Property-based Testing of SPARQL queries

Jesús M. Almendros-Jiménez
Antonio Becerra-Terón
University of Almería, SPAIN

16th International Symposium on Database Programming Languages (DBPL 2017), Munich, Germany

Type Checking and Testing of SPARQL

- SPARQL program bugs: debugging and testing
- Type Checking
  - Wrongly typed (empty answers
- Testing
  - Disagreement between expected behavior and answers (

Type Checking

```
SELECT ?Person ?Paper ?University

WHERE

{     ?Person

      ?Person

      ?Person

      ?Paper

      ?Paper

      FILTER (
```

Query:

Universities of Invited Talks of 2-authors papers

Type Checking

```
SELECT ?Person ?Paper ?University

WHERE

{     ?Person :attends :DBPL .

      ?Person :affiliation



      FILTER (?na >= 2)  }
```

Wrong typing:

has not

type :University

Wrong typing:

?University has not type

:Person

Wrong typing:

has not type :Paper

Testing

```
SELECT ?Person ?Paper ?University
WHERE
{     ?Person :attends :DBPL .
      ?Person :affiliation ?University .
      ?Paper rfd:type :Invited_talk .
      ?Paper :numberofAuthors ?na .
      FILTER (?na >= 2)  }
```

Missing Triple!

Unexpected behavior!
Wrong answers!!

Testing

```
SELECT ?Person ?Student ?University

WHERE

{     ?Person :author :Paper .

      ?Student :author Paper .

      ?Paper rfd:type  :Student_Paper .


      FILTER (?age <= 30)  }
```

Wrong Triple!

?Person :age ?age

Unexpected behavior!
Wrong answers!!

Testing

```
SELECT ?Person ?Paper ?University

{

?Person :attends :DBPL .

?Person :affiliation ?University .

?Paper rfd:type :Invited_talk

?Paper :numberofAuthors ?na

FILTER (?na >= 2)

}
```

Property:

is ?Person the author of ?Paper ?

Testing

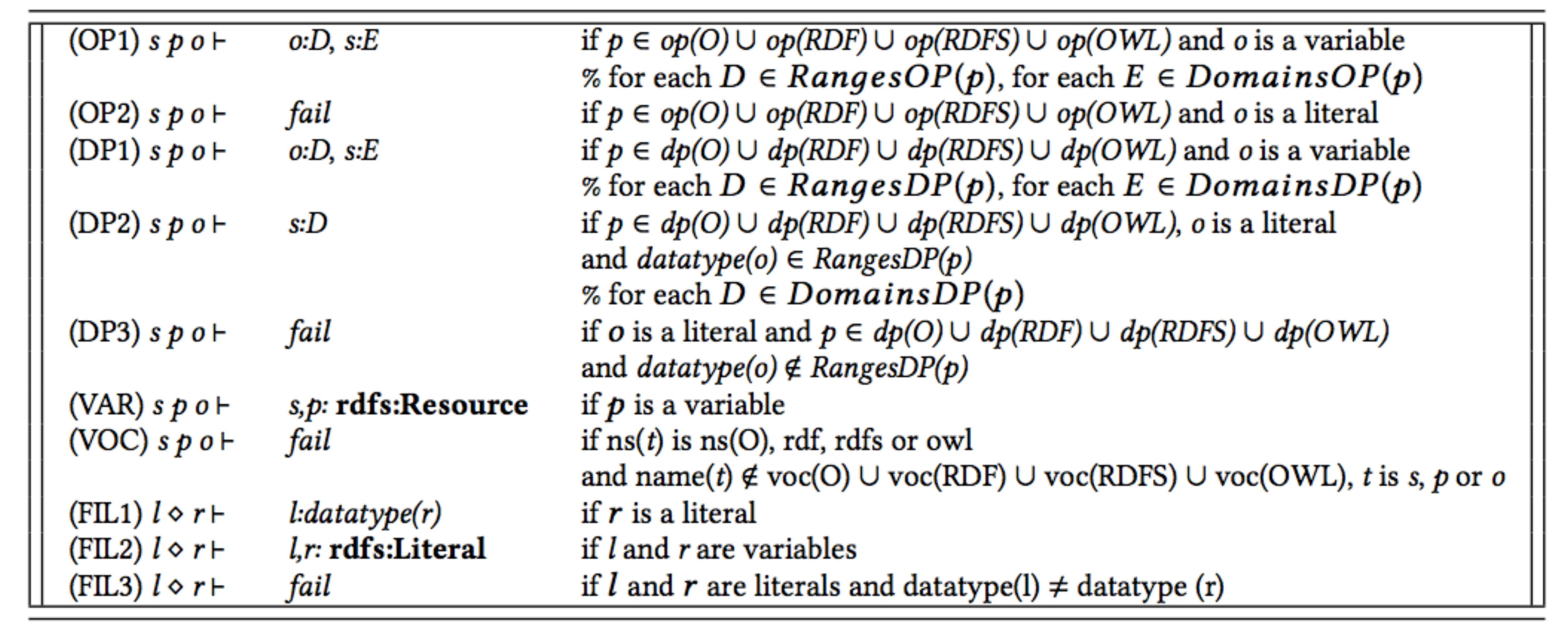
```
SELECT ?Person ?Student ?University

WHERE

{     ?Person :author :Paper .

      ?Student :author Paper .

      ?Paper rfd:type  :Student_paper .

      ?Person :age  ?age.

      FILTER (?age <= 30)  }
```

Property:

is ?Student

younger than 30?

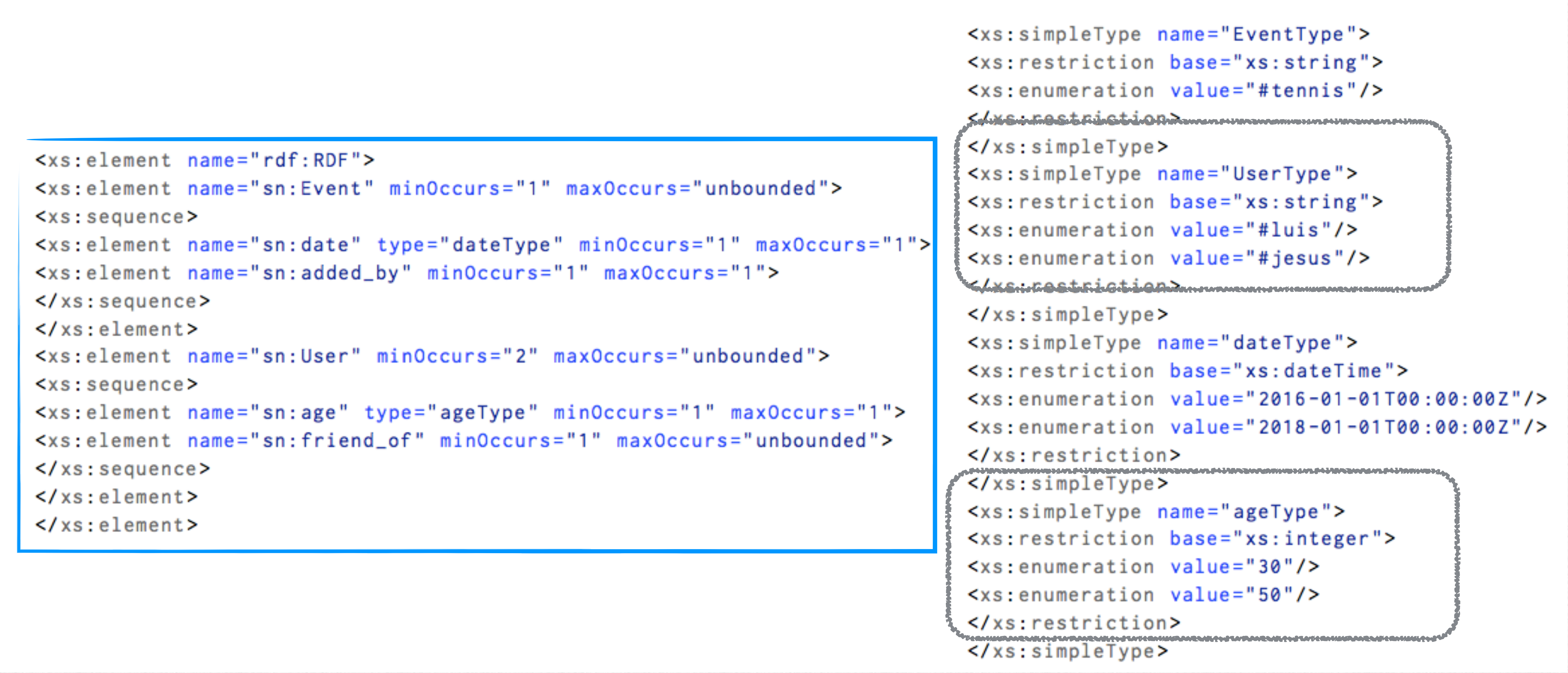
Type System for SPARQL



(OP1) $s\ p\ o \vdash$	$o{:}D,\ s{:}E$	if $p \in op(O) \cup op(RDF) \cup op(RDFS) \cup op(OWL)$ and o is a variable % for each $D \in RangesOP(p)$, for each $E \in DomainsOP(p)$
(OP2) $s\ p\ o \vdash$	*fail*	if $p \in op(O) \cup op(RDF) \cup op(RDFS) \cup op(OWL)$ and o is a literal
(DP1) $s\ p\ o \vdash$	$o{:}D,\ s{:}E$	if $p \in dp(O) \cup dp(RDF) \cup dp(RDFS) \cup dp(OWL)$ and o is a variable % for each $D \in RangesDP(p)$, for each $E \in DomainsDP(p)$
(DP2) $s\ p\ o \vdash$	$s{:}D$	if $p \in dp(O) \cup dp(RDF) \cup dp(RDFS) \cup dp(OWL)$, o is a literal and $datatype(o) \in RangesDP(p)$ % for each $D \in DomainsDP(p)$
(DP3) $s\ p\ o \vdash$	*fail*	if o is a literal and $p \in dp(O) \cup dp(RDF) \cup dp(RDFS) \cup dp(OWL)$ and $datatype(o) \notin RangesDP(p)$
(VAR) $s\ p\ o \vdash$	s,p: **rdfs:Resource**	if p is a variable
(VOC) $s\ p\ o \vdash$	*fail*	if ns(t) is ns(O), rdf, rdfs or owl and name(t) $\notin$ voc(O) $\cup$ voc(RDF) $\cup$ voc(RDFS) $\cup$ voc(OWL), t is s, p or o
(FIL1) $l \diamond r \vdash$	$l{:}datatype(r)$	if r is a literal
(FIL2) $l \diamond r \vdash$	l,r: **rdfs:Literal**	if l and r are variables
(FIL3) $l \diamond r \vdash$	*fail*	if l and r are literals and datatype(l) $\neq$ datatype (r)

Testing of SPARQL

Ontology to XML Schema Mapping
Test Case Generation



```
<xs:element name="rdf:RDF">
<xs:element name="sn:Event" minOccurs="1" maxOccurs="unbounded">
<xs:sequence>
<xs:element name="sn:date" type="dateType" minOccurs="1" maxOccurs="1">
<xs:element name="sn:added_by" minOccurs="1" maxOccurs="1">
</xs:sequence>
</xs:element>
<xs:element name="sn:User" minOccurs="2" maxOccurs="unbounded">
<xs:sequence>
<xs:element name="sn:age" type="ageType" minOccurs="1" maxOccurs="1">
<xs:element name="sn:friend_of" minOccurs="1" maxOccurs="unbounded">
</xs:sequence>
</xs:element>
</xs:element>
```

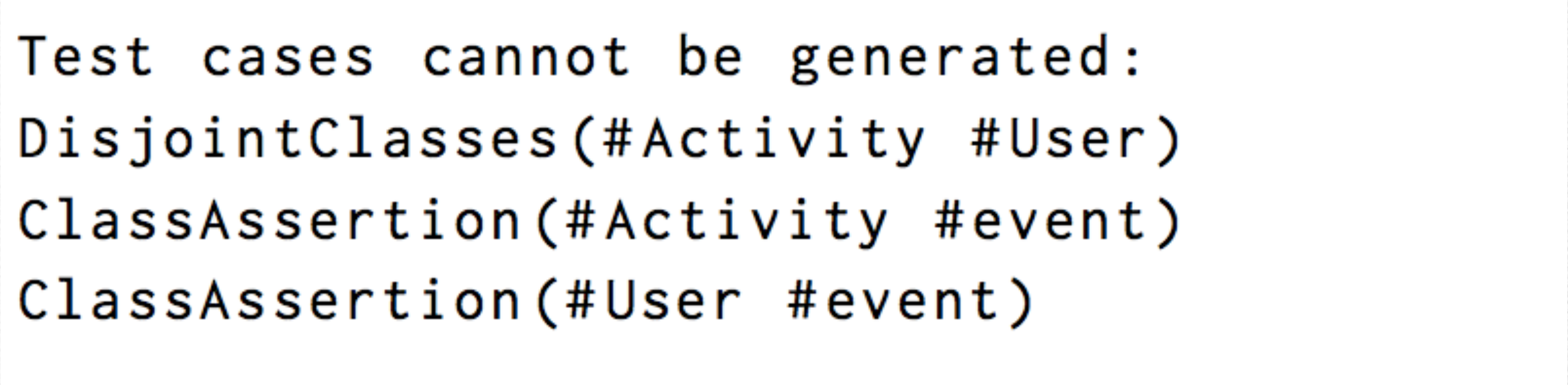
```
<xs:simpleType name="EventType">
<xs:restriction base="xs:string">
<xs:enumeration value="#tennis"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="UserType">
<xs:restriction base="xs:string">
<xs:enumeration value="#luis"/>
<xs:enumeration value="#jesus"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="dateType">
<xs:restriction base="xs:dateTime">
<xs:enumeration value="2016-01-01T00:00:00Z"/>
<xs:enumeration value="2018-01-01T00:00:00Z"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="ageType">
<xs:restriction base="xs:integer">
<xs:enumeration value="30"/>
<xs:enumeration value="50"/>
</xs:restriction>
</xs:simpleType>
```

Type Checking and Testing of SPARQL

- Type Checking using an Ontology Reasoner (Hermit) for
- Testing using an Ontology Reasoner (Hermit) for test cases consistence
- Testing using

Type Checking and Testing of SPARQL

Type Checking Tool results:



```
Test cases cannot be generated:
DisjointClasses(#Activity #User)
ClassAssertion(#Activity #event)
ClassAssertion(#User #event)
```

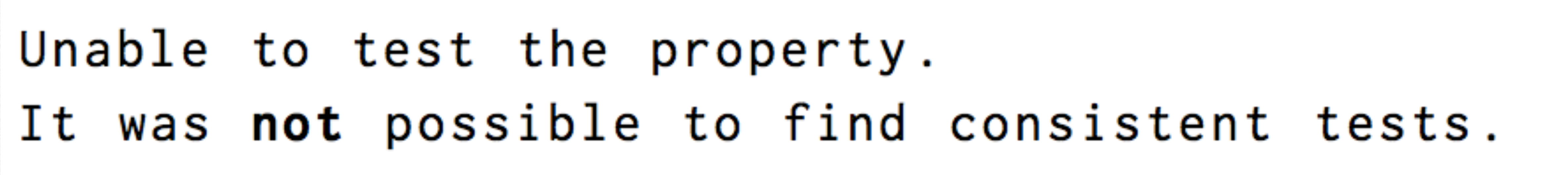
Type Checking and Testing of SPARQL

Randomly generated test cases

Problem => Inconsistent Test Cases

Example:

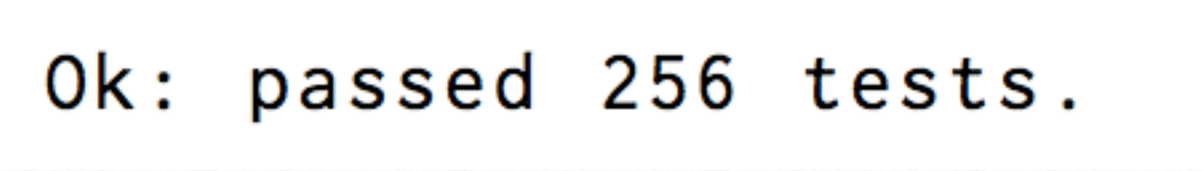
:coauthor is irreflexive



```
Unable to test the property.
It was not possible to find consistent tests.
```

Type Checking and Testing of SPARQL

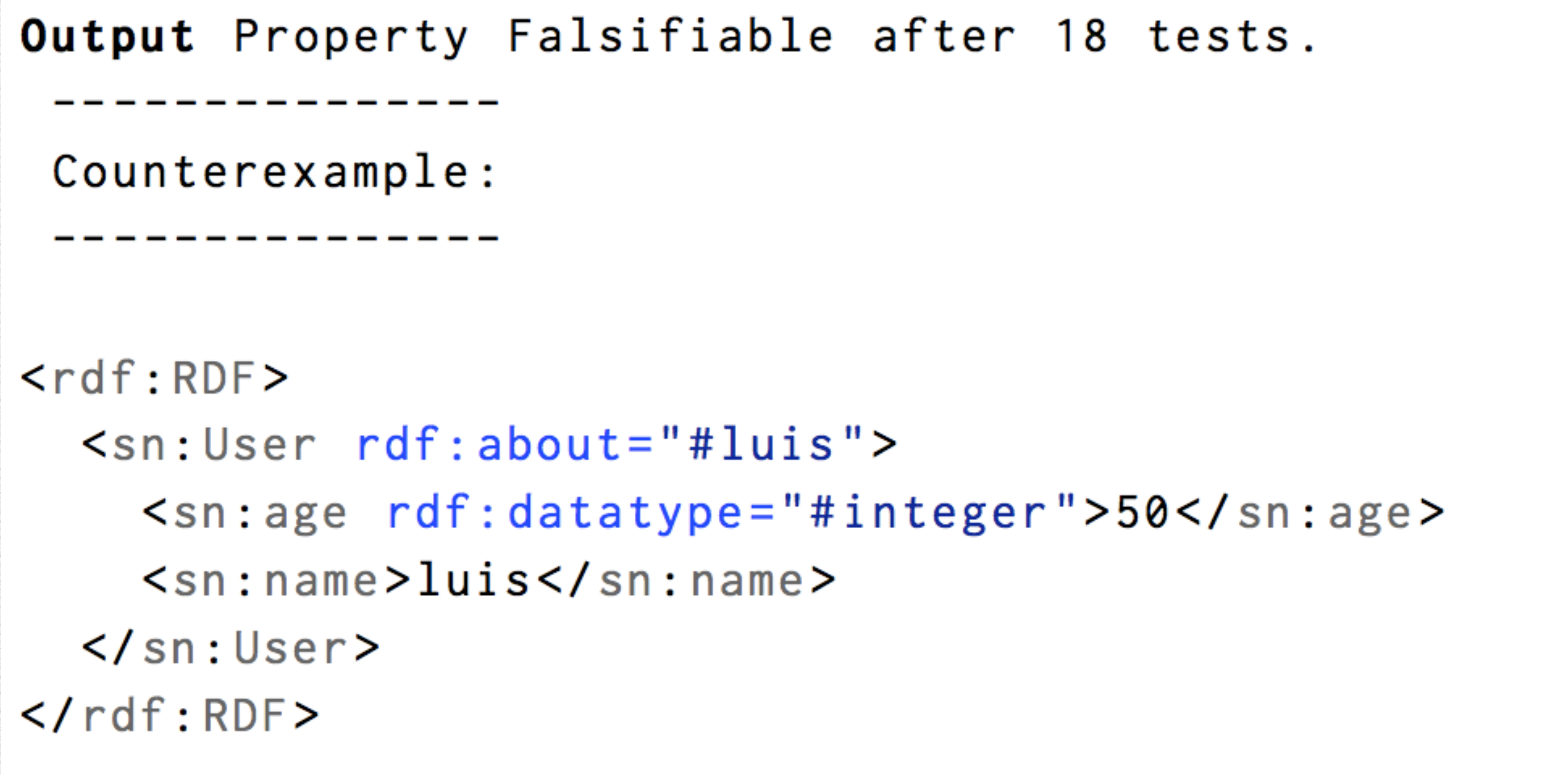
Testing success



```
Ok: passed 256 tests.
```

Type Checking and Testing of SPARQL

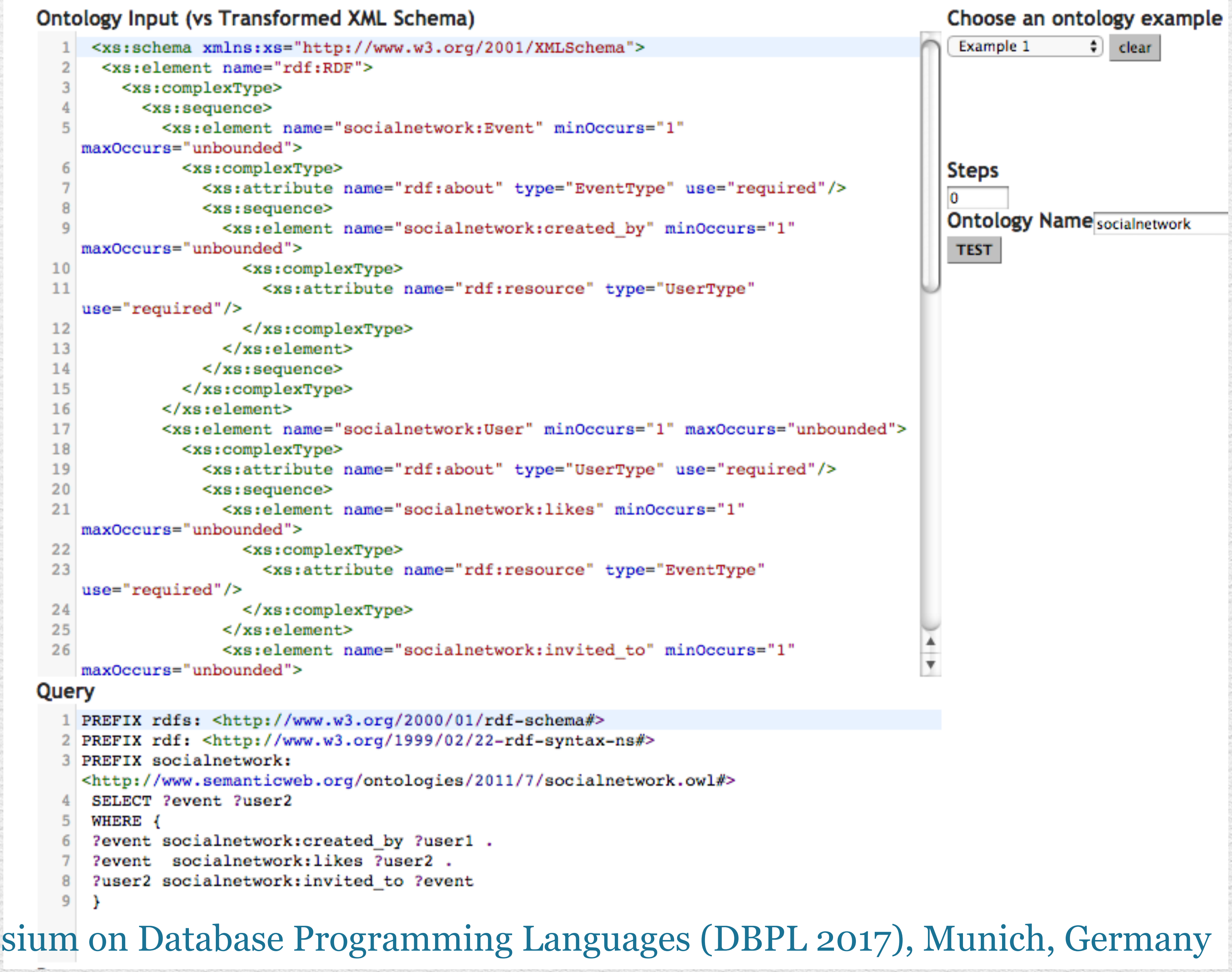
Testing fail



```
Output Property Falsifiable after 18 tests.
 ----------------
 Counterexample:
 ----------------


<rdf:RDF>
  <sn:User rdf:about="#luis">
    <sn:age rdf:datatype="#integer">50</sn:age>
    <sn:name>luis</sn:name>
  </sn:User>
</rdf:RDF>
```

Type Checking and Testing Tool

http://minerva.ual.es:8080/SPARQL



Ontology Input (vs Transformed XML Schema)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="rdf:RDF">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="socialnetwork:Event" minOccurs="1" maxOccurs="unbounded">
         <xs:complexType>
           <xs:attribute name="rdf:about" type="EventType" use="required"/>
           <xs:sequence>
             <xs:element name="socialnetwork:created_by" minOccurs="1" maxOccurs="unbounded">
               <xs:complexType>
                 <xs:attribute name="rdf:resource" type="UserType" use="required"/>
               </xs:complexType>
             </xs:element>
           </xs:sequence>
         </xs:complexType>
       </xs:element>
       <xs:element name="socialnetwork:User" minOccurs="1" maxOccurs="unbounded">
         <xs:complexType>
           <xs:attribute name="rdf:about" type="UserType" use="required"/>
           <xs:sequence>
             <xs:element name="socialnetwork:likes" minOccurs="1" maxOccurs="unbounded">
               <xs:complexType>
                 <xs:attribute name="rdf:resource" type="EventType" use="required"/>
               </xs:complexType>
             </xs:element>
             <xs:element name="socialnetwork:invited_to" minOccurs="1" maxOccurs="unbounded">
```

Choose an ontology example

Example 1 clear

Steps: 0

Ontology Name: socialnetwork

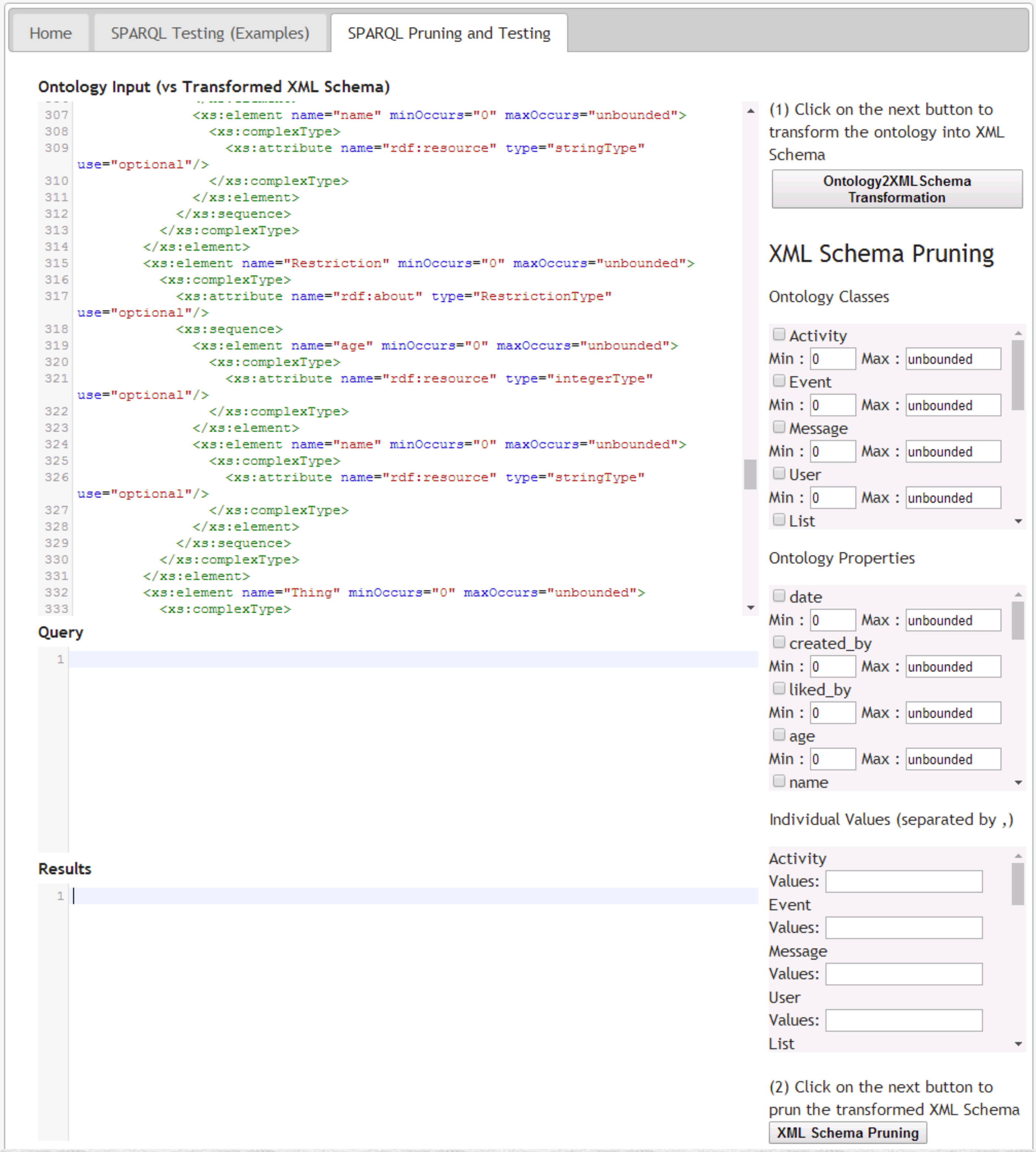
TEST

Query

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX socialnetwork: <http://www.semanticweb.org/ontologies/2011/7/socialnetwork.owl#>
 SELECT ?event ?user2
 WHERE {
 ?event socialnetwork:created_by ?user1 .
 ?event  socialnetwork:likes ?user2 .
 ?user2 socialnetwork:invited_to ?event
 }
```

Type Checking and Testing Tool

Pruning and Customization of Test Cases



Ontology Input (vs Transformed XML Schema)

```
            <xs:element name="name" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:attribute name="rdf:resource" type="stringType"
use="optional"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Restriction" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="rdf:about" type="RestrictionType"
use="optional"/>
          <xs:sequence>
            <xs:element name="age" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:attribute name="rdf:resource" type="integerType"
use="optional"/>
              </xs:complexType>
            </xs:element>
            <xs:element name="name" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:attribute name="rdf:resource" type="stringType"
use="optional"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Thing" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
```

Query

Results

(1) Click on the next button to transform the ontology into XML Schema

XML Schema Pruning

Ontology Classes

□ Activity
Min : 0 Max : unbounded
□ Event
Min : 0 Max : unbounded
□ Message
Min : 0 Max : unbounded
□ User
Min : 0 Max : unbounded
□ List

Ontology Properties

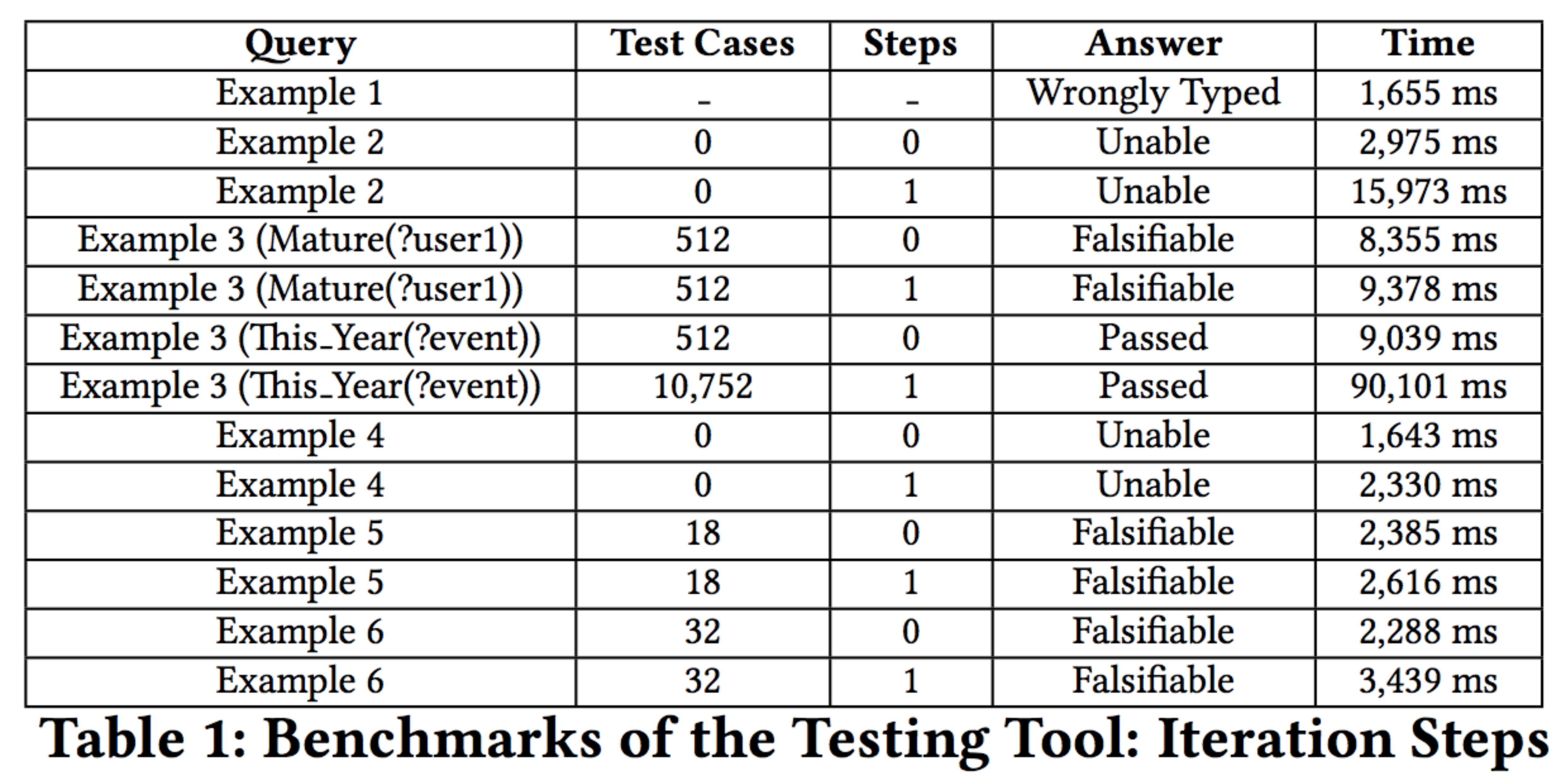
□ date
Min : 0 Max : unbounded
□ created_by
Min : 0 Max : unbounded
□ liked_by
Min : 0 Max : unbounded
□ age
Min : 0 Max : unbounded
□ name

Individual Values (separated by ,)

Activity
Values:
Event
Values:
Message
Values:
User
Values:
List

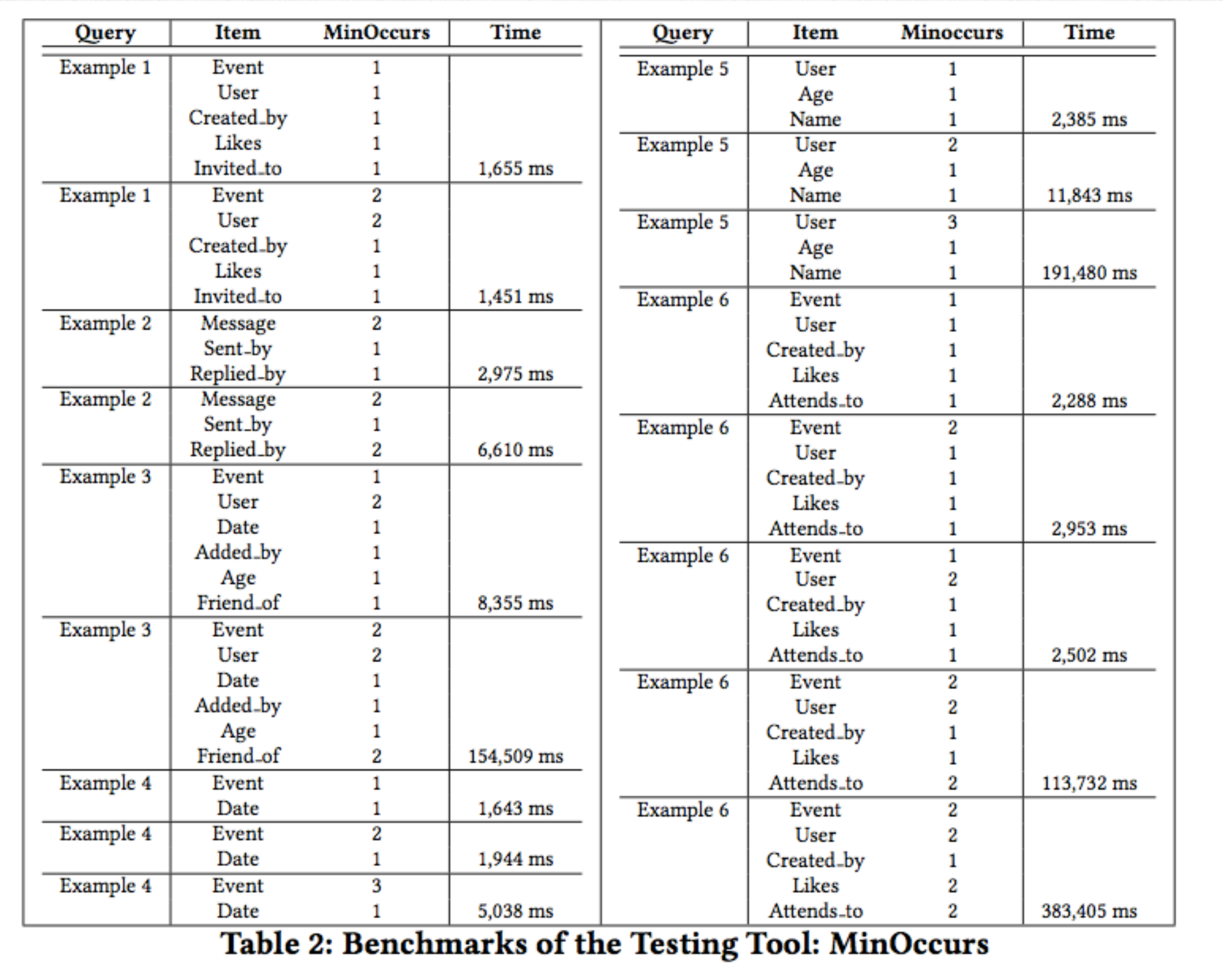
(2) Click on the next button to prun the transformed XML Schema

Benchmarks



Query	Test Cases	Steps	Answer	Time
Example 1	_	_	Wrongly Typed	1,655 ms
Example 2	0	0	Unable	2,975 ms
Example 2	0	1	Unable	15,973 ms
Example 3 (Mature(?user1))	512	0	Falsifiable	8,355 ms
Example 3 (Mature(?user1))	512	1	Falsifiable	9,378 ms
Example 3 (This_Year(?event))	512	0	Passed	9,039 ms
Example 3 (This_Year(?event))	10,752	1	Passed	90,101 ms
Example 4	0	0	Unable	1,643 ms
Example 4	0	1	Unable	2,330 ms
Example 5	18	0	Falsifiable	2,385 ms
Example 5	18	1	Falsifiable	2,616 ms
Example 6	32	0	Falsifiable	2,288 ms
Example 6	32	1	Falsifiable	3,439 ms

Table 1: Benchmarks of the Testing Tool: Iteration Steps



Query	Item	MinOccurs	Time
Example 1	Event	1	
	User	1	
	Created_by	1	
	Likes	1	
	Invited_to	1	1,655 ms
Example 1	Event	2	
	User	2	
	Created_by	1	
	Likes	1	
	Invited_to	1	1,451 ms
Example 2	Message	2	
	Sent_by	1	
	Replied_by	1	2,975 ms
Example 2	Message	2	
	Sent_by	1	
	Replied_by	2	6,610 ms
Example 3	Event	1	
	User	2	
	Date	1	
	Added_by	1	
	Age	1	
	Friend_of	1	8,355 ms
Example 3	Event	2	
	User	2	
	Date	1	
	Added_by	1	
	Age	1	
	Friend_of	2	154,509 ms
Example 4	Event	1	
	Date	1	1,643 ms
Example 4	Event	2	
	Date	1	1,944 ms
Example 4	Event	3	
	Date	1	5,038 ms
Query	**Item**	**Minoccurs**	**Time**
Example 5	User	1	
	Age	1	
	Name	1	2,385 ms
Example 5	User	2	
	Age	1	
	Name	1	11,843 ms
Example 5	User	3	
	Age	1	
	Name	1	191,480 ms
Example 6	Event	1	
	User	1	
	Created_by	1	
	Likes	1	
	Attends_to	1	2,288 ms
Example 6	Event	2	
	User	1	
	Created_by	1	
	Likes	1	
	Attends_to	1	2,953 ms
Example 6	Event	1	
	User	2	
	Created_by	1	
	Likes	1	
	Attends_to	1	2,502 ms
Example 6	Event	2	
	User	2	
	Created_by	1	
	Likes	1	
	Attends_to	2	113,732 ms
Example 6	Event	2	
	User	2	
	Created_by	1	
	Likes	2	
	Attends_to	2	383,405 ms

Table 2: Benchmarks of the Testing Tool: MinOccurs

Conclusions and Future Work

- Testing tool based on Test Case Generation from XML Schema
- Detection of Buggy SPARQL Queries
- Extending the output property set
- White-Box Testing: Constraint Solving

Thanks for your Attention

Jesús M. Almendros-Jiménez
Antonio Becerra-Terón
University of Almería SPAIN