultValue`

Contraintes de cardinalités

Contraintes	Description
sh:minCount	Nombre mininal de triplets qui incluent le noeud et le prédicat ciblé. Valeur par défaut: 0
sh:maxCount	Idem à sh:minCount, mais selon le nombre maximale Par défaut, aucune limite

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :knows ;
    sh:minCount 2 ;
    sh:maxCount 3
  ] .
```

```
@prefix : <http://example.com/> .

:alice a :Person ;
  :knows :bob, :jean .

:bob a :Person ;
  :knows :alice .

:jean a :Person ;
  :knows :alice, :bob, :jean, :karol .
```

Contraintes sur les valeurs

Contraintes	Description
`sh:datatype`	Restriction du type de valeur possible.

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :birthday ;
    sh:datatype xsd:date
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:alice a :Person ;
  :birthday "1987-06-20"^^xsd:date .

:bob a :Person ;
  :birthday "Inconnu"^^xsd:date .

:jean a :Person ;
  :birthday "1990" .
```

Contraintes la classe des valeurs

Contraintes	Description
`sh:class`	Les valeurs doit être une instance de cette classe. La notion d'instance est définie par: rdf:type/rdfs:subClassOf*

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:LivingBeing a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :knows ;
    sh:class :LivingBeing ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person rdfs:subClassOf :LivingBeing .

:alice a :Person ;
  :knows :bob, :jean .

:bob :knows :jean .

:jean a :LivingBeing ;
  :knows :alice .
```

Contraintes le type de valeur d'un « shape »

Contraintes	Description
sh:nodeKind	sh:BlankNode, sh:IRI, sh:Literal, sh:BlankNodeOrIRI, sh:BlankNodeOrLiteral, sh:IRIOrLiteral

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:nodeKind sh:IRI ;
  sh:property [
    sh:path :name ;
    sh:minCount 0 ;
    sh:nodeKind sh:Literal .
  ]
```

```
@prefix : <http://example.com/> .

:alice a :Person ;
  :name :aliceName .

<http://example.com/Bob> a :Person .

_:b1 a :Person
```

Contraintes des valeurs possibles d'un « shape »

Contraintes	Description
sh:hasValue	Vérifie que le noeud ciblé et la propriété ciblée contiennent l'objet spécifié
sh:in	La valeur de l'objet fait partie d'une liste de valeurs

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :knows ;
    sh:hasValue :AlbertEinstein
  ] ;

  sh:property [
    sh:path :gender ;
    sh:in (:Male :Female)
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:alice a :Person ;
  :gender :Female ;
  :knows :Einstein .

:bob a :Person ;
  :gender :Male ;
  :knows :AlbertEinstein .

:jean a :Person ;
  :gender :Unknown ;
  :knows :AlbertEinstein .
```

Contraintes de valeurs vers un autre « shape »

Contraintes	Description
sh:node	Les valeurs d'une propriétés doivent satisfaire la « shape » visée. ATTENTION: la « shape » visée n'est pas obligé d'être du type visée comme l'exemple avec :Polytechnique

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :worksFor ;
    sh:node :Company ;
  ] .

:Company a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :name ;
    sh:minCount 1 ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>
.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:alice a :Person ;
  :worksFor :Polytechnique .

:bob a :Person ;
  :worksFor :MyCompany .

:Polytechnique :name "Polytechnique Montréal" .

:MyCompany rdfs:label "MyCompany" .
```

Comment éviter la récurcivité ?

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :worksFor ;
    sh:node :Company ;
  ] .

:Company a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :employee ;
    sh:node :Person ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:alice a :Person ;
  :worksFor :Polytechnique .

:bob a :Person ;
  :worksFor :Polytechnique .

:Polytechnique :name "Polytechnique Montréal" ;
  :employee :alice, :bob .
```

Comment éviter la récurcivité ?

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :worksFor ;
    sh:class :Company ;
  ] .

:Company a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :employee ;
    sh:class :Person ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:alice a :Person ;
  :worksFor :Polytechnique .

:bob a :Person ;
  :worksFor :Polytechnique .

:Polytechnique a :Company ;
  :name "Polytechnique Montréal" ;
  :employee :alice, :bob .
```

Opérateurs logiques

Contraintes	Description
`sh:and`	Conjonction d'une liste de property shape. Valeur par défaut
`sh:or`	Disjonction d'une liste de property shape.
`sh:not`	Négation d'un property shape
`sh:xone`	Au moins 1 property shape doit être validé

Opérateur sh:or

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:or (
    [
      sh:property [
        sh:path :name ;
        sh:minCount 1 ;
      ]
    ]
    [
      sh:property [
        sh:path rdfs:label ;
        sh:minCount 1 ;
      ]
    ]
  ) .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-
schema#> .

:alice a :Person ;
  :name "Alice" .

:bob a :Person ;
  rdfs:label "Bob" .

:jean a :Person .
```

Intervalle de valeurs

Contraintes
sh:minInclusive
sh:maxInclusive
sh:minExclusive
sh:maxExclusive

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :age ;
    sh:minIclusive 0 ;
    sh:maxInclusive 100 ;
    sh:datetype xsd:integer ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:alice a :Person ;
  :age "60"^^xsd:integer .

:bob a :Person ;
  :age "-1"^^xsd:integer .
```

Intervalle de valeurs

Contraintes	Description
sh:minLength sh:maxLength	Min/max du nombre de caractères de la valeur
sh:pattern	Vérifie que la valeur “match” l’expression régulière
sh:stem	Vérifie si les valeurs possède un préfixe spécifié
sh:uniqueLang	Vérifie que des langues différentes ne sont pas utilisées pour une même valeur

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :matricule ;
    sh:pattern "^p\\d{6}$" ;
    sh:flags "i" ;
  ] ;
  sh:property [
    sh:path :name ;
    sh:minLength 4 ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:alice a :Person ;
  :matricule "1530204" ;
  :name "Alice" .

:bob a :Person ;
  :matricule "p486034" ;
  :name "Bob" .
```

Contraintes entre deux propriétés

Contraintes	Description
sh:equals	Les valeurs des deux propriétés doivent être égales
sh:disjoint	Les valeurs des deux propriétés ne doivent pas être égales
sh:lessThan	La valeur d'une propriété doit être inférieure à l'autre propriété
sh:lessThanOrEquals	La valeur d'une propriété doit être inférieure ou égale à l'autre propriété

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:property [
    sh:path :name ;
    sh:equals :givenName ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:alice a :Person ;
  :name "Alice" ;
  :givenName "Alice" .

:bob a :Person ;
  :name "Bob" .

:jean a :Person ;
  :name "Jean" ;
  :givenName "JeanIV" .
```

Les shapes fermées

Contraintes	Description
sh:closed	Le noeud visé doit uniquement avoir les propriétés spécifiées dans un sh:property
sh:ignoredProperties	Liste optionnelle de propriétés qui peuvent être utilisées

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:Person a sh:NodeShape, rdfs:Class ;
  sh:closed true ;
  sh:ignoredProperties ( rdf:type ) ;
  sh:property [
    sh:path :name ;
    sh:minCount 1 ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:alice a :Person ;
  :name "Alice" .

:jean a :Person ;
  :name "Jean" ;
  :givenName "JeanIV" .
```

Exercice

- Représenter en SHACL les contraintes suivantes :
  - Un utilisateur possède zéro ou un prénom et exactement un nom
  - Le nom et le prénom doivent avoir des valeurs différentes
  - Un utilisateur possède au moins une coordonnée de communication. Cette coordonnée peut être soit une adresse courriel ou une adresse postale.

Exercice

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:User a sh:Nodeshape, rdfs:Class ;
  sh:property [
    sh:path :givenName ;
    sh:minCount 0 ;
    sh:maxCount 1 ;
    sh:nodeKind sh:Literal ;
  ] ;
  sh:property [
    sh:path :lastName ;
    sh:minCount 1 ;
    sh:maxCount 1 ;
    sh:nodeKind sh:Literal ;
  ] ;
  sh:property [
    sh:path :coordinate ;
    sh:minCount 1 ;
    sh:or (
      [ sh:class :EmailAddress ]
      [ sh:class :PostalAddress ]
    ) ;
    sh:nodeKind sh:IRI ;
  ] .
```

```
@prefix : <http://example.com/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:alice a :User ;
  :givenName "Alice" ;
  :lastName "Cooper" ;
  :coordinate :aliceEmail .

:aliceEmail a :EmailAddress .

:bob a :User ;
  :lastName "Bob" ;
  :coordinate :bobPostalAddress .

:bobPostalAddress a :PostalAddress .
```