ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ · ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ

2η ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ

1α ΕΡΩΤΗΜΑ

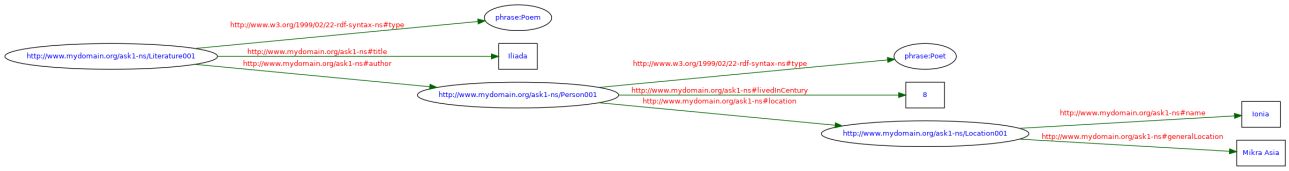
Στην πρόταση,

«Η Ιλιάδα συνθέθηκε απο ποιητή που έζησε τον 8ο αιώνα π.Χ. στην Ιωνία της Μικράς Ασίας»,

θεωρούμε ως υποκείμενο την Ιλιάδα (αναπαρίσταται ως τη ξεχωριστή οντότητα, `Literature001`), ως κατηγόρημα το "συνθέθηκε" (που αντιστοιχεί στο `author`) και ως αντικείμενο τον ποιητή (αναπαρίσταται ως τη οντότητα `Person001`). Το `Person001` είναι τύπου `Poet` και περιλαμβάνει τα ορίσματα `livedInCentury` και `location`. Αυτά τα ορίσματα καλύπτουν την αναφορική πρόταση που επεξηγεί το πότε και πού έζησε ο ποιητής. Το `livedInCentury` περιλαμβάνει το λεκτικό "8", ενώ το `location` περιλαμβάνει ως resource την οντότητα `Location001`, η αποτελείται από τα ορίσματα `name` (Ιωνία) και `generalLocation` (Μικρά Ασία).

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:phrase="http://www.mydomain.org/ask1-ns#">

    <rdf:Description rdf:about="http://www.mydomain.org/ask1-ns/Literature001">
        <rdf:type rdf:resource="phrase:Poem"/>
        <phrase:title>Iliada</phrase:title>
        <phrase:author rdf:resource="http://www.mydomain.org/ask1-ns/Person001"/>
    </rdf:Description>

    <rdf:Description rdf:about="http://www.mydomain.org/ask1-ns/Person001">
        <rdf:type rdf:resource="phrase:Poet"/>
        <phrase:livedInCentury>8</phrase:livedInCentury>
        <phrase:location rdf:resource="http://www.mydomain.org/ask1-ns/Location001"/>
    </rdf:Description>

    <rdf:Description rdf:about="http://www.mydomain.org/ask1-ns/Location001">
        <phrase:name>Ionia</phrase:name>
        <phrase:generalLocation>Mikra Asia</phrase:generalLocation>
    </rdf:Description>

</rdf:RDF>
```

Number	Subject	Predicate	Object
1	http://www.mydomain.org/ask1-ns/Literature001	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	phrase:Poem
2	http://www.mydomain.org/ask1-ns/Literature001	http://www.mydomain.org/ask1-ns#title	"Iliada"
3	http://www.mydomain.org/ask1-ns/Literature001	http://www.mydomain.org/ask1-ns#author	http://www.mydomain.org/ask1-ns/Person001
4	http://www.mydomain.org/ask1-ns/Person001	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	phrase:Poet
5	http://www.mydomain.org/ask1-ns/Person001	http://www.mydomain.org/ask1-ns#livedInCentury	"8"
6	http://www.mydomain.org/ask1-ns/Person001	http://www.mydomain.org/ask1-ns#location	http://www.mydomain.org/ask1-ns/Location001
7	http://www.mydomain.org/ask1-ns/Location001	http://www.mydomain.org/ask1-ns#name	"Ionia"
8	http://www.mydomain.org/ask1-ns/Location001	http://www.mydomain.org/ask1-ns#generalLocation	"Mikra Asia"



(Ο γράφος περιλαμβάνεται ως ξεχωριστή εικόνα στο .zip)

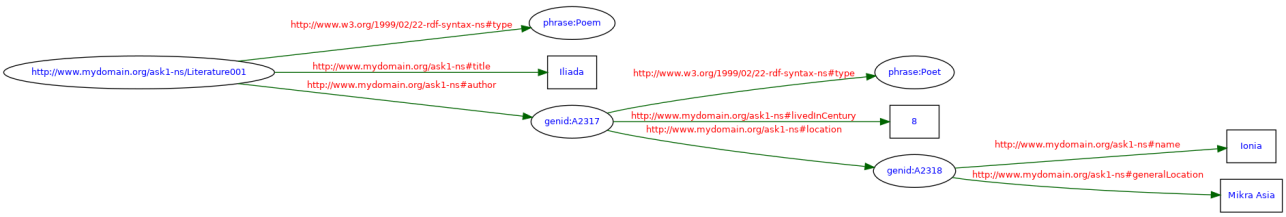
1β ΕΡΩΤΗΜΑ

Στο (α) ερώτημα είχαμε ονοματίσει τις οντότητες (ως `Literature001`, `Person001` κτλ.) και έπειτα τις ορίσαμε σε άλλο σημείο του κώδικα και τις συνδέσαμε ως resources.

Σε αυτή τη περίπτωση για την δημιουργία κενών κόμβων θα κάνουμε τον ορισμό τους απευθείας στο σημείο όπου τις είχαμε συνδέσει μέσω του resource. Έτσι παραμένουν «ανώνυμες», έχοντας ως αποτέλεσμα κενούς κόμβους.

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:phrase="http://www.mydomain.org/ask1-ns#">

    <rdf:Description rdf:about="http://www.mydomain.org/ask1-ns/Literature001">
         <rdf:type rdf:resource="phrase:Poem"/>
         <phrase:title>Iliada</phrase:title>
        <phrase:author>
            <rdf:Description>
                <rdf:type rdf:resource="phrase:Poet"/>
                <phrase:livedInCentury>8</phrase:livedInCentury>
                <phrase:location>
                    <rdf:Description>
                        <phrase:name>Ionia</phrase:name>
                        <phrase:generalLocation>Mikra Asia</phrase:generalLocation>
                    </rdf:Description>
                </phrase:location>
            </rdf:Description>
        </phrase:author>
  </rdf:Description>

</rdf:RDF>
```

Number	Subject	Predicate	Object
1	http://www.mydomain.org/ask1-ns/Literature001	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	phrase:Poem
2	http://www.mydomain.org/ask1-ns/Literature001	http://www.mydomain.org/ask1-ns#title	"Iliada"
3	http://www.mydomain.org/ask1-ns/Literature001	http://www.mydomain.org/ask1-ns#author	genid:A2317
4	genid:A2317	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	phrase:Poet
5	genid:A2317	http://www.mydomain.org/ask1-ns#livedInCentury	"8"
6	genid:A2317	http://www.mydomain.org/ask1-ns#location	genid:A2318
7	genid:A2318	http://www.mydomain.org/ask1-ns#name	"Ionia"
8	genid:A2318	http://www.mydomain.org/ask1-ns#generalLocation	"Mikra Asia"



(Ο γράφος περιλαμβάνεται ως ξεχωριστή εικόνα στο .zip)

2 ΕΡΩΤΗΜΑ

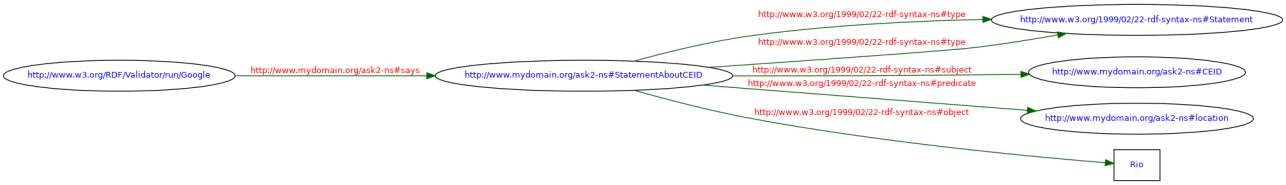
«Η Google αναφέρει ότι το Τμήμα Η/Υ & Πληροφορικής βρίσκεται στο Ρίο.»

Θεωρούμε ως `StatementAboutCEID` τη φράση "Το Τμήμα Η/Υ & Πληροφορικής βρίσκεται στο Ρίο". Σε αυτό το statement, subject είναι το `CEID`, predicate το `location` και object το `Rio`.

Τέλος δημιουργούμε μια οντότητα `Google` που περιλαμβάνει την ιδιότητα `says` η οποία έχει συνδεδεμένο resource το `StatementAboutCEID`.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:phrase="http://www.mydomain.org/ask2-ns#">

    <rdf:Statement rdf:about="http://www.mydomain.org/ask2-ns#StatementAboutCEID">
        <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
        <rdf:subject rdf:resource="http://www.mydomain.org/ask2-ns#CEID"/>
        <rdf:predicate rdf:resource="http://www.mydomain.org/ask2-ns#location"/>
        <rdf:object>Rio</rdf:object>
    </rdf:Statement>

    <rdf:Description rdf:about="Google">
        <phrase:says rdf:resource="http://www.mydomain.org/ask2-ns#StatementAboutCEID"/>
    </rdf:Description>

</rdf:RDF>
```

Number	Subject	Predicate	Object
1	http://www.mydomain.org/ask2-ns#StatementAboutCEID	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement
2	http://www.mydomain.org/ask2-ns#StatementAboutCEID	http://www.w3.org/1999/02/22-rdf-syntax-ns#subject	http://www.mydomain.org/ask2-ns#CEID
3	http://www.mydomain.org/ask2-ns#StatementAboutCEID	http://www.w3.org/1999/02/22-rdf-syntax-ns#predicate	http://www.mydomain.org/ask2-ns#location
4	http://www.mydomain.org/ask2-ns#StatementAboutCEID	http://www.w3.org/1999/02/22-rdf-syntax-ns#object	"Rio"
5	http://www.mydomain.org/ask2-ns#StatementAboutCEID	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement
6	http://www.w3.org/RDF/Validator/run/Google	http://www.mydomain.org/ask2-ns#says	http://www.mydomain.org/ask2-ns#StatementAboutCEID



(Ο γράφος περιλαμβάνεται ως ξεχωριστή εικόνα στο .zip)

3 ΕΡΩΤΗΜΑ

Όλο το RDFS αρχείο συμπεριλαμβάνεται στο .zip.

Δημιουργία κλάσεων

Η δομή κάθε ορισμού κλάσης είναι η εξής:

```
<rdf:Description rdf:about="http://www.mydomain.org/uni-ns#ΟΝΟΜΑ_ΚΛΑΣΗΣ">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Class" />
</rdf:Description>
```

Εναλλακτικά του Description θα μπορούσαμε απευθείας να γράψουμε `Class`, παραλείποντας την δεύτερη γραμμή. Για να δηλώσουμε ότι μια κλάση είναι subclass μιας άλλης, χρησιμοποιούμε το `rdfs:subClassOf` με `resource` την υπερκλάση. Για παράδειγμα για τη κλάση του `Professor`:

```
<rdf:Description rdf:about="http://www.mydomain.org/uni-ns#Professor">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Class" />
    <rdfs:subClassOf rdf:resource="http://www.mydomain.org/uni-ns#Person"/>
</rdf:Description>
```

Δημιουργία ιδιότητων

Η δομή κάθε ορισμού ιδιότητας είναι εξής:

```
<rdf:Property rdf:about="http://www.mydomain.org/uni-ns#ΟΝΟΜΑ_ΙΔΙΟΤΗΤΑΣ ">
    <rdfs:domain rdf:resource="http://www.mydomain.org/uni-ns#ΟΝΟΜΑ_DOMAIN"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema/ΟΝΟΜΑ_RANGE"/>
</rdf:Property>
```

Σε συμφωνία με την εκφώνηση ορίζουμε τις ιδιότητες που ζητούνται. Στην περίπτωση της ιδιότητας `has_age` όπου ζητείται integer, χρησιμοποιούμε και εκεί `Literal`, αφού ο ακέραιος είναι υποπερίπτωση της λεκτικής κλάσης.

Δημιουργία στιγμιοτύπων

Ορίζουμε το `rdf:about` όλων των στιγμιοτύπων ως **ΣΤΙΓΜΙΟΤΥΠΟ_XXX**, με τα `XXX` να αναπαριστούν έναν αύξοντα τριψήφιο αριθμό. Στην περίπτωση των 6 `Department` που έχουν δημιουργηθεί, έχουμε ορίσει τις λεκτικές ιδιότητες `dep_name` και `dep_city`.

Όσον αφορά τους καθηγητές, έχει οριστεί ο τύπος τους ως `Professor`, και στην συνέχεια προσδιορίστηκαν οι παρακάτω ιδιότητες για δέκα περιπτώσεις καθηγητών. Οι ιδιότητες είναι αυτές που έχουν ως domain το Professor ή το Person (αφού το Professor είναι υποκλάση του). Στις περιπτώσεις των `member_of` και `teaches` έχουν δηλωθεί τα URLs ως resources της ιδιότητας.

PROFESSORS	
has_name	λεκτικό
has_phone	λεκτικό
has_email	λεκτικό
has_age	λεκτικό
member_of	Department
teaches	Lesson

Ορίστηκαν αντίστοιχα οι παρακάτω ιδιότητες για 20 φοιτητές:

STUDENTS	
has_name	λεκτικό
has_phone	λεκτικό
has_email	λεκτικό
has_age	λεκτικό
member_of	Department

και αντίστοιχα για τα μαθήματα. Ο τύπος της κλάσης των μαθημάτων δεν έχει οριστεί.

LESSONS	
les_name	λεκτικό
taught_by	Professor

Για την δημιουργία των αιθουσών ορίζουμε το τύπο ως Classroom και τις υπόλοιπες ιδιότητες. Στο `room_department` χρησιμοποιούμε ως resource τα URLs των Departments. Έχουν δημιουργηθεί 9 αίθουσες.

CLASSROOMS	
room_name	λεκτικό
room_capacity	λεκτικό
room_department	Department

4i ΕΡΩΤΗΜΑ

Όλοι οι .rq κώδικες συμπεριλαμβάνονται στο .zip. Περιλαμβάνονται επίσης .txt αρχεία με το output της κονσόλας.

```
1  SELECT ?phone
2  WHERE {
3      ?prof rdf:type uni:Professor ;
4          [1]    uni:has_phone ?phone
5  }
```

Στη γραμμή 3 μεταφέρονται στην `?prof` όλα τα instances που έχουν τύπο Professor (πρακτικά τα `ProfessorXXX`). Εξ' αυτών, στην επόμενη τριάδα στη γραμμή 4, τα δεδομένα της ιδιότητας `has_phone` μεταφέρονται στην μεταβλητή `?phone`. Στην γραμμή 1 κάνουμε `SELECT` αυτή τη μεταβλητή, άρα το αποτέλεσμα είναι η εμφάνιση ενός πίνακα με τα τηλέφωνα των καθηγητών.

phone
"2610996975"
"2610996929"
"2610996968"
"2610996909"
"2610996915"
"2610996990"
"2610997505"
"2610996911"
"2610996912"
"2610996924"

Για πρακτικούς λόγους μπορούμε να προσθέσουμε την `?name`, η οποία αφορά τα δεδομένα της ιδιότητας `has_name`, ώστε στον πίνακα να εμφανίζονται και τα ονόματα των καθηγητών.

```
1  SELECT ?prof_name ?prof_phone
2  WHERE {
3      ?prof_inst  rdf:type uni:Professor ;
4                  uni:has_name ?prof_name   ;
5                  uni:has_phone ?prof_phone
6  }
```

prof_name	prof_phone
"Konstantinos Mpermperidis"	"2610996975"
"Nikolos Dimitris"	"2610996929"
"Makris Christos"	"2610996968"
"Ioannis Garofalakis"	"2610996909"
"Andreas Komninos"	"2610996915"
"Kyriakos Vlachos"	"2610996990"
"Stavros Kosmadakis"	"2610997505"
"Efstratios Galopoulos"	"2610996911"
"Christos Zaroliagis"	"2610996912"
"Haris Vergos"	"2610996924"

[1] Με την προσθήκη του ";" στο τέλος της προηγούμενης γραμμής, εννοείται το υποκείμενο ?prof και στην επόμενη γραμμή.

4ii ΕΡΩΤΗΜΑ

```
SELECT ?stud_name ?stud_phone
WHERE {
   ?stud_inst  rdf:type uni:Student ;
                uni:has_name ?stud_name   ;
                uni:has_phone ?stud_phone   ;
                uni:has_age ?stud_age
                FILTER (?stud_age>"23")
}
```

Με την `?stud_inst` επιλέγουμε όλα τα instances που αφορούν Students. Στις μεταβλητές `?stud_name`, `?stud_phone`, `?stud_age` αποθηκεύουμε τα ονόματα, τα τηλέφωνα και τις ηλικίες των φοιτητών. Από αυτές, επιλέγουμε να εμφανίζονται τα τηλέφωνα και επιπλέον τα ονόματα για λόγους ευχρηστίας. Χρησιμοποιώντας το `FILTER` επιλέγουμε την εμφάνιση μόνο των instances όπου έχουν `?stud_age > 23` (ηλικία άνω των 23). Το αποτέλεσμα είναι ο παρακάτω πίνακας:

stud_name	stud_phone
"Student Fourteen"	"2610900014"
"Student Nineteen"	"2610900019"
"Student Six"	"2610900006"
"Student Fiveteen"	"2610900015"
"Student Nine"	"2610900009"
"Student Sixteen"	"2610900016"

4iii ΕΡΩΤΗΜΑ

Μπορούμε να εμφανίσουμε όλα τα instances που ανήκουν στην κλάση Persons με τον παρακάτω τρόπο:

```
SELECT ?x ?name ?depart
WHERE {
     ?x        rdf:type/rdfs:subClassOf* uni:Person;
                       uni:has_name ?name;
                       uni:member_of/uni:resource ?depart
}
```

x	name	depart
uni:Student001	"Student One"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor008	"Andreas Komninos"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student002	"Student Two"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student006	"Student Six"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student008	"Student Eight"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor003	"Haris Vergos"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student016	"Student Sixteen"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student018	"Student Eighteen"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student012	"Student Twelve"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student020	"Student Twenty"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor002	"Nikolos Dimitris"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor009	"Stavros Kosmadakis"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student009	"Student Nine"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor010	"Konstantinos Mpermperidis"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student015	"Student Fiveteen"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor007	"Christos Zaroliagis"	"http://www.mydomain.org/uni-ns#Department001"
uni:Student011	"Student Eleven"	"http://www.mydomain.org/uni-ns#Department001"
uni:Professor006	"Ioannis Garofalakis"	"http://www.mydomain.org/uni-ns#Department001"

[2]

Η προσθήκη "`/rdfs:subClassOf*`" ήταν αναγκαία για την εμφάνιση όλων των αποτελεσμάτων. Με αυτό τον τρόπο η SPARQL ελέγχει και επιστρέφει κάποιο από αυτά τα δύο πιθανά paths, τα οποία έχουν ως αντικείμενο το `uni:Person`. Χρησιμοποιώντας απλώς το `rdf:type`, η SPARQL δεν μπορεί να αντιληφθεί την ύπαρξη των subclasses (χρησιμοποιώ αυτό το Version). Επομένως ή δεν υπάρχει κάποιος Μηχανισμός Συμπερασμού στο build που έχω εγκαταστήσει, ή απλά κάνω κάτι λάθος.

```
❯ ./bin/sparql --version
Jena:       VERSION: 2.7.4
Jena:       BUILD_DATE: 2012-10-20T17:03:29+0100
ARQ:        VERSION: 2.9.4
ARQ:        BUILD_DATE: 2012-10-20T17:03:29+0100
~/Dow/apache-jena-2.7.4
```

Για το φιλτράρισμα των τμημάτων που βρίσκονται στην Πάτρα θα μπορούσαμε χρησιμοποιώντας τα αποτελέσματα που έχουν αποθηκευτεί στο `?depart`, να τα φιλτάρουμε ως `dep_city = "Patras"`, και έτσι να κρατήσουμε μόνο αυτά που ζητάει η εκφώνηση. Παρόλα αυτά, σε οποιονδήποτε χειρισμό που έκανα ο οποίος αφορούσε το `?depart`, κατέληγα σε άδειους πίνακες στην SPARQL.

[2] Όλοι οι φοιτητές/καθηγητές είναι μέλη του Department001 / CEID.

4iv ΕΡΩΤΗΜΑ

```
SELECT ?name ?capacity
WHERE {
    ?x      rdf:type   uni:Classroom;
            uni:room_name   ?name;
            uni:room_capacity  ?capacity
            FILTER(?capacity > "100")
}
```

name	capacity
"G"	"200"
"BA"	"200"

Το `?x` περιλαμβάνει όλα τα instances των αιθουσών, το `?name` (τα ονόματα) και το `?capacity` (την χωρητικότητα). Με το `FILTER(?capacity > "100")` εμφανίζονται οι περιπτώσεις όπου η χωρητικότητα είναι άνω των 100.

Για το φιλτράρισμα των τμημάτων που βρίσκονται στην Πάτρα, πάλι χρειαζόμαστε Inference Engine.

5 ΕΡΩΤΗΜΑ

Δεν έχει πραγματοποιηθεί.