ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Παρουσίαση της SPARQL με χρήση του Jena Adapter για Oracle

Του φοιτητή

Πατσίκα Κωνσταντίνου

Αρ. Μητρώου: 04/2566

Επιβλέπων καθηγητής

Δρ. Ευκλείδης Κεραμόπουλος

Θεσσαλονίκη 2011

ΠΡΟΛΟΓΟΣ

Η διπλωματική εργασία με τίτλο "Παρουσίαση της SPARQL με χρήση του Jena Adapter για Oracle" έχει σκοπό να παρουσιάσει τις δυνατότητες που προσφέρει η γλώσσα SPARQL, μια γλώσσα επερωτήσεων για RDF η οποία αποτελεί πρότυπο του W3C, με την χρήση του Jena Adapter για αποθήκευση των δεδομένων σε βάση δεδομένων Oracle.

ΠΕΡΙΛΗΨΗ

Το RDF (*Resource Description Framework*), πρότυπο του W3C, είναι ένα μοντέλο δεδομένων για την αναπαράσταση πληροφορίας στον Παγκόσμιο Ιστό και αποτελεί τη θεμελίωση ενός συνόλου τεχνολογιών για τη μοντελοποίηση κατανεμημένης γνώσης στο Σημασιολογικό Ιστό. Ο στόχος του Σημασιολογικού Ιστού είναι η δημιουργία και επαναχρησιμοποίηση των δεδομένων. Η γλώσσα RDF είναι μια γλώσσα αναπαράστασης της πληροφορίας σχετικά με πόρους στον ιστό. Η παρούσα διπλωματική εργασία περιλαμβάνει τη μελέτη της τεχνολογίας RDF και της σημασιολογικής επέκτασης αυτής, του RDF Schema και αξιολογήθηκαν οι δυνατότητες που προσφέρει η γλώσσα SPARQL, μια γλώσσα επερωτήσεων για RDF η οποία αποτελεί πρότυπο του W3C, μέσω του Jena Adapter για Oracle. Στην παρούσα πτυχιακή αναλύουμε την γλώσσα SPARQL το συντακτικό με παραδείγματα για την χρήση της.

ABSTRACT

The RDF (Resource Description Framework), model of W3C, is a model data on the representation of information in the World Web and it constitutes the foundation of total of technologies for the modeling of distributed knowledge in the Semantic Web. The objective of Semantic Web is the creation and re-use of data. RDF is a language of representation of information with regard to resources in the web. The present thesis includes the study of technology RDF and this semantic extension, the RDF Schema and evaluated the possibilities that language SPARQL offers, a language of for RDF which constitutes model of W3C, via Jena Adapter for Oracle. In present thesis we analyze SPARQL language syntactic with examples for her use.

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία δεν θα είχε ολοκληρωθεί με την σημερινή της μορφή χωρίς την βοήθεια και την στήριξη κάποιων ανθρώπων που θα ήθελα να ευχαριστίσω θερμά.

Καταρχήν θέλω να εκφράσω τις θερμές μου ευχαριστίες στον καθηγητή Κύριο Ευκλίδη Κεραμόπουλο, επιβλέποντα της πτυχιακής εργασίας του οποίου οι συμβουλές και συμπαράσταση ήταν πολύτιμες καθ'όλη την ολοκλήρωση της εργασίας. Επίσης θα ήθελα να ευχαριστίσω την οικογένια μου για την υποστήριξη που μου έδειξε όλα αυτα τα χρόνια.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ **2**
ΠΕΡΙΛΗΨΗ **3**
ABSTRACT **4**
ΕΥΧΑΡΙΣΤΙΕΣ **5**
ΠΕΡΙΕΧΟΜΕΝΑ **6**
1. ΕΙΣΑΓΩΓΗ **10**
2. Το Πλαίσιο RDF **11**
2.1 Ιστορική αναδρομή 11
2.2 Ο ρόλος του RDF στο Σημασιολογικό Ιστό 13
2.3 Βασικά Δομικά Στοιχεία του RDF 15
2.3.1 URI 17
2.3.2 RDF/XML 19
2.4 Το RDF μοντέλο 19
2.4.1 Λεκτικά και Κενοί Κόμβοι 23
2.4.2 Υποστασιοποίηση 24
2.4.3 Άλλα Χαρακτηριστικά του RDF 25
2.4.3.1 Υποδοχές – RDF Containers 25
2.4.3.2 Συλλογές – RDF Collections 26
2.5 Μορφότυποι Αρχείων για Δεδομένα RDF 27
3. RDFSchema/Owl **33**
ΕΙΣΑΓΩΓΗ 33
3.1 RDF Schema 33
3.2 Δομή του RDF Schema 35
3.2.1 Κλάσεις 35
3.2.2 Ιδιότητες 36

3.3 OWL 39
3.3.1 Δομή της OWL 40
3.3.2 Υπογλώσσες OWL 41
3.3.2.1 OWL Lite 41
3.3.2.2 OWL DL 47
3.3.2.3 OWL FULL 47
3.3.3 Reasoning 50
4. Oracle 52
ΕΙΣΑΓΩΓΗ 52
4.1 Χαρακτηριστικά της Oracle 53
4.2 Βασικές Ικανότητες της Oracle 54
4.3 Διεπαφές της Oracle 56
4.4 Προετοιμασία Oracle και δημιουργία Semantic Network 57
5 Πλαίσια Διαχείρισης Εγγράφων RDF 59
ΕΙΣΑΓΩΓΗ 59
5.1 Jena 59
5.2 Sesame 61
5.3 Jena Adapter 63
6. Η ΓΛΩΣΣΑ SPARQL 64
6.1 ΕΙΣΑΓΩΓΗ 64
6.2 Χαρακτηριστικά της SPARQL 65
6.2.1 Μορφή Ερωτήσεων SPARQL 65
6.3 Σύνταξη SPARQL με Jena Adapter για Oracle 68
6.3.1 Εισαγωγή δεδομένων στην Oracle 68
6.3.2 Select επερώτηση 69
6.3.3 Select επερώτηση με OPTIONAL 70
6.3.4 Select επερώτηση με FILTER 71
6.3.5 Select επερώτηση με FILTER και OPTIONAL 72
6.3.6 Select επερώτηση με UNION 73
6.3.7 Select επερώτηση με UNION 2 74
6.3.8 Select επερώτηση με Order by 75
6.3.9 Select επερώτηση με DISTINCT 76
6.3.10 Select επερώτηση με LIMIT 77
6.3.11 Construct 78

6.3.12 Describe 79
6.3.13 Ask 80
6.3.14 Ask 2 81
ΕΠΙΛΟΓΟΣ 82
ΒΙΒΛΙΟΓΡΑΦΙΑ 83
ΠΑΡΑΡΤΗΜΑΤΑ 86
ΠΑΡΑΡΤΗΜΑ 1. Εισαγωγή δεδομένων στην Oracle με bulk loader 86
ΠΑΡΑΡΤΗΜΑ 2.Select επερώτηση 89
ΠΑΡΑΡΤΗΜΑ 3. Select επερώτηση με OPTIONAL 91
ΠΑΡΑΡΤΗΜΑ 4. Select επερώτηση με FILTER 93
ΠΑΡΑΡΤΗΜΑ 5. Select επερώτηση με FILTER και OPTIONAL 94
ΠΑΡΑΡΤΗΜΑ 6. Select επερώτηση με Union 96
ΠΑΡΑΡΤΗΜΑ 7. Select επερώτηση με Union 2 98
ΠΑΡΑΡΤΗΜΑ 8. Select επερώτηση με ORDER BY 100
ΠΑΡΑΡΤΗΜΑ 9. Select επερώτηση με DISTINCT 102
ΠΑΡΑΡΤΗΜΑ 10. Select επερώτηση με LIMIT 104
ΠΑΡΑΡΤΗΜΑ 11. Construct 105
ΠΑΡΑΡΤΗΜΑ 12. Describe 108
ΠΑΡΑΡΤΗΜΑ 13.ASK 110
ΠΑΡΑΡΤΗΜΑ 14.ASK 2 112
ΠΑΡΑΡΤΗΜΑ 15 Οντολογία Ολυμπιακών Αγώνων σε N-Triple 114

1. ΕΙΣΑΓΩΓΗ

Στόχος της παρούσας διπλωματικής εργασίας αποτέλεσε η μελέτη του RDF, του RDF Schema, OWL και η μελέτη της SPARQL, της πιο ευρέως χρησιμοποιούμενης γλώσσας επερωτήσεων για RDF, ώστε μέσα από αυτή να προβληθούν τα βασικότερα χαρακτηριστικά τους και να γίνει ευδιάκριτη η σημασία και η λειτουργικότητά τους μέσα στον Σημασιολογικό Ιστό. Παρουσιάζεται η θέση της μέσα στη στοίβα του Σημασιολογικού Ιστού και η αλληλεπίδρασή της με τις άλλες τεχνολογίες του πεδίου. Ωστόσο, οι δυνατότητες που κρύβονται πίσω από την τεχνολογία του RDF δε θα είχαν ιδιαίτερη αξία χωρίς την ύπαρξη μιας κατάλληλα δομημένης γλώσσας επερωτήσεων, που θα μπορεί να τα αξιοποιήσει στο μέγιστο βαθμό και να οδηγήσει στην ανάκτηση της επιθυμητής γνώσης.

Η ανάγκη για ευρεία διάδοση του RDF/S και του OWL, δεδομένου των απαιτήσεων για ύπαρξη αυστηρά δομημένης και κατανοητής από της μηχανές πληροφορίας στο πεδίο του Σημασιολογικού Ιστού, γίνεται ακόμα πιο εμφανής. Το γεγονός αυτό, σε συνδυασμό με την απουσία ενός κοινά αποδεκτού προτύπου από το W3C για την αποθήκευση και διαχείριση εγγράφων RDF, οδήγησε στην εμφάνιση διαφόρων νέων πλαισίων για την αξιοποίηση τέτοιου είδους δεδομένων αλλά και στη βελτιστοποίηση των ήδη υπαρχόντων.

Ένα ευρέως διαδεδομένο πλαίσιο υλοποιημένο σε Java με δυνατότητες αποθήκευσης, διαχείρισης και επερώτησης εγγράφων RDF, αλλά και υποστήριξη των RDF Schema και OWL, είναι το Jena όπου χρησιμοποιήθηκε για την παρούσα πτυχιακή εργασία.

2. Το Πλαίσιο RDF

Το Πλαίσιο Περιγραφής Πόρων είναι ένα πρότυπο της Κοινοπραξίας του Παγκόσμιου Ιστού. Αποτελεί το θεμέλιο αρκετών τεχνολογιών για τη μοντελοποίηση κατανεμημένης γνώσης και προορίζεται να χρησιμοποιηθεί σαν βάση για το Σημασιολογικό Ιστό. Ουσιαστικά πρόκειται για μια γλώσσα αναπαράστασης πληροφορίας για τους πόρους (*resources*) του Παγκόσμιου Ιστού. Κύριο στόχο του αποτελεί η αναπαράσταση των μεταδεδομένων

τους, ωστόσο, γενικεύοντας τον όρο «πόρος του Ιστού», το RDF μπορεί επιπρόσθετα να χρησιμοποιηθεί και για την αναπαράσταση πληροφορίας αντικειμένων που αναγνωρίζονται με μοναδικό τρόπο στον Ιστό, ακόμα και όταν εκείνα δεν μπορούν να ανακτηθούν άμεσα από αυτόν.

Το πλαίσιο RDF αναγνωρίζεται ως μια θεμελίωση για την επεξεργασία μεταδεδομένων και παρέχει διαλειτουργικότητα μεταξύ των εφαρμογών που ανταλλάσσουν πληροφορία κατανοητή για τις μηχανές. Αναμφισβήτητα, αποτελεί αναπόσπαστο κομμάτι του Σημασιολογικού Ιστού. Μεταξύ άλλων, η προσφορά του γίνεται αντιληπτή σε τομείς όπως οι μηχανές αναζήτησης, των οποίων βελτιώνει σημαντικά τις δυνατότητες, ενώ επίσης απαντάται στην περιγραφή καταλόγων θεματικών ιεραρχιών σε πύλες και ψηφιακές βιβλιοθήκες και υποστηρίζει την ανταλλαγή γνώσης μεταξύ πρακτόρων λογισμικού.

2.1 Ιστορική αναδρομή

Το RDF προτάθηκε για πρώτη φορά το 1995 από την Κοινοπραξία του Παγκόσμιου Ιστού (*W3C*) ως επέκταση της Πλατφόρμας PICS για επιλογή περιεχομένου στο διαδίκτυο (*Platform for Internet Content Selection - PICS*). Το

PICS αποτελούσε ένα μηχανισμό για την περιγραφή των περιεχομένων ιστοσελίδων και την αξιολόγηση τους.

Οι πληροφορίες ανταλλάσσονταν μεταξύ ενός Εξυπηρετητή Ιστού (*Web Server*) και των πελατών- φυλλομετρητών (*Web Browsers*) και αξιολογούνταν από τον εκάστοτε οργανισμό ή άτομο ανάλογα με τις τρέχουσες απαιτήσεις. Αυτή η δυνατότητα δινόταν στο σύστημα με κατάλληλη ρύθμιση των φυλλομετρητών. Η δημιουργία ενός τέτοιου μηχανισμού ανέκυψε από την ανάγκη για περιορισμό και αξιολόγηση της πληθώρας των περιεχομένων του Παγκόσμιου Ιστού.

Η αρχική ιδέα λοιπόν είχε γεννηθεί, και ήταν η ανταλλαγή πληροφοριών κατανοητών τόσο από τον άνθρωπο όσο και από τις μηχανές. Μέχρι εκείνη τη στιγμή, οι πληροφορίες που ανταλλάσσονταν στον Παγκόσμιο Ιστό ήταν ναι μεν κατανοητές από τον άνθρωπο, αλλά για τον υπολογιστή δεν είχαν κανένα νόημα, καθώς αυτός αποτελούσε απλά το μέσο για την μεταφορά και την παρουσίαση των πληροφοριών. Έτσι, δημιουργήθηκε μία νέα ομάδα εργασίας από το W3C, της οποίας τα μέλη συνεργάστηκαν για την δημιουργία ενός κοινώς αποδεκτού προτύπου για την περιγραφή πόρων στον Παγκόσμιο Ιστό.

Το εν λόγω πρότυπο δεν έμεινε ανεπηρέαστο από άλλες περιοχές και κοινότητες. Άντλησε στοιχεία τόσο από την περιοχή των ψηφιακών βιβλιοθηκών, όσο και από την περιοχή της αναπαράστασης γνώσεων. Δέχτηκε σημαντικές επιδράσεις από την περιοχή των δομημένων εγγράφων, από την SGML και ιδιαίτερα από την XML. Τέλος, καθώς η ομάδα περιλάμβανε μέλη από ομάδες άλλων προσπαθειών προτυποποίησης, όπως το Dublin Core, το Warwick Framework και την αρχιτεκτονική Metadata Content Framework , επηρεάστηκε σε σημαντικό βαθμό και από αυτές. Το Φεβρουάριο του 2004 η Κοινοπραξία του Παγκοσμίου Ιστού ανακοίνωσε την τελική έγκριση αυτής της κεντρικής τεχνολογίας

του Σημασιολογικού Ιστού, του αναθεωρημένου Πλαισίου Περιγραφής Πόρων, του RDF. Η αναθεώρηση του πλαισίου RDF και η καθιέρωσή του ως πρότυπο σηματοδοτεί την εξέλιξη του Σημασιολογικού Ιστού σε μια ευρεία πλατφόρμα δεδομένων στον Παγκόσμιο Ιστό. Παράλληλα, η χρήση του σε εμπορικά προϊόντα και υπηρεσίες σηματοδοτεί τη μετάβαση της τεχνολογίας του Σημασιολογικού Ιστού από ένα έργο έρευνας και προηγμένης ανάπτυξης τα τελευταία χρόνια, σε πιο πρακτική τεχνολογία, που χρησιμοποιείται σε μαζικά εργαλεία της αγοράς και που επιτρέπει πιο ευέλικτη πρόσβαση σε δομημένη πληροφορία στον Παγκόσμιο Ιστό.

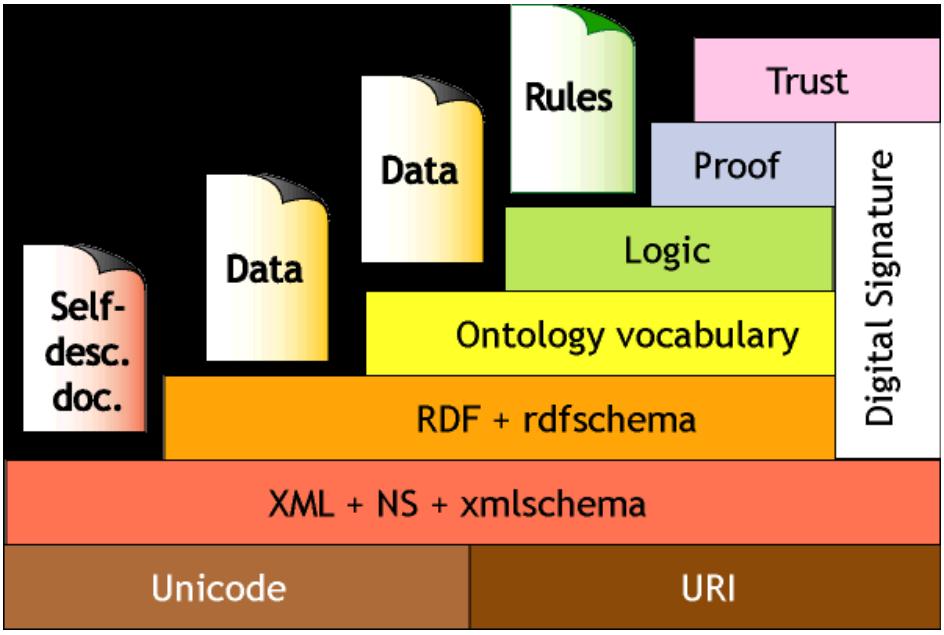
2.2 Ο ρόλος του RDF στο Σημασιολογικό Ιστό

Σύμφωνα με τον Eric Miller, επικεφαλή της δραστηριότητας του Σημασιολογικού Ιστού του W3C:

"*ο Σημασιολογικός Ιστός είναι περισσότερο Εξέλιξη του Παγκόσμιου Ιστού παρά Επανάσταση. Ο Σημασιολογικός Ιστός προκύπτει από μικρές αλλαγές, φέρνοντας περιγραφές που μπορούν να διαβάσουν μηχανές, στα δεδομένα και έγγραφα που ήδη υπάρχουν στον Παγκόσμιο Ιστό. Τα XML, RDF και OWL επιτρέπουν να γίνει ο Παγκόσμιος Ιστός μια παγκόσμια υποδομή για την κοινή χρήση εγγράφων και δεδομένων, που κάνει την αναζήτηση και επαναχρησιμοποίηση της πληροφορίας ευκολότερη αλλά και πιο αξιόπιστη.*"

Η Κοινοπραξία του W3C στηρίχθηκε σε μια άποψη του Berners-Lee, την επέκτεινε, και παρουσίασε την ιδέα της για τον μελλοντικό Ιστό μέσα από τη λεγόμενη Στοίβα του Σημασιολογικού Ιστού (*Semantic Web Stack*). Στο σχήμα που ακολουθεί (Εικόνα 1) παρουσιάζεται ο τρόπος που αναπτύσσεται αυτή η στοίβα. Πρόκειται για μια πυραμίδα από τεχνολογίες που φανερώνουν τις

εξαρτήσεις και τους στόχους των τεχνολογιών που εμπλέκονται στον Ιστό του μέλλοντος. Όπως προκύπτει, αναπόσπαστο κομμάτι μιας τέτοιας θεώρησης είναι και η τεχνολογία RDF.



Εικόνα 1 – Στοίβα του Σημασιολογικού Ιστού

Το κατώτερο στρώμα της στοίβας υποδεικνύει την ανάγκη για ένα κοινά αποδεκτό πρότυπο στην κωδικοποίηση κειμένου (*Unicode*) και τη δυνατότητα αναγνώρισης ενός πόρου με μοναδικό τρόπο (*URI*). Το επόμενο επίπεδο αποτελείται από τις τεχνολογίες XML, Name spaces (*NS*) και XML Schema. Το RDF και το RDF Schema, που χρησιμοποιούνται προκειμένου να προσδώσουν σημασιολογική έννοια στους πόρους, απαντώνται στο αμέσως πιο πάνω στρώμα. Όπως είναι προφανές, η παρουσία και η συνδεσιμότητα τους με τα ανώτερα επίπεδα της στοίβας του Σημασιολογικού Ιστού είναι ουσιαστική.

Στη δεξιά πλευρά του σχήματος παρουσιάζεται η Ψηφιακή Υπογραφή (*Digital Signature*), τεχνολογία που χρησιμοποιείται για την υπογραφή εγγράφων και κατά συνέπεια για την εκχώρηση αυτών στους δημιουργούς τους, ώστε κανείς να μην μπορεί να τα τροποποιήσει. Το Λεξιλόγιο Οντολογιών (*Ontology Vocabulary*) έχει ως στόχο την παροχή ενός κοινού λεξιλογίου καθώς και τρόπων για τη διενέργεια ελέγχων πιστότητας και απλών συμπερασμών, τεχνολογίες που όμως ακόμη δεν έχουν πλήρως υλοποιηθεί. Επόμενο βήμα στον μελλοντικό Ιστό αποτελεί η εφαρμογή Λογισμού (*Logic*) στα υπάρχοντα δεδομένα, ενώ με τη χρήση ενός συνόλου κανόνων (*Rules*) θα ήταν δυνατή η απόδειξη (P*roof*) νέων δηλώσεων και η επικύρωση συστημάτων. Η εξασφάλιση Εμπιστοσύνης (*Trust*) υποθέτει ότι όλες οι δηλώσεις στον Ιστό απαντώνται σε ένα περιβάλλον (*context*) το οποίο μπορεί να εκτιμηθεί ως προς την εγκυρότητά του. Ήδη, τα τρία ανώτερα επίπεδα αποτελούν θέμα συζήτησης και για ορισμένα από αυτά υπάρχει διαθέσιμη κάποια υποτυπώδης υλοποίηση.

2.3 Βασικά Δομικά Στοιχεία του RDF

Με τη βοήθεια του RDF μπορούμε να αναλύσουμε τη γνώση σε επιμέρους κομμάτια, τα οποία διέπονται από αναγκαίους κανόνες που αφορούν στη σημασιολογία τους, ή πιο απλά τη σημασία τους. Στόχος είναι να αποτελεί το RDF μια μέθοδο που να μπορεί, όχι μόνο να εκφράζει με απλό τρόπο όλα τα γεγονότα, αλλά παράλληλα να παρέχει σε αυτά και την κατάλληλη δόμηση ώστε να είναι δυνατή η μετέπειτα αξιοποίησή τους από τους ηλεκτρονικούς υπολογιστές. Και χρησιμοποιείται ο όρος «μέθοδος» αντί για «μορφότυπος», καθώς τα επιμέρους κομμάτια γνώσης είναι δυνατόν να γραφτούν με διαφορετικές εναλλακτικές και

παρόλα αυτά να συνεχίσουν να διατηρούν την αρχική τους δομή και πληροφορία, όπως ακριβώς μια έννοια μπορεί να εκφραστεί σε διαφορετικές ομιλούμενες γλώσσες ή μια δομή δεδομένων μπορεί να υλοποιηθεί με ποικίλους τρόπους.

Το RDF βασίζεται στην ιδέα ότι τα περιγραφόμενα αντικείμενα έχουν **ιδιότητες** (*properties*) με ορισμένες **τιμές** (*property values*), καθώς και στο ότι οι πόροι μπορούν να περιγραφούν φτιάχνοντας **δηλώσεις** (*statements*) οι οποίες προσδιορίζουν γι αυτούς τις ιδιότητες και τις αντίστοιχες τιμές τους. Το πλαίσιο RDF αναφέρεται στα επιμέρους τμήματα μιας δήλωσης με συγκεκριμένους όρους. Ειδικότερα, το τμήμα που προσδιορίζει τον πόρο για τον οποίο γίνεται η δήλωση καλείται **υποκείμενο** (*subject*). Το τμήμα που αναφέρεται σε μια ιδιότητα ή ένα χαρακτηριστικό του υποκειμένου είναι το **κατηγόρημα** (*predicate*) ενώ το κομμάτι που προσδιορίζει την τιμή αυτής της ιδιότητας είναι το **αντικείμενο** (*object*). Για παράδειγμα, αν θέλουμε να πούμε ότι πρωτεύουσα της Ελλάδας είναι η Αθήνα, θα μπορούσαμε να κάνουμε την εξής δήλωση: «Η Ελλάδα έχει πρωτεύουσα την Αθήνα». Στη δήλωση αυτή η «Ελλάδα» είναι το υποκείμενο, το «έχει πρωτεύουσα» το κατηγόρημα και η «Αθήνα» το αντικείμενο. Επειδή οι δηλώσεις αποτελούνται πάντα από τα τρία προαναφερθέντα τμήματα, καλούνται και **τριπλέτες** (*triples*).

Ωστόσο, ενώ η παραπάνω δήλωση είναι κατανοητή σε άτομα που μιλούν την ελληνική γλώσσα, στόχος του RDF είναι η δημιουργία δηλώσεων που να είναι κατανοητές και επεξεργάσιμες από τις μηχανές. Για την επίτευξη του σκοπού αυτού είναι αναγκαίες δύο προϋποθέσεις:

- η ύπαρξη ενός συστήματος αναγνωριστικών (*identifiers*) που θα προσδιορίζουν το υποκείμενο, το κατηγόρημα ή το αντικείμενο μιας

δήλωσης με μοναδικό τρόπο, εξαλείφοντας την πιθανότητα σύγχυσης μεταξύ παρόμοιων αναγνωριστικών στον Παγκόσμιο Ιστό.

- η ύπαρξη μιας κατανοητής και επεξεργάσιμης από τους υπολογιστές γλώσσας για την αναπαράσταση και την ανταλλαγή αυτών των δηλώσεων μεταξύ των μηχανών.

Η υπάρχουσα αρχιτεκτονική του Παγκόσμιου Ιστού ικανοποιεί και τις δύο παραπάνω απαιτήσεις. Συγκεκριμένα, η πρώτη ικανοποιείται μέσω ενός ήδη υπάρχοντος συστήματος αναγνωριστικών του Ιστού, του *Uniform Resource Identifier* ή *URI* ενώ η δεύτερη μέσω μιας γλώσσας που στηρίζεται στην XML και καλείται RDF/XML.

2.3.1 URI

Ο Ιστός παρέχει έναν ήδη γνωστό σε όλους μας τύπο αναγνωριστικών, το URL (*Uniform Resource Locator*). Το URL είναι μια συμβολοσειρά χαρακτήρων που προσδιορίζει με μοναδικό τρόπο έναν πόρο του Ιστού, αντιπροσωπεύοντας τον πρωταρχικό μηχανισμό προσπέλασής του, που ουσιαστικά είναι η δικτυακή του τοποθεσία. Όμως, το URL χρησιμοποιείται για Ιστοσελίδες και δεν καλύπτει περιπτώσεις αντικειμένων χωρίς διαθέσιμο δικτυακό τόπο.

Εντούτοις, ο Παγκόσμιος Ιστός παρέχει έναν ακόμα πιο γενικό τύπο αναγνωριστικών, το URI (*Uniform Resource Identifier*). Το URL αποτελεί μια ειδική μορφή του URI. Όλα τα URI χαρακτηρίζονται από τη δυνατότητα να δημιουργούνται ανεξάρτητα, από διαφορετικά άτομα ή οργανισμούς, οι οποίοι στη συνέχεια μπορούν να τα χρησιμοποιήσουν προκειμένου να αναγνωρίσουν με μοναδικό τρόπο διάφορα αντικείμενα. Συνεπώς, το URI δεν περιορίζεται μόνο στο να προσδιορίζει αντικείμενα που έχουν κάποια δικτυακή τοποθεσία ή που γενικά

προσπελαύνονται μέσω ενός οποιουδήποτε μηχανισμού στο διαδίκτυο. Ένα URI δημιουργείται προκειμένου να χαρακτηρίσει οτιδήποτε μπορεί να αναφερθεί μέσω μιας δήλωσης, συμπεριλαμβανομένων και των παρακάτω:

- Αντικείμενα προσβάσιμα μέσω του διαδικτύου, όπως για παράδειγμα ηλεκτρονικά έγγραφα, εικόνες, υπηρεσίες, είτε σύνολα από διάφορους άλλους πόρους.
- Αντικείμενα που δεν είναι προσβάσιμα μέσα από το διαδίκτυο, όπως άνθρωποι ή οργανισμοί.
- Αφηρημένες έννοιες που δεν υπάρχουν σαν φυσικές οντότητες, όπως η έννοια του «δημιουργού»

Λόγω αυτής της γενικότητας, το RDF χρησιμοποιεί τα URI σαν βάση του μηχανισμού που διαθέτει ώστε να αναγνωρίζει με μοναδικό τρόπο τα υποκείμενα, τα κατηγορήματα και τα αντικείμενα μέσα στις δηλώσεις. Πιο σωστά, το RDF χρησιμοποιεί όχι απλά URI αλλά αναφορές URI (*URI references - URIs*). Μια αναφορά URI (*URIref* ή *URIs*) είναι ένα URI, μαζί με ένα κομμάτι αναγνωριστικού στο τέλος. Για παράδειγμα, η αναφορά URI http://www.example.org/countries#Italy αποτελείται από το URI http://www.example.org/countries και το κομμάτι αναγνωριστικού Italy, όπου διαχωρίζεται με τον χαρακτήρα '#'. Στο RDF οι αναφορές URI μπορούν να περιέχουν χαρακτήρες UNICODE, δίνοντας έτσι σε πολλές γλώσσες τη δυνατότητα να χρησιμοποιούν τα URIrefs. Το RDF ορίζει έναν πόρο σαν οτιδήποτε μπορεί να γίνει αναγνωρίσιμο μέσω μιας αναφοράς URI. Επομένως η χρήση των URIrefs επιτρέπει πρακτικά στο RDF να περιγράφει οτιδήποτε, και να εκφράζει οποιαδήποτε σχέση μεταξύ αντικειμένων.

2.3.2 RDF/XML

Προκειμένου οι δηλώσεις να αναπαρίστανται με έναν κατανοητό για τους ηλεκτρονικούς υπολογιστές τρόπο, το πλαίσιο RDF χρησιμοποιεί την markup γλώσσα XML. Η XML σχεδιάστηκε με τρόπο, ώστε να επιτρέπει στον οποιονδήποτε να δημιουργεί τη δική του μορφή εγγράφων. Το RDF ορίζει μια συγκεκριμένη XML markup γλώσσα για την αναπαράσταση και την ανταλλαγή RDF πληροφορίας, γνωστή ως RDF/XML.

Όπως και η HTML, έτσι και η RDF/XML είναι επεξεργάσιμη από τις μηχανές ενώ με τη βοήθεια των URIs μπορεί να συνδέσει κομμάτια πληροφορίας στον Ιστό. Ωστόσο, σε αντίθεση με το κλασικό υπερκείμενο, οι αναφορές URI μπορούν να αναφερθούν σε οποιοδήποτε αντικείμενο έχει τη δυνατότητα αναγνώρισης, συμπεριλαμβανομένων και εκείνων που δεν μπορούν να ανακτηθούν άμεσα από τον Ιστό. Εκτενέστερη αναφορά στην RDF/XML γίνεται σε επόμενο κεφάλαιο.

2.4 Το RDF μοντέλο

Το RDF δημιουργεί δηλώσεις για να περιγράψει τους πόρους του Ιστού, χρησιμοποιεί αναφορές URI προκειμένου να αναγνωρίζονται με μοναδικό τρόπο τα αντικείμενα των RDF δηλώσεων, ενώ ως μέθοδο αναπαράστασης των τριπλετών RDF χρησιμοποιεί την RDF/XML. Συγκεκριμένα, το RDF στηρίζεται στην ιδέα ότι για κάθε πόρο εκφράζουμε απλές δηλώσεις, όπου η κάθε δήλωση αποτελείται από ένα υποκείμενο, ένα κατηγόρημα και ένα αντικείμενο. Επομένως, η απλή δήλωση:

Παράδειγμα 2.1:

Greece has a capital called Athens

θα μπορούσε να εκφραστεί σε RDF με τη χρήση URIs έχοντας ως:

***υποκείμενο*:** http://www.example.org/ccex#Greece

***κατηγόρημα*:** http://www.example.org/ccex/opt/hasCapital

***αντικείμενο*:** http://www.example.org/ccex#Athens

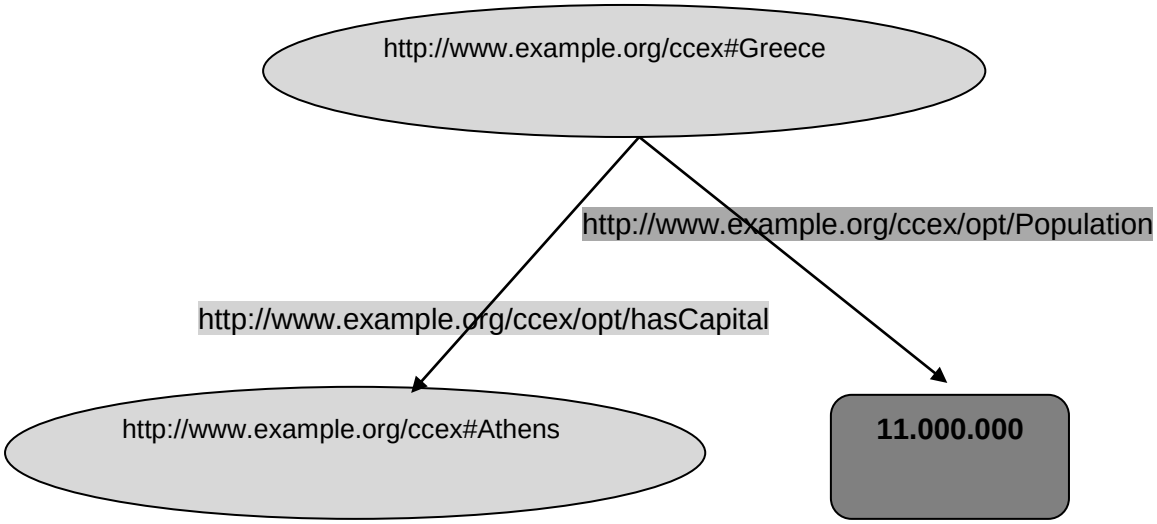
Η δυνατότητα του RDF να χρησιμοποιεί URIs για τον προσδιορισμό αντικειμένων και να περιγράφει πόρους με τη χρήση απλών ιδιοτήτων και των αντίστοιχων τιμών τους, του επιτρέπει να αναλύει και να αναπαριστά τις δηλώσεις γι αυτούς τους πόρους με τη μορφή γραφήματος. Συγκεκριμένα, το RDF μοντελοποιεί απλές δηλώσεις σε γράφημα με τρόπο, ώστε για κάθε υποκείμενο ή αντικείμενο να αντιστοιχίζεται ένας κόμβος του γραφήματος ενώ η εννοιολογική τους σύνδεση υλοποιείται δημιουργώντας ένα τόξο με φορά από τον κόμβο-υποκείμενο προς τον κόμβο-αντικείμενο. Ο κάθε κόμβος, ανάλογα με το αν αναπαριστά μια αναφορά URI ή μια σταθερά, έχει τη μορφή έλλειψης ή ορθογωνίου αντίστοιχα. Έτσι, αν επαυξήσουμε το αρχικό παράδειγμα, προσθέτοντας την παρακάτω δήλωση:

Παράδειγμα 2.2:

Greece has a population of 11,000,000

Τότε το συνολικό γράφημα που θα προκύψει, θα έχει τη μορφή που φαίνεται στο

ακόλουθο σχήμα (Εικόνα 2) :



Εικόνα 2 – Γράφημα RDF

Επειδή ο σχεδιασμός γραφημάτων για την αναπαράσταση δηλώσεων RDF δεν αποτελεί ιδιαίτερα πρακτική μέθοδο, χρησιμοποιείται μια εναλλακτική λύση, οι τριπλέτες. Σύμφωνα με τη θεώρηση των τριολέτων, κάθε δήλωση που συναντάμε σε ένα γράφημα αναπτύσσεται σε μια απλή τριάδα της μορφής: *υποκείμενο, κατηγόρημα, αντικείμενο* με αυτή ακριβώς τη σειρά. Για παράδειγμα, το γράφημα της Εικόνα 2 σειριοποιείται σε μορφή κειμένου, ως εξής:

Παράδειγμα 2.1:

<http://www.example.org/ccex#Greece> <http://www.example.org/ccex/opt/hasCapital> <http://www.example.org/gs/cities#Athens>.

Παράδειγμα 2.2:

<http://www.example.org/ccex#Greece> <http://www.example.org/ccex/opt/Population> “11,000,000”.

Μια τέτοια σειριοποίηση ενός γραφήματος RDF οδηγεί σε μεγάλες γραμμές κειμένου. Για την αποφυγή τους, οδηγούμαστε στη χρήση συντομεύσεων. Έτσι, μια πλήρης αναφορά URI αντικαθίσταται από ένα εξειδικευμένο όνομα XML (*XML qualified name* ή *QName*). Ένα QName αποτελείται από ένα πρόθεμα (*prefix*), που αντιστοιχίζεται σε ένα συγκεκριμένο URI, ακολουθούμενο από άνω κάτω τελεία και ένα τοπικό όνομα στο τέλος.

Αν στα παραπάνω παραδείγματα, χρησιμοποιηθούν τα παρακάτω προθέματα:

ccex: για το URI: http://www.example.org/ccex#

ccopt: για το URI: http://www.example.org/ccex/opt/

Τότε, το γράφημα της Εικόνα 2 μπορεί εκ νέου να σειριοποιηθεί και να αποκτήσει τη μορφή:

Παράδειγμα 2.1:

ccex:Greece ccopt:hasCapital ccex:Athens .

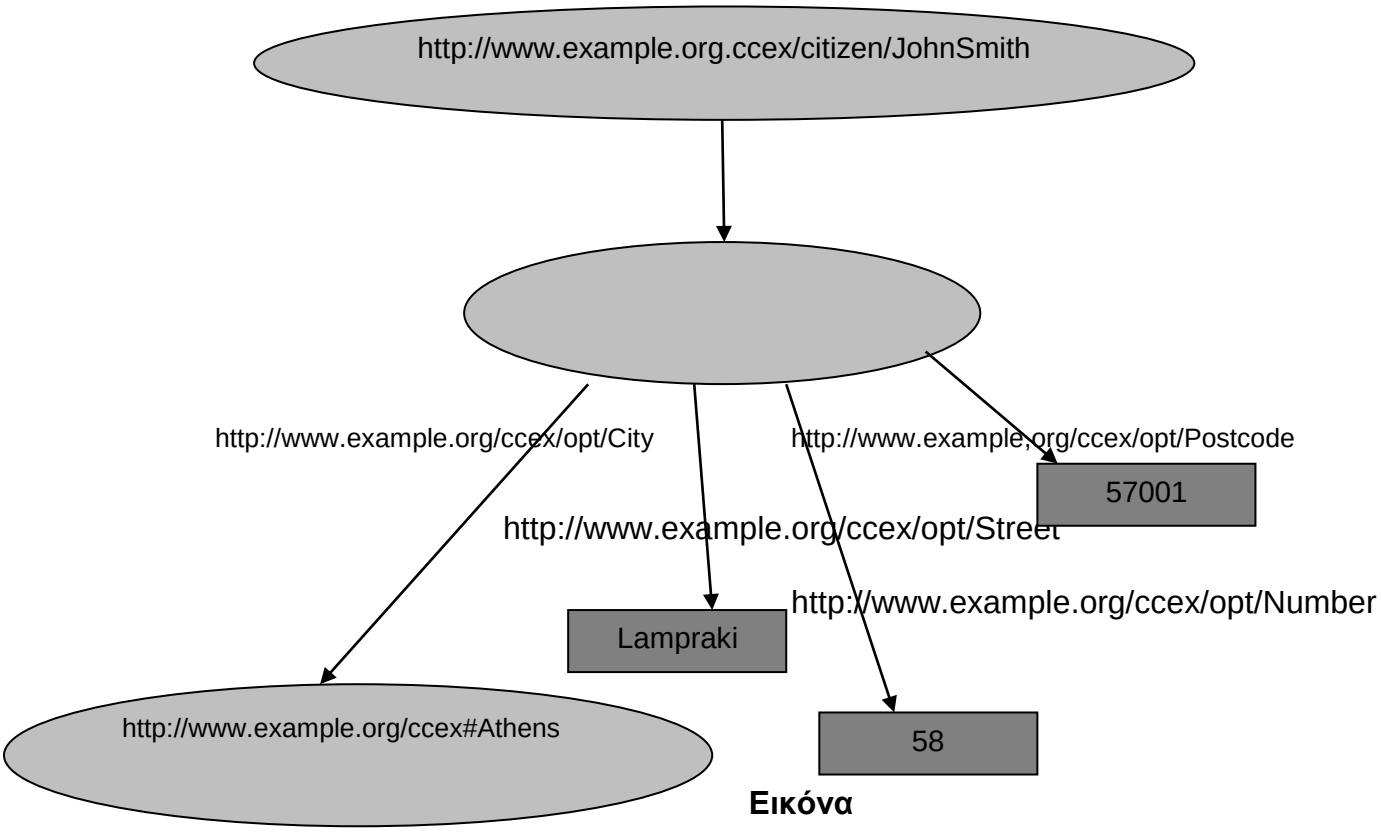
Παράδειγμα 2.2:

ccex:Greece ccopt:Population “11,000,000” .

Από τη στιγμή που το RDF για να κατονομάσει τα αντικείμενα στις δηλώσεις χρησιμοποιεί URIrefs και όχι απλές λέξεις, αναφερόμαστε σε ένα σύνολο τέτοιων URIrefs με τον όρο λεξιλόγιο (*Vocabulary*).

2.4.1 Λεκτικά και Κενοί Κόμβοι

Δύο βασικά στοιχεία του RDF είναι τα Λεκτικά (*Literals*) και οι Κενοί Κόμβοι (Anonymous or *Blank Nodes*). Τα λεκτικά μπορούν να υποκαταστήσουν το αντικείμενο μέσα σε μια τριπλέτα RDF και ουσιαστικά πρόκειται για ένα σύνολο χαρακτήρων, δηλαδή για κείμενο (*raw text*). Σε αντίθεση με τα URIs, τα οποία αντιστοιχίζονται σε αντικείμενα του πραγματικού κόσμου, οι τιμές των λεκτικών αποτελούν απλό, ακατέργαστο κείμενο δεδομένων που εισάγεται με αυτή τη μορφή στο γράφημα. Για παράδειγμα, τα λεκτικά μπορούν αν χρησιμοποιηθούν για να συνδέσουν ένα άτομο με το όνομα του ή ένα βιβλίο με τον αριθμό του ISBN. Οι κενοί κόμβοι είναι κόμβοι του γραφήματος που δεν έχουν κάποιο όνομα. Στους κόμβους αυτούς δεν εκχωρήθηκε ετικέτα είτε γιατί είναι άγνωστη, είτε γιατί δεν υπάρχει. Μια βασική χρήση των κενών κόμβων είναι στην αναπαράσταση αθροιστικών εννοιών, όπως για παράδειγμα η διεύθυνση ενός ατόμου. Σε μια τέτοια περίπτωση, αν θέλουμε να δημιουργήσουμε δηλώσεις διαχωρίζοντας τα επιμέρους στοιχεία της αθροιστικής έννοιας – όπως είναι για τη διεύθυνση η πόλη, η οδός, ο αριθμός και ο ταχυδρομικός κώδικας – θα μπορούσε να δημιουργηθεί μια κενή αναπαράσταση αυτής, δηλαδή ένας κενός κόμβος ο οποίος με τη σειρά του συνδέεται με όλα τα επιμέρους στοιχεία. Ένα τέτοιο παράδειγμα παρατίθεται στο αμέσως επόμενο σχήμα (Εικόνα 3). Η κενή έλλειψη αντιστοιχεί σε έναν κενό κόμβο ο οποίος εκφράζει τη διεύθυνση ενός συγκεκριμένου ατόμου.



Εικόνα

3 – Κενοί κόμβοι σε γράφημα RDF

2.4.2 Υποστασιοποίηση

Πολύ συχνά δημιουργείται η ανάγκη να εκφράσουμε μέσω του RDF δηλώσεις για άλλες δηλώσεις. Αυτό συμβαίνει στην περίπτωση που, για παράδειγμα, θέλουμε να καταγράψουμε κάποιες πληροφορίες για μια δήλωση, όπως το πρόσωπο που την έκανε ή τη χρονική στιγμή κατά την οποία έγινε ή όταν επιθυμούμε γενικότερα να εκφράσουμε κάποιες πεποιθήσεις. Για να διατυπώσουμε δηλώσεις σχετικά με άλλες δηλώσεις πρέπει να χτίσουμε ένα μοντέλο της αρχικής δήλωσης εκφράζοντας το σαν πόρο στον οποίο στη συνέχεια μπορούμε να επισυνάψουμε ιδιότητες. Η διαδικασία αυτή ονομάζεται **Υποστασιοποίηση** (**reification**).

Το RDF παρέχει ένα ενσωματωμένο λεξιλόγιο ειδικά για την περιγραφή δηλώσεων για δηλώσεις RDF, το οποίο αναφέρεται ως λεξιλόγιο υποστασιοποίησης. Αποτελείται από τον τύπο rdf:Statement και τις ιδιότητες rdf:subject, rdf:predicate και rdf:object. Να σημειωθεί ότι η αρχική δήλωση και η υποστασιοποίησή της αποτελούν δύο τελείως διαφορετικές οντότητες και καμιά δεν υπονοεί την άλλη. Για παράδειγμα, μέσα από την ακόλουθη δήλωση:

Παράδειγμα 2.3:

Encarta mentions that Greece has a population of 11,000,000

διατυπώνεται ένας ισχυρισμός που αποδίδεται σε μια συγκεκριμένη οντότητα.

2.4.3 Άλλα Χαρακτηριστικά του RDF

2.4.3.1 Υποδοχές – RDF Containers

Αρκετά συχνά, μέσω του RDF, χρειάζεται να περιγράψουμε ομάδες αντικειμένων, όπως οι συγγραφείς ενός βιβλίου όταν αυτοί είναι παραπάνω από ένας ή οι φοιτητές που παρακολουθούν ένα συγκεκριμένο μάθημα. Το RDF ορίζει ένα λεξιλόγιο γι αυτό το σκοπό, το λεξιλόγιο για υποδοχές, που αποτελείται από τρεις προκαθορισμένους τύπους. Ένα περίβλημα (*container*) είναι ένας πόρος που μέσα του εμπερικλείει κι άλλες οντότητες. Οι περιεχόμενες αυτές οντότητες καλούνται μέλη και μπορεί να είναι πόροι – συμπεριλαμβανομένων και των κενών κόμβων – ή λεκτικά. Το RDF ορίζει τρία είδη υποδοχών:

Bag: Είναι ένας πόρος τύπου rdf:Bag και αναπαριστά μια ομάδα από πόρους ή

λεκτικά στην οποία δεν παίζει ρόλο η σειρά των μελών, ενώ επιτρέπονται οι επαναλήψεις στοιχείων.

Sequence ή *Seq*: Είναι ένας πόρος τύπου rdf:Seq και αναπαριστά ένα σύνολο από πόρους ή λεκτικά στην οποία η σειρά των μελών έχει σημασία, ενώ επιτρέπονται οι επαναλήψεις στοιχείων.

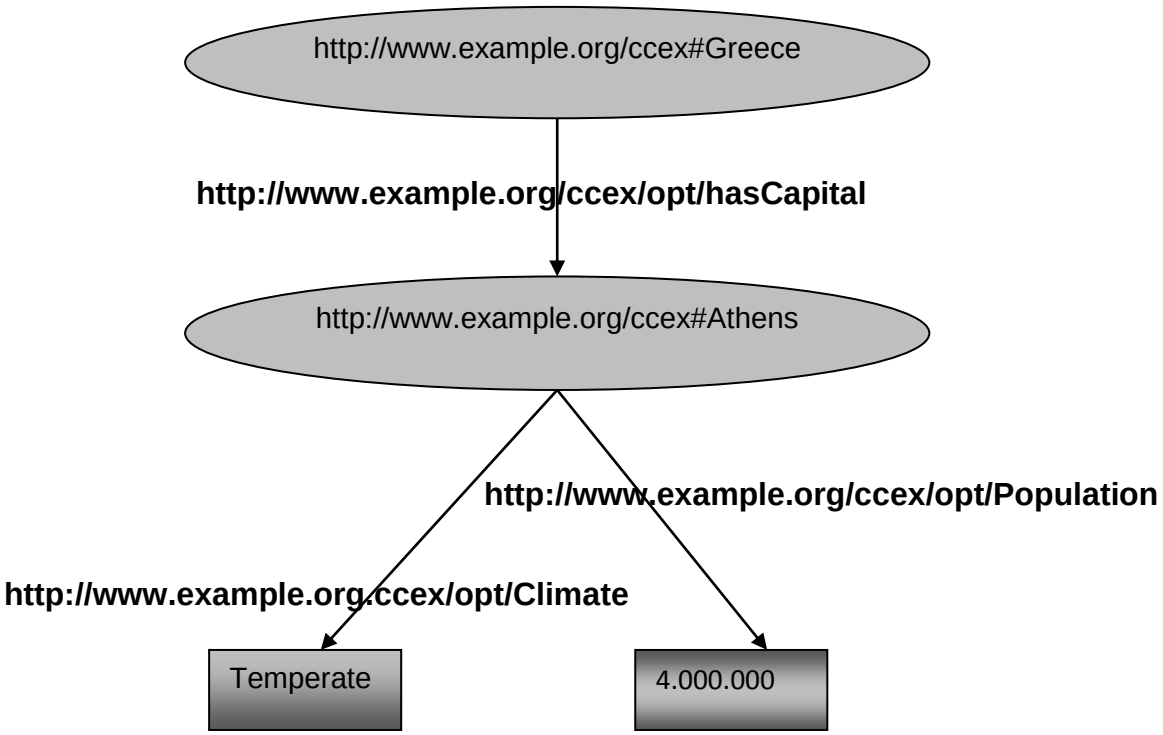
Alternative ή *Alt*: Είναι ένας πόρος τύπου rdf:Alt και αναπαριστά μια ομάδα από πόρους ή λεκτικά που αποτελούν εναλλακτικές ενός άλλου αντικειμένου (συνήθως μιας τιμής μιας ιδιότητας), όπως οι εναλλακτικοί τίτλοι μιας ταινίας.

2.4.3.2 Συλλογές – RDF Collections

Ένας περιορισμός που χαρακτηρίζει τις υποδοχές RDF είναι ότι δεν μπορούν να δηλώσουν πότε ένα σύνολο είναι πλήρες. Δηλαδή, παρόλο που έχουν τη δυνατότητα να αναγνωρίζουν πόρους ως μέλη, δεν μπορούν να επιβεβαιώσουν την ύπαρξη ή όχι άλλων μελών. Επίσης, ενώ ένα γράφημα ενδέχεται να περιγράφει ορισμένα από τα μέλη μιας υποδοχής, δεν υπάρχει τρόπος να αποκλειστεί η πιθανότητα να υπάρχει κάπου αλλού και άλλο γράφημα που να περιγράφει κι άλλα μέλη. Για το σκοπό αυτό υπάρχουν οι συλλογές. Μια συλλογή RDF είναι μια ομάδα από αντικείμενα που αναπαρίστανται ως δομή λίστας στο γράφο RDF. Αυτή η δομή κατασκευάζεται χρησιμοποιώντας τους όρους
ενός λεξιλογίου που αποτελείται από τον προκαθορισμένο τύπο rdf:List, τις ιδιότητες rdf:first και rdf:rest και τον προκαθορισμένο πόρο rdf:nil.

2.5 Μορφότυποι Αρχείων για Δεδομένα RDF

Το μοντέλο δεδομένων RDF παρέχει ένα αφηρημένο, εννοιολογικό πλαίσιο για τον ορισμό και τη χρήση μεταδεδομένων. Ωστόσο, για τη δημιουργία και την ανταλλαγή μεταδεδομένων είναι απαραίτητη η ύπαρξη μιας συγκεκριμένης σύνταξης και ως τέτοια φαίνεται να μπορεί να χρησιμοποιηθεί η XML. Επίσης, σύμφωνα με την προδιαγραφή του, το RDF απαιτεί και τη χρήση XML namespaces ώστε να συνδέει με ακρίβεια κάθε ιδιότητα (*property*) με το σχήμα που ορίζει αυτήν την ιδιότητα. Όσον αφορά στην RDF/XML, τα βασικότερα στοιχεία ενός τέτοιου εγγράφου, όπως αυτά προκύπτουν από την προδιαγραφή του, είναι ότι χαρακτηρίζεται από δύο τύπους κόμβων: τους κόμβους πόρων (*resource nodes*) και τους κόμβους ιδιοτήτων (*property nodes*). Οι κόμβοι πόρων αποτελούν τα υποκείμενα και τα αντικείμενα των δηλώσεων, ενώ συνήθως συνοδεύονται από το γνώρισμα rdf:about όπου φανερώνει το URI του πόρου που παρουσιάζουν.



Εικόνα 4-Αναπαράσταση κλίματος και πληθυσμού

Για το παραπάνω παράδειγμα (Εικόνα 4), η αναπαράστασή του γραφήματος σε RDF/XML παίρνει την εξής μορφή:

```
1.  <rdf:RDF
2.            xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.             xmlns:ccex="http://www.example.org/ccex#"
4.             xmlns:ccopt="http://www.example.org/ccex/opt/">
5.            <rdf:Description about="http://www.example.org/ccex#Greece">
6.           <ccopt:hasCapital rdf:resource="http://www.example.org/ccex#Athens"/>
7.    </rdf:Description>
8.    <rdf:Description about=" http://www.example.org/ccex#Athens">
9.       <ccopt:Climate>temperate</ccopt:Climate>
```

```
10.     <ccopt:Population>4,000,000</ccopt:Population>
11.       </rdf:Description>
12. </rdf:RDF>
```

Πίνακας 1 –Σειριακή σύνταξη RDF/XML

Στο παράδειγμα που αναφέραμε ο μόνο κόμβος πόρου που χρησιμοποιείται είναι ο rdf:Description, όπου στην πρώτη περίπτωση αναπαριστά οτιδήποτε υπονοείται από το URI http://www.example.org/ccex#Greece (που προφανώς πρόκειται για μια χώρα, την Ελλάδα) και στη δεύτερη, οτιδήποτε σχετίζεται με το URI http://www.example.org/ccex#Athens (δηλαδή την πόλη Αθήνα). Οι κόμβοι πόρων περιλαμβάνουν μόνο κόμβους ιδιοτήτων, που αναπαριστούν δηλώσεις. Οι κόμβοι ιδιοτήτων, με τη σειρά τους, περιέχουν τιμές λεκτικών (όπως είναι παραπάνω το temperate και το 4,000,000 – γρ. 9 και 10 αντίστοιχα), μια αναφορά σε έναν πόρο αντικείμενο χρησιμοποιώντας το γνώρισμα rdf:resource (γρ. 6), ή τέλος μπορεί να περιλαμβάνουν και έναν πλήρη κόμβο πόρου.

Η παραπάνω σύνταξη (Πίνακας 2), που αναπαριστά το γράφημα RDF της Εικόνα 4, αποτελεί μία από τις δύο μορφές που παρέχονται από την RDF/XML. Πρόκειται για τη σειριακή σύνταξη (*serialization syntax*) η οποία εκφράζει τις πλήρεις δυνατότητες του μοντέλου με έναν συστηματικό τρόπο. Όμως, υπάρχει και η συντετμημένη σύνταξη (*abbreviated syntax*) όπου παρέχει επιπλέον δομές και δίνουν τη δυνατότητα αναπαράστασης ενός υποσυνόλου του μοντέλου με έναν πιο συμπαγή τρόπο. Αφορά ιδιότητες οι οποίες δεν επαναλαμβάνονται εντός ενός

στοιχείου rdf:Description και μπορούν να γραφούν σαν γνωρίσματα της XML επισυναπτόμενα στο στοιχείο rdf:Description. Επιπρόσθετα, αφορά και εμφωλευμένα στοιχεία. Με τη συντετμημένη σύνταξη ο Πίνακας 1 μετασχηματίζεται ως εξής:

```
1. <rdf:RDF
2.          xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.          xmlns:ccex="http://www.example.org/ccex#"
4.          xmlns:ccopt="http://www.example.org/ccex/opt/">
5.          <rdf:Description about="http://www.example.org/ccex#Greece">
6.       <ccopt:hasCapital rdf:resource="http://www.example.org/ccex#Athens"
9.                              ccopt:Climate = "temperate"
10.                             ccopt:Population ="4,000,000" />
11.         </rdf:Description>
12. </rdf:RDF>
```

Πίνακας 2 – Συντετμημένη σύνταξη RDF/XML

Η **Notation 3** (ή N3) και η **Turtle** αποτελούν άλλες δύο μεθόδους για την αναπαράσταση γραφημάτων RDF με τη μορφή κειμένου, όπου παρουσιάζουν ελάχιστες διαφοροποιήσεις μεταξύ τους. Η διαφορά τους σε σχέση με την RDF/XML είναι ότι προσφέρουν μεγαλύτερη ευκολία στην διάκριση και ανάγνωση των τριπλετών. Το παράδειγμα του γραφήματος της Εικόνα 4 γράφεται στη N3 ως εξής:

```
@prefix ccex: <http://www.example.org/ccex#> .
@prefix ccopt: <http://www.example.org/ccex/opt/> .

ccex:Greece ccopt:hasCapital ccex:Athens .
ccex:Athens ccopt:Climate “temperate” .
ccex:Athnes ccopt:Population 4,000,000 .
```

Πίνακας 3 – Σύνταξη N3

Όπως φαίνεται, στην N3 και στην Turtle οι τριπλέτες γράφονται σαν μια ακολουθία υποκείμενο-κατηγόρημα-αντικείμενο. Πρώτο γράφεται το URI που παίζει το ρόλο του υποκειμένου, ακολουθεί το URI του κατηγορήματος και τρίτο έρχεται το αντικείμενο, ακολουθούμενο από μια τελεία που δηλώνει το τέλος της τριπλέτας. Όσον αφορά στα προθέματα, αυτά δηλώνονται στην αρχή, μετά το χαρακτηριστικό σύμβολο @prefix. Σε μια τέτοια περίπτωση, τα πλήρη URI ανακτώνται συνενώνοντας τα κομμάτια που τα απαρτίζουν και αφού γίνουν οι απαραίτητες αντικαταστάσεις των προθεμάτων. Η N3 επιτρέπει κάποιες επιπλέον συντομεύσεις. Συγκεκριμένα, αν γίνεται επανάληψη του ίδιου υποκειμένου και κατηγορήματος σε κάποιες τριπλέτες, μπορεί να γραφτούν η μία μετά την άλλη, χρησιμοποιώντας κόμμα για το διαχωρισμός τους και παραλείποντας τους κοινούς όρους. Αναλόγως, αν πρόκειται για τριπλέτες με κοινό υποκείμενο, διαχωρίζονται με το ελληνικό ερωτηματικό. Η **NTriples** μοιάζει κι αυτή με την N3, ωστόσο όπως και η Turtle διαθέτουν λιγότερα περίπλοκα στοιχεία σε σχέση με την N3. Στην ουσία, η NTriples αποτελεί υποσύνολο της N3, χωρίς όμως να διαθέτει τη δυνατότητα των συντετμημένων τύπων. Η Turtle αποτελεί επέκταση της NTriples φροντίζονται παράλληλα να κρατά τα πιο κατάλληλα και χρήσιμα χαρακτηριστικά

της N3. Το παράδειγμα της Εικόνα 4 γράφεται σε NTriples όπως φαίνεται παρακάτω:

```
<http://www.example.org/ccex#Greece>
<http://www.example.org/ccex/opt/hasCapital>
<http://www.example.org/ccex#Athens> .

<http://www.example.org/ccex#Athens>
<http://www.example.org/ccex/opt/Climate> "temperate" .

<http://www.example.org/ccex#Athens>
<http://www.example.org/ccex/opt/Population> "4,000,000" .
```

Πίνακας 4 – Σύνταξη NTriples

3. RDFSchema/Owl

ΕΙΣΑΓΩΓΗ

Το RDF, όπως αναφέραμε, χρησιμοποιείται για την περιγραφή των πόρων και δεν μπορεί να περιγράψει την σημασιολογία τους. Για αυτήν την δουλειά είναι το RDFS όπου περιέχει μηχανισμό ομαδοποίησης των πόρων και συσχέτιση μεταξύ τους.

Η OWL, έχοντας δημιουργηθεί πάνω στο RDF και RDFS, προσδιορίζει της σχέσεις και την ιεραρχία που μπορούν να εκφραστούν μεταξύ των πόρων.

3.1 RDF Schema

Η πρώτη έκδοση του RDF Schema ανακοινώθηκε το 1998 ενώ η τελική έκδοση τον Φεβρουάριο του 2004. Το RDF Schema στηρίζεται πάνω σε κλάσεις και ιδιότητες όπου γίνεται ο ορισμός των κλάσεων, η ιεραρχία των κλάσεων και η δημιουργία της κληρονομικότητας μέσω αυτών δημιουργούνται οι οντολογίες.

Με τη βοήθεια του RDFS, είναι δυνατός ο προσδιορισμός μηχανισμών καθορισμού κλάσεων πόρων, καθώς και ο περιορισμός των πιθανών συνδυασμών κλάσεων μεταξύ τους χρησιμοποιώντας κατάλληλες συσχετίσεις.

Ένα RDF Schema αποτελείται από τις δηλώσεις κλάσεων, γνωρισμάτων και των σχέσεων μεταξύ των κλάσεων. Όμοιοι πόροι είναι ομαδοποιημένοι κάτω από την ίδια κλάση. Με βάση τα παραπάνω, μπορούμε να διακρίνουμε τρία διαφορετικά επίπεδα αφαίρεσης στο μοντέλο δεδομένων RDF/S: Στο κατώτερο επίπεδο υπάρχουν οι ίδιοι οι πόροι (έγγραφα, δικτυακοί τόποι, πρόσωπα ή οτιδήποτε άλλο).

Το επόμενο επίπεδο αφαίρεσης είναι το επίπεδο δεδομένων, όπου γίνεται η περιγραφή των πληροφοριακών πόρων με τη χρήση λεξιλογίων, τα οποία περιγράφονται στο τελευταίο επίπεδο, το επίπεδο σχήματος. Το επίπεδο σχήματος είναι το επίπεδο αφαίρεσης όπου αναπτύσσονται RDF σχήματα για να διευκολύνουν τη σημασιολογική περιγραφή των πόρων. Σε αυτό το επίπεδο, οι κλάσεις αναπαριστούν αφηρημένες οντότητες και αναφέρονται συλλογικά σε σύνολα παρομοίων αντικειμένων. Για την γραφική απεικόνιση των RDF/S περιγραφών και σχημάτων χρησιμοποιείται ένα μοντέλο κατευθυνόμενων γράφων με ετικέτες τόσο στις ακμές όσο και στους κόμβους που μπορεί εύκολα να συνδυάσει πολλά διαφορετικά λεξιλόγια και να επεκταθεί προσθέτοντας απλώς περισσότερες ακμές. Οι κόμβοι αναπαριστούν αντικείμενα (πόρους ή κλάσεις) και οι ακμές συμβολίζουν σχέσεις μεταξύ των κόμβων (ιδιότητες). Ως κόμβους αναπαριστούμε επίσης και τύπους Literal, δηλαδή αλφαριθμητικά και άλλους βασικούς τύπους δεδομένων, όπως αυτοί ορίζονται στο πρότυπο XML schema. Οι κόμβοι συμβολίζονται ως ελλείψεις και οι ατομικές τιμές ως παραλληλόγραμμα. Οι ακμές μπορεί να είναι τριών ειδών: απόδοσης γνωρισμάτων (attributes), δημιουργίας στιγμιότυπων (instances) και υπαλληλίας. Οι ακμές απόδοσης γνωρισμάτων αναπαριστούν γνωρίσματα κόμβων και σχέσεις μεταξύ τους, ενώ οι ακμές υπαλληλίας χρησιμοποιούνται για να δηλώσουν σε επίπεδο σχήματος ότι ένας κόμβος (κλάση) ή ιδιότητα είναι υποκατηγορία ενός σημασιολογικά ευρύτερου κόμβου ή ιδιότητας αντίστοιχα. Τέλος, οι ακμές δημιουργίας στιγμιότυπων αποτελούν τη σύνδεση ανάμεσα στα πρότυπα RDF και RDFS, επιτρέποντας τη δημιουργία στιγμιότυπων μίας κλάσης και την απόδοση τύπων σε πληροφοριακούς πόρους που περιγράφονται.

3.2 Δομή του RDF Schema

Μία τριπλέτα RDFS αποτελείται από κλάσεις, ιδιότητες και τιμές. Έτσι οι μηχανές βάση στις κλάσεις και στις ιδιότητες είναι σε θέση να αποφασίσουν την σημασία τους και να προχωρήσουν σε επεξεργασία και εξαγωγή των συμπερασμάτων.

3.2.1 Κλάσεις

Οι πόροι είναι δυνατό να χωριστούν σε ομάδες που ονομάζονται **κλάσεις** (**classes**). Τα μέλη μια κλάσης ονομάζονται **στιγμιότυπα** (**instances**) της κλάσης. Οι κλάσεις είναι και οι ίδιες πόροι. Στην RDF υπάρχει διάκριση μεταξύ μιας κλάσης και των στιγμιότυπων της. Με κάθε κλάση συνδέεται ένα σύνολο το οποίο ονομάζεται **επέκταση της κλάσης** (**class extension**), το οποίο είναι το σύνολο των στιγμιότυπων της κλάσης. Δύο κλάσεις μπορεί να έχουν το ίδιο σύνολο στιγμιότυπων αλλά να είναι διαφορετικές κλάσεις. Μια κλάση μπορεί να είναι μέλος της επέκτασης της καθώς και στιγμιότυπο του εαυτού της. Η συλλογή των πόρων οι οποίοι είναι κλάσεις της RDF Schema είναι επίσης κλάση και ονομάζεται rdfs:class.

Αν μια κλάση C είναι υποκλάση μιας κλάσης B τότε όλα τα στιγμιότυπα της C είναι επίσης και στιγμιότυπα της B.

Η ιδιότητα rdfs:subClassOf μπορεί να χρησιμοποιηθεί για να δηλώσει ότι μια κλάση είναι υποκλάση μιας άλλης κλάσης. Ο όρος **υπερκλάση** (super-class) χρησιμοποιείται σαν ο ανάστροφος του όρου **υποκλάση**.

Αν μια κλάση B είναι *υπερκλάση* μιας κλάσης C

τότε όλα τα στιγμιότυπα της C είναι επίσης και στιγμιότυπα της B

Όλα τα αντικείμενα τα οποία περιγράφονται με RDF ονομάζονται **πόροι**, και είναι στιγμιότυπα της κλάσης rdfs:Resource. Αυτή η κλάση περιλαμβάνει τα πάντα. Όλες οι άλλες κλάσεις είναι **υποκλάσεις** της κλάσης αυτής.

Η rdfs:Resource είναι ένα στιγμιότυπο της rdfs:Class.

Η rdfs:Class είναι η κλάση των πόρων που είναι κλάσεις της RDF.Η rdfs:Class είναι στιγμιότυπο της rdfs:Class.

Η rdfs:Literal είναι η κλάση που περιλαμβάνει όλες τις τιμές με χαρακτήρες. Η rdfs:Literal είναι ένα στιγμιότυπο της rdfs:Class. Η rdfs:Literal είναι υποκλάση της rdfs:Resource.Η rdf:Property είναι η κλάση των ιδιοτήτων της RDF.

Η rdf:Property είναι στιγμιότυπο της rdfs:Class.

Η rdfs:Datatype είναι η κλάση όλων των τύπων δεδομένων. Η rdfs:Datatype είναι ταυτόχρονα και στιγμιότυπο και υποκλάση της rdfs:Class. Κάθε στιγμιότυπο της rdfs:Datatype είναι υποκλάση της rdfs:Literal.

3.2.2 Ιδιότητες

Η ιδιότητα **rdfs:subPropertyOf** μπορεί να χρησιμοποιηθεί για να δηλώσει ότι μια ιδιότητα είναι **υπο-ιδιότητα** (**sub-property**) μιας άλλης.

Αν μια ιδιότητα A είναι υπο-ιδιότητα μιας ιδιότητας B, τότε όλα τα ζεύγη πόρων τα οποία συσχετίζονται με την A συσχετίζονται επίσης με την B.

Ο όρος **υπερ-ιδιότητα** (**super-property**) χρησιμοποιείται συχνά σαν ο αντίστροφος του όρου υπο-ιδιότητα.

Αν μια ιδιότητα A είναι υπερ-ιδιότητα μιας ιδιότητας B, τότε όλα τα ζεύγη πόρων τα οποία συσχετίζονται με την B συσχετίζονται επίσης με την A.

Δεν έχει οριστεί μια ιδιότητα η οποία είναι υπερ-ιδιότητα όλων των ιδιοτήτων.

Η ιδιότητα **rdfs:range** είναι στιγμιότυπο της **rdf:Property** και χρησιμοποιείται για να δηλώσει ότι οι τιμές μιας ιδιότητας είναι στιγμιότυπα μιας ή περισσοτέρων κλάσεων.

Η δήλωση**: P rdfs:range C**

υποδηλώνει ότι το P είναι ένα στιγμιότυπο της κλάσης rdf:Property η C είναι ένα στιγμιότυπο της rdfs:class και οι πόροι που αντιστοιχούν στα αντικείμενα των τριάδων των οποίων το κατηγόρημα είναι το P είναι στιγμιότυπα της κλάσης C.

Η ιδιότητα **rdfs:domain** είναι στιγμιότυπο της **rdf:Property** και χρησιμοποιείται για να δηλώσει ότι κάθε πόρος που έχει μια συγκεκριμένη ιδιότητα είναι στιγμιότυπο μιας η **περισσοτέρων** κλάσεων.

Η δήλωση: **P rdfs:domain C**

υποδηλώνει ότι το P είναι ένα στιγμιότυπο της κλάσης rdf:Property
η C είναι ένα στιγμιότυπο της rdfs:class και οι πόροι που αντιστοιχούν στα

υποκείμενα των τριάδων των οποίων το κατηγόρημα είναι το P είναι στιγμιότυπα της κλάσης C.

Η ιδιότητα **rdf:type** είναι στιγμιότυπο της **rdf:Property** και χρησιμοποιείται για να δηλώσει ότι ένας πόρος είναι στιγμιότυπο μιας κλάσης.

Η δήλωση: **R rdfs:type C**

υποδηλώνει ότι το C είναι ένα στιγμιότυπο της rdfs:class και το R είναι στιγμιότυπο της C.

Η ιδιότητα **rdfs:subClassOf** είναι στιγμιότυπο **της rdf:Property** και χρησιμοποιείται για να δηλώσει ότι τα στιγμιότυπα μιας κλάσης είναι και στιγμιότυπα της άλλης.

Η ιδιότητα rdfs:subClassOf είναι μεταβατική.

Η ιδιότητα **rdfs:subPropertyOf** είναι στιγμιότυπο της **rdf:Property** και χρησιμοποιείται για να δηλώσει ότι όλοι οι πόροι που σχετίζονται με μια ιδιότητα σχετίζονται επίσης και με μια άλλη.

Η δήλωση: **P1 rdfs:subPropertyOf P2**

υποδηλώνει ότι τα P1 και P2 είναι στιγμιότυπα της rdf:Property, καθώς και ότι το P1 είναι υπο-ιδιότητα της P2.

Η ιδιότητα rdfs:subPropertyOf είναι μεταβατική.

Η ιδιότητα **rdfs:label** παρέχει μια εύκολα αναγνώσιμη από τον άνθρωπο παραλλαγή του ονόματος ενός πόρου.

Η δήλωση**: R rdfs:label L**

υποδηλώνει ότι το L μια εύκολα αναγνώσιμη από τον άνθρωπο ετικέτα για τον πόρο R

.

Η ιδιότητα **rdfs:comment** χρησιμοποιείται για να παρέχει μια αναγνώσιμη από τον άνθρωπο περιγραφή ενός πόρου. Ένα σχόλιο σε μορφή κειμένου διευκολύνει στο ξεκαθάρισμα της σημασίας των κλάσεων και των ιδιοτήτων της RDF.

3.3 OWL

ΕΙΣΑΓΩΓΗ

Ο όρος οντολογία προέρχεται από τη φιλοσοφία και αναφέρεται στην επιστήμη της περιγραφής των ειδών οντοτήτων στον κόσμο και πώς συσχετίζονται. Μια οντολογία καθορίζει τους όρους που χρησιμοποιούνται για να περιγράψουν και να αντιπροσωπεύσουν έναν τομέα της γνώσης. Οι οντολογίες χρησιμοποιούνται από τους ανθρώπους, τις βάσεις δεδομένων, και τις εφαρμογές που πρέπει να μοιραστούν τις πληροφορίες διαφόρων θεματικών περιοχών (domains). Οι οντολογίες περιλαμβάνουν ορισμούς βασικών εννοιών που αφορούν κάποια περιοχή και τις σχέσεις μεταξύ τους.

Ο όρος οντολογία έχει χρησιμοποιηθεί για να περιγράψει αντικείμενα με διαφορετικούς βαθμούς δομής. Αυτοί ποικίλουν από απλές ταξινομίες (όπως η ιεραρχία Yahoo), ως και θεωρίες λογικής. Οι οντολογίες αυτές πρέπει να διευκρινίζουν τις παρακάτω έννοιες:

- Τάξεις αντικειμένων (classes) στις διάφορες περιοχές ενδιαφέροντος
- Τις σχέσεις (relationships) που μπορούν να υπάρξουν μεταξύ των αντικειμένων
- Τις ιδιότητες (properties, attributes) που τα αντικείμενα μπορούν να έχουν.

3.3.1 Δομή της OWL

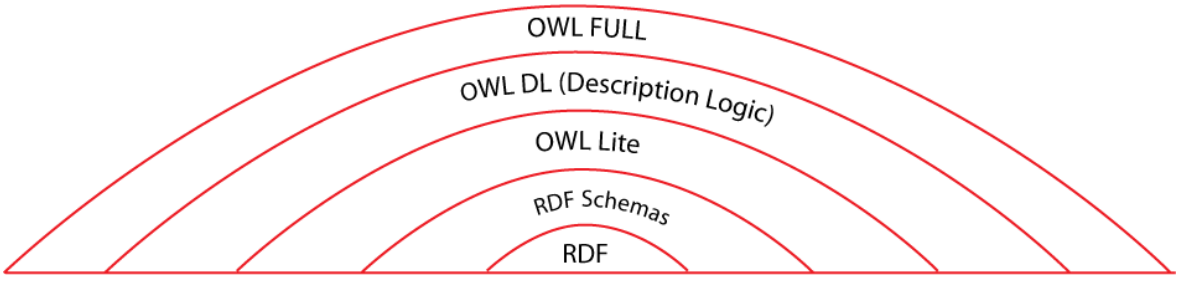
Η Γλώσσα Οντολογιών Ιστού OWL (Web Ontology Language) είναι μια σημασιολογική γλώσσα σήμανσης για την δημιουργία και τη διανομή οντολογιών στο διαδίκτυο. Έχει αναπτυχθεί από το Web Ontology Working Group ως τμήμα του W3C Semantic Web Activity και προορίζεται να χρησιμοποιηθεί σε περιπτώσεις όπου η πληροφορία που περιλαμβάνεται στα έγγραφα του Ιστού πρέπει να υποβληθεί σε επεξεργασία από εφαρμογές λογισμικού. Μπορεί να χρησιμοποιηθεί για να αναπαραστήσει τις έννοιες των όρων και των σχέσεων ανάμεσά τους. Αυτή η σημασιολογική αναπαράσταση των όρων και των αλληλεξαρτήσεών τους καλείται οντολογία. Η OWL παρέχει περισσότερες δυνατότητες κατά τον ορισμό οντολογιών από ό,τι η XML, η RDF, και το RDFS.

Η OWL αναπτύχθηκε ως επέκταση του λεξιλογίου της RDF (Resource Description Framework) και είναι απόγονος της γλώσσας οντολογιών DAML+OIL, ενσωματώνοντας έτσι λύσεις για προβλήματα που εμφανίστηκαν κατά τον σχεδιασμό και την εφαρμογή της DAML+OIL. Έχει πλεονεκτήματα που

προέρχονται από μια διαδικασία αυστηρότερης αναθεώρησης και τη σημαντική συμβολή στον σχεδιασμό της ερευνητών με βαθιά εμπειρία στην υλοποίηση μηχανών εξαγωγής συμπερασμάτων. Γενικά, συνιστάται στους υπεύθυνους ανάπτυξης λογισμικού που ασχολούνται με εφαρμογές του Σημασιολογικού Ιστού να χρησιμοποιήσουν το RDFS ή, κατά προτίμηση, την OWL ως την γλώσσα των οντολογιών τους. Ωστόσο ένα σημαντικό τμήμα των υπαρχουσών οντολογιών του Σημασιολογικού Ιστού χρησιμοποιεί σήμερα την DAML+OIL.

3.3.2 Υπογλώσσες OWL

Η OWL παρέχει τρεις υπογλώσσες, τις OWL Lite, DL και FULL, που σχεδιάστηκαν για να χρησιμοποιηθούν από διαφορετικές κοινότητες δημιουργών λογισμικού και χρηστών. Οι τρεις αυτές γλώσσες διαφέρουν ως προς την εκφραστικότητά τους και παρακάτω παρουσιάζονται στη σειρά, από την λιγότερο προς την περισσότερο εκφραστική. Κάθε μια από αυτές τις υπογλώσσες είναι μια επέκταση του απλούστερου προκατόχου της.



Εικόνα 5 - Τα επίπεδα της OWL

3.3.2.1 OWL Lite

Η OWL Lite υποστηρίζει εκείνους τους χρήστες που χρειάζονται πρώτιστα μια ταξινομία και απλούς περιορισμούς. Παραδείγματος χάριν, ενώ υποστηρίζει

περιορισμούς στον αριθμό των στοιχείων ενός συνόλου, επιτρέπει μόνο τις τιμές αριθμού στοιχείων 0 ή 1. Είναι απλή για να είναι δυνατή η εύκολη υποστήριξή της από εργαλεία και επιτρέπει σε θησαυρούς και σε άλλων ειδών ταξινομίες να μετασχηματιστούν εύκολα σε αυτήν.

Χαρακτηριστικά γνωρίσματα

- **Class**: Καθορίζει μια τάξη οντοτήτων που μοιράζονται μερικές κοινές ιδιότητες. Παραδείγματος χάριν, οι οντότητες Mary και ο Paul είναι και οι δύο μέλη της τάξης Person. Οι τάξεις μπορούν να οργανωθούν σε μια ιεραρχία ορίζοντας υποτάξεις με την χρήση του χαρακτηριστικού subClassOf. Υπάρχει μια ενσωματωμένη γενικότερη τάξη που ονομάζεται Thing και είναι υπερ- τάξη όλων των οντοτήτων της OWL. Υπάρχει επίσης μια ενσωματωμένη τάξη που ονομάζεται Nothing και είναι η τάξη στην οποία δεν ανήκει καμία οντότητα και δεν έχει καμία υπο-τάξη.
- **rdfs:subClassOf:** Οι ιεραρχίες των τάξεων μπορούν να δημιουργηθούν κάνοντας μια ή περισσότερες δηλώσεις που δείχνουν ότι μια τάξη είναι υπο-τάξη μιας ή περισσότερων άλλων τάξεων (υποστηρίζοντας πολλαπλή κληρονομικότητα). Παραδείγματος χάριν, η τάξη Person θα μπορούσε να δηλωθεί ως υπο-τάξη της τάξης Mammal. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν μια οντότητα ανήκει στην τάξη Person, τότε ανήκει επίσης στην τάξη Mammal.
- **rdf:Property:** Οι ιδιότητες μπορούν να χρησιμοποιηθούν για να δηλώσουν σχέσεις μεταξύ οντοτήτων ή χαρακτηριστικά οντοτήτων που παίρνουν τιμές

από συγκεκριμένους τύπους δεδομένων. Παραδείγματα ιδιοτήτων μπορεί να είναι οι hasChild, hasRelative, hasSibling, και hasAge.

- **rdfs:subPropertyOf:** Οι ιεραρχίες ιδιοτήτων μπορούν να δημιουργηθούν εισάγοντας μια ή περισσότερες δηλώσεις που δείχνουν ότι μια ιδιότητα είναι υποπερίπτωση μιας ή περισσότερων άλλων ιδιοτήτων. Παραδείγματος χάριν, η ιδιότητα hasSibling μπορεί να δηλωθεί ως subproperty της hasRelative. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν μια οντότητα συσχετίζεται με μια άλλη μέσω της ιδιότητας hasSibling, τότε συσχετίζεται επίσης μαζί της με την ιδιότητα hasRelative.
- **rdfs:domain:** Το πεδίο ορισμού μιας ιδιότητας περιορίζει τις οντότητες στις οποίες η ιδιότητα μπορεί να εφαρμοστεί. Εάν μια ιδιότητα συσχετίζει μια οντότητα με μια άλλη οντότητα και έχει οριστεί το πεδίο ορισμού της ιδιότητας, τότε η πρώτη οντότητα θα πρέπει να ανήκει στο πεδίο ορισμού. Παραδείγματος χάριν, η ιδιότητα hasChild μπορεί να δηλωθεί με πεδίο ορισμού την τάξη Mammal. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν Frank hasChild Anna, τότε ο Frank θα πρέπει να είναι Mammal. Σημειώστε ότι η rdfs:domain είναι ένας ολικός περιορισμός αφού δηλώνεται στην ιδιότητα συνολικά και όχι όταν αυτή σχετίζεται με κάποια συγκεκριμένη τάξη (περισσότερες πληροφορίες στην ενότητα που αναφέρεται στους περιορισμούς ιδιοτήτων).
- **rdfs:range:** Το πεδίο τιμών μιας ιδιότητας περιορίζει τις οντότητες που η ιδιότητα μπορεί να έχει ως τιμή της. Εάν μια ιδιότητα συσχετίζει μια οντότητα με μια άλλη οντότητα και έχει οριστεί το πεδίο τιμών της ιδιότητας, τότε η δεύτερη οντότητα θα πρέπει να ανήκει στο πεδίο τιμών.

Παραδείγματος χάριν, η ιδιότητα hasChild μπορεί να δηλωθεί με πεδίο τιμών την τάξη Mammal. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν η Louise συσχετίζεται με την Deborah βάση της ιδιότητας hasChild (δηλαδή η Deborah είναι το παιδί της Louise), τότε η Deborah είναι Mammal.

Ισότητες και Ανισότητες

- **equivalentClass:** Δύο τάξεις μπορούν να δηλωθούν ως ισοδύναμες. Οι ισοδύναμες τάξεις έχουν τις ίδιες οντότητες. Η ισοδυναμία μπορεί να χρησιμοποιηθεί για να δημιουργήσει συνώνυμες τάξεις. Παραδείγματος χάριν, η τάξη Car μπορεί να δηλωθεί ως equivalentClass της τάξης Automobile. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι οποιαδήποτε οντότητα που ανήκει στην τάξη Car είναι επίσης οντότητα της τάξης Automobile και αντίστροφα.
- **equivalentProperty:** Δύο ιδιότητες μπορούν να δηλωθούν ως ισοδύναμες. Η ισοδυναμία ιδιοτήτων μπορεί να χρησιμοποιηθεί για να δημιουργήσει συνώνυμες ιδιότητες. Παραδείγματος χάριν, η ιδιότητα hasLeader μπορεί να δηλωθεί ως equivalentProperty της ιδιότητας hasHead. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν το X συσχετίζεται με το Y με την ιδιότητα hasLeader, τότε το X συσχετίζεται επίσης με το Y με την ιδιότητα hasHead και αντίστροφα.
- **sameAs:** Δύο οντότητες μπορούν να δηλωθούν ως ίδιες. Παραδείγματος χάριν, η οντότητα Deborah μπορεί να δηλωθεί ως ίδια με την DeborahMcGuinness.

- **differentFrom:** Μια οντότητα μπορεί να δηλωθεί ως διαφορετική από άλλες οντότητες. Παραδείγματος χάριν, η οντότητα Frank μπορεί να δηλωθεί ως διαφορετική από τις οντότητες Deborah και Jim. Κατά συνέπεια, εάν μια ιδιότητα που δηλώνεται ως συναρτησιακή (functional) συσχετίζει την ίδια οντότητα με τις οντότητες Frank και Deborah, τότε θα υπάρχει αντίφαση.
- **AllDifferent:** Ένας αριθμό οντοτήτων μπορούν να δηλωθούν ως αμοιβαία διαφορετικές σε μια δήλωση AllDifferent. Παραδείγματος χάριν, ο Frank, η Deborah, και ο Jim θα μπορούσαν να δηλωθούν ως αμοιβαία διαφορετικές οντότητες χρησιμοποιώντας την δήλωση AllDifferent. Η δήλωση AllDifferent είναι ιδιαίτερα χρήσιμη όταν υπάρχουν σύνολα ευδιάκριτων αντικειμένων και όταν θέλουμε να εισάγουμε την υπόθεση της μοναδικότητας των ονομάτων μεταξύ των αντικειμένων του συνόλου.

Χαρακτηριστικά Ιδιοτήτων

- **inverseOf:** Μια ιδιότητα μπορεί να δηλωθεί ως αντίστροφη μιας άλλης ιδιότητας. Εάν αν η ιδιότητα P1 δηλωθεί ως αντίστροφη της ιδιότητας P2 και το X συσχετίζεται με το Y με την ιδιότητα P2, τότε το Y συσχετίζεται με το X με την ιδιότητα P1. Παραδείγματος χάριν, εάν η ιδιότητα hasChild είναι αντίστροφη της hasParent και Deborah hasParent Louise, τότε μια μηχανή μπορεί να συναγάγει ότι Louise hasChild Deborah.
- **TransitiveProperty:** Οι ιδιότητες μπορούν να δηλωθούν ως μεταβατικές. Παραδείγματος χάριν, εάν η ιδιότητα ancestor δηλωθεί ως μεταβατική, και εάν Sara ancestor Louise και Louise ancestor Deborah, τότε μια μηχανή μπορεί να συμπεράνει ότι Sara ancestor Deborah. Η OWL Lite (και η OWL DL) επιβάλλουν τον επιπλέον περιορισμό ότι οι μεταβατικές

ιδιότητες (και τα superproperties τους) δεν μπορούν να έχουν τον περιορισμό maxCardinality 1.

- **SymmetricProperty:** Οι ιδιότητες μπορούν να δηλωθούν ως συμμετρικές. Παραδείγματος χάριν, η ιδιότητα Friend μπορεί να δηλωθεί ως μια συμμετρική ιδιότητα. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι εάν Frank friend Deborah τότε και Deborah friend Frank.
- **FunctionalProperty:** Οι ιδιότητες μπορούν να δηλωθούν ώστε να έχουν μια μοναδική τιμή ανά οντότητα του πεδίου ορισμού. Αυτές οι ιδιότητες λέγονται συναρτησιακές. Εάν μια ιδιότητα είναι συναρτησιακή, επιτρέπεται να μην έχει τιμή για κάποια οντότητα του πεδίου ορισμού. Παραδείγματος χάριν, η ιδιότητα hasEmployer μπορεί να δηλωθεί ως FunctionalProperty. Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι καμία οντότητα δεν μπορεί να έχει περισσότερους από έναν εργοδότες. Ωστόσο αυτό δεν υπονοεί ότι κάθε οντότητα πρέπει να έχει τουλάχιστον έναν εργοδότη.
- **InverseFunctionalProperty:** Οι ιδιότητες μπορούν να δηλωθούν ως αντιστρόφως συναρτησιακές (inverse functional). Εάν μια ιδιότητα είναι αντιστρόφως συναρτησιακή, τότε η αντίστροφη της ιδιότητας αυτής είναι συναρτησιακή. Κατά συνέπεια η αντίστροφη της ιδιότητας έχει το πολύ μια τιμή για κάθε οντότητα. Αυτό το χαρακτηριστικό έχει αναφερθεί επίσης ως σαφής ιδιότητα. Παραδείγματος χάριν, η ιδιότητα hasUSSocialSecurityNumber (μοναδικό προσδιοριστικό για τους κατοίκους των ΗΠΑ) μπορεί να δηλωθεί ως αντιστρόφως συναρτησιακή (ή σαφής). Από το παραπάνω μια μηχανή μπορεί να συμπεράνει ότι δεν υπάρχουν δύο διαφορετικές οντότητες της τάξης Person που να έχουν τον ίδιο αριθμό αμερικάνικης κοινωνικής ασφάλισης. Επίσης, μια μηχανή μπορεί να

συναγάγει ότι εάν δύο οντότητες της τάξης Person έχουν τον ίδιο αριθμό κοινωνικής ασφάλισης, τότε οι δύο αυτές οντότητες θα πρέπει να είναι ίδιες.

3.3.2.2 OWL DL

Η OWL DL(OWL Description Logics), επεκτείνει την εκφραστικότητα της OWL Lite με λειτουργικότητα περιγραφικής λογικής και τύπους δεδομένων. Η OWL DL είναι το πιο εκφραστικό από τα είδη της OWL που εγγυώνται περατότητα και αποτελεσματικότητα της εξαγωγής συμπερασμάτων.

Τόσο η OWL DL όσο και η OWL Full χρησιμοποιούν το ίδιο λεξιλόγιο, το οποίο είναι υπερσύνολο εκείνου της OWL Lite, μόνο που η OWL DL έχει κάποιους επιπλέον περιορισμούς στη χρήση του λεξιλογίου αυτού. Η OWL DL υποστηρίζει εκείνους τους χρήστες που θέλουν τη μέγιστη εκφραστικότητα διατηρώντας την υπολογιστική πληρότητα (όλα τα ορθά συμπεράσματα είναι δυνατόν να παραχθούν).

3.3.2.3 OWL FULL

Η OWL Full απευθύνεται στους χρήστες που θέλουν τη μέγιστη εκφραστικότητα και τη συντακτική ελευθερία της RDF χωρίς τις υπολογιστικές εγγυήσεις. Η OWL Full επιτρέπει σε μια οντολογία να αυξήσει την έννοια του προκαθορισμένου (RDF ή OWL) λεξιλογίου. Είναι απίθανο για το λογισμικό να είναι σε θέση να υποστηρίξει την σημασιολογική επεξεργασία κάθε χαρακτηριστικού γνωρίσματος της OWL Full.

Κατά προσέγγιση, η βασική διαφορά μεταξύ OWL DL και OWL Full είναι ότι

η OWL DL απαιτεί τον διαχωρισμό των τύπων (μια τάξη δεν μπορεί να είναι ταυτόχρονα οντότητα ή ιδιότητα, μια ιδιότητα δεν μπορεί να είναι επίσης μια οντότητα ή μια τάξη). Επιπλέον, η OWL DL απαιτεί οι ιδιότητες να είναι είτε ObjectProperties είτε DatatypeProperties. Οι DatatypeProperties είναι σχέσεις οντοτήτων με Literals3 της RDFS και τύπους δεδομένων του XML σχήματος, ενώ οι ObjectProperties είναι σχέσεις μεταξύ οντοτήτων. Παρακάτω περιγράφονται οι επεκτάσεις που επιφέρουν στο λεξιλόγιο της OWL Lite οι OWL DL και OWL Full.

Ιδιότητες OWL DL και OWL FULL

- **oneOf:** *(*απαριθμημένες τάξεις): Οι τάξεις μπορούν να οριστούν από την απαρίθμηση των οντοτήτων που τις αποτελούν. Παραδείγματος χάριν, η τάξη daysOfTheWeek μπορεί να οριστεί αν απλά απαριθμηθούν οι οντότητες Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday. Από το παραπάνω μια μηχανή μπορεί να συναγάγει το μέγιστο αριθμό τιμών (7) οποιασδήποτε ιδιότητας που έχει περιορισμό allValuesFrom στην τάξη daysOfTheWeek.
- **hasValue:** (τιμές ιδιότητας): Μια ιδιότητα μπορεί να δηλωθεί να έχει μια ορισμένη οντότητα ως τιμή. Παραδείγματος χάριν, οι οντότητες της τάξης dutchCitizens μπορούν να οριστούν ως εκείνοι οι άνθρωποι που έχουν την οντότητα theNetherlands ως τιμή της υπηκοότητάς τους.
- **disjointWith:** Οι τάξεις μπορούν να δηλωθούν ως ξένες μεταξύ τους. Παραδείγματος χάριν, οι τάξεις Man και Woman μπορούν να δηλωθούν ως ξένες. Από αυτήν την δήλωση disjointWith, μια μηχανή μπορεί να εντοπίσει μια ασυνέπεια όταν μια οντότητα δηλωθεί να ανήκει και στις δύο τάξεις και

ομοίως μια μηχανή μπορεί να συμπεράνει ότι εάν το A είναι μια οντότητα της τάξης Man, τότε το A δεν μπορεί να είναι ταυτόχρονα μια οντότητα της τάξης Woman.

- **unionOf, complementOf, intersectionOf:** (πράξεις μεταξύ συνόλων): Η OWL DL και η OWL Full επιτρέπουν τον υπολογισμό ενώσεων, τομών και συμπληρωμάτων συνόλων. Παραδείγματος χάριν, χρησιμοποιώντας unionOf, μπορούμε να δηλώσουμε ότι μια τάξη περιέχει τα πράγματα που είναι είτε USCitizens είτε DutchCitizens. Χρησιμοποιώντας complementOf, θα μπορούσαμε να δηλώσουμε ότι τα παιδιά δεν είναι SeniorCitizens (δηλ. η τάξη Children είναι μια υποκατηγορία του συμπληρώματος της τάξης SeniorCitizens). Η τομή τάξεων για τις OWL DL και FULL δεν προϋποθέτει τους περιορισμούς που προαναφέραμε για την OWL Lite.

- **minCardinality, maxCardinality, cardinality** (πλήρης υποστήριξη του περιορισμού πληθάριθμου συνόλου): Ενώ στην OWL Lite οι ελάχιστοι ή μέγιστοι πληθάριθμοι περιορίζονται σε ακριβώς 1 ή 0, οι OWL DL και OWL Full επιτρέπουν πληθάριθμους με τιμές αυθαίρετους μη αρνητικούς ακέραιους αριθμούς. Παραδείγματος χάριν η τάξη DINKs ("Dual Income, No Kids") θα έθετε τον πληθάριθμο του συνόλου τιμών της ιδιότητας hasIncome σε ακριβώς 2, ενώ ο αντίστοιχος αριθμός της ιδιότητας hasChild θα έπρεπε να είναι 0.
- **σύνθετες τάξεις:** Σε πολλές δηλώσεις του λεξιλογίου της, η OWL Lite περιορίζει τη σύνταξη σε μοναδικά ονόματα τάξεων (π.χ. στις δηλώσεις subClassOf ή equivalentClass). Η OWL Full αίρει πλήρως αυτόν τον περιορισμό για να επιτρέψει αυθαίρετα σύνθετες περιγραφές τάξεων (η

OWL DL διατηρεί επίσης κάποιους περιορισμούς), που αποτελούνται από απαριθμημένες τάξεις, περιορισμούς ιδιοτήτων, και συνδυασμούς με ενώσεις, τομές και συμπληρώματα. Επίσης, η OWL Full επιτρέπει στις τάξεις να χρησιμοποιηθούν ως οντότητες (ενώ η OWL DL και η OWL Lite όχι).

3.3.3 Reasoning

Η διαθέσιμη πληροφορία αποκτά μεγαλύτερο ενδιαφέρον, εάν μέσω μιας κατάλληλης επεξεργασίας εξάγουμε από αυτή πολυδιάστατη και συνδυαστική γνώση, η οποία σε συνδυασμό με κάποιο ή κάποια μαθηματικά ή εννοιολογικά μοντέλα και κανόνες, δημιουργεί το φαινόμενο του συμπερασμού. Επομένως, η θεωρία του συμπερασμού (*Reasoning*) είναι μία πολύ σημαντική αλγοριθμική διαδικασία και αποτελεί το συνδετικό κρίκο μεταξύ της πληροφορίας και της γνώσης. Η εξαγωγή συμπερασμάτων (*Reasoning*) είναι ένα χαρακτηριστικό του RDFS το οποίο στηρίζεται μέσα από τους μηχανισμούς περιγραφής πόρων του Ιστού που αυτό διαθέτει, και κυρίως από τη δυνατότητα του να προσδιορίζει τις σχέσεις που υπάρχουν μεταξύ τους. Κατά συνέπεια, με τη βοήθεια των κλάσεων και των ιδιοτήτων που ορίζει το RDFS μπορούμε να οδηγηθούμε σε συμπεράσματα και να ανακτήσουμε γνώση που διαφορετικά θα ήταν αδύνατον να εξάγουμε. Στην εικόνα 6 φαίνεται μια τέτοια περίπτωση εξαγωγής συμπερασμών, όπου συνδυάζοντας τις πληροφορίες από δύο διαθέσιμες τριπλέτες οδηγούμαστε σε μια συμπερασματική δήλωση:

ccex:France ccopt:Population “62,000,000” .

ccopt:Population rdfs:domain ccopt:Country .

Συμπέρασμα

ccex:France rdfs:subClassOf ccopt:Country

Εικόνα 6-Εξαγωγή συμπερασμάτων (reasoning)

4. Oracle

ΕΙΣΑΓΩΓΗ

Οι σημασιολογικές τεχνολογίες βάσεων δεδομένων της Oracle 11g είναι μια ανοικτή, πρότυπη, εξελικτική, ασφαλής, αξιόπιστη και αποδοτική διοικητική RDF πλατφόρμα. Βασισμένοι σε ένα πρότυπο στοιχείων γραφικών παραστάσεων, τα στοιχεία RDF (τριπλέτες), συντάσσονται και ρωτούνται, όπως άλλους αντικειμενοστραφείς τύπους στοιχείων. Οι υπεύθυνοι για την ανάπτυξη εφαρμογής χρησιμοποιούν τη δύναμη της βάσης δεδομένων της Oracle 11g για να σχεδιάσουν και να αναπτύξουν ένα ευρύ φάσμα των σημασιολογικών επιχειρηματικών εφαρμογών στις περιοχές που περιλαμβάνουν τη νοημοσύνη, την επιβολή νόμων, την ενσωματωμένη βίο-πληροφορική και την πληροφορική υγειονομικής περίθαλψης, τη χρηματοδότηση, το κοινωνικό δίκτυο, τα παιχνίδια, και τη διαχείριση περιεχομένου.

4.1 Χαρακτηριστικά της Oracle

Παρακάτω παρουσιάζονται τα βασικά χαρακτηριστικά της Oracle 11g με ενεργοποιημένα τα Semantic Technologies για την Oracle

- Μεγάλο εύρος αποθήκευσης δεδομένων με πραγματικές σχέσεις που ενώνονται με την άλγεβρα Μπουλ για να επιτύχουν πιο σημασιολογικά και πλήρη αποτελέσματα επερώτησης.
- Επιτρέπει την μηχανή-οδηγημένη(machine-driven) ανακάλυψη νέων σχέσεων χρησιμοποιώντας την εγγενή μηχανή συμπεράσματος βάσεων δεδομένων της Oracle, τις οντολογίες, και τη σημασιολογία RDFS /SKOS/OWL και τους καθορισμένους από το χρήστη κανόνες.
- Σχέδια επερωτήσεων που χρησιμοποιούν είτε την άμεση πρόσβαση SPARQL στη βάση δεδομένων της Oracle 11g μέσω της Jena, του Sesame είτε του Joseki, είτε του SQL με τις SPARQL-ομοειδείς προτάσεις.
- Υποστηρίζει W3C πρότυπα για την αντιπροσώπευση, το λεξιλόγιο, και τη υλοποίηση των σχέσεων, συμπεριλαμβανομένου RDF, SKOS, RDFS, OWL, και SPARQL.
- Υποστηρίζει τις βασικές τεχνολογίες ανοιχτού κώδικα, συμπεριλαμβανομένης της Jena, Joseki, ARQ, Sesame, Pellet, D2RQ, Jetty, Cytoscape, Gate, και Protégé.
- Εξασφαλίζει ότι τα RDF στοιχεία στη γραφική παράσταση, οι τριπλέτες και οι ιδιότητες περιέχουν πολλαπλή στάθμη ασφάλειας EAL4+ και τον υποχρεωτικό έλεγχο προσπέλασης.

- Η σημασιολογική εύρεση βρίσκει τα έγγραφα αποθηκευμένα στη βάση δεδομένων της Oracle και λειτουργεί με τις υπηρεσίες εξαγωγής οντοτήτων τριών συμβαλλόμενων μερών, όπως OpenCalais και Gate.

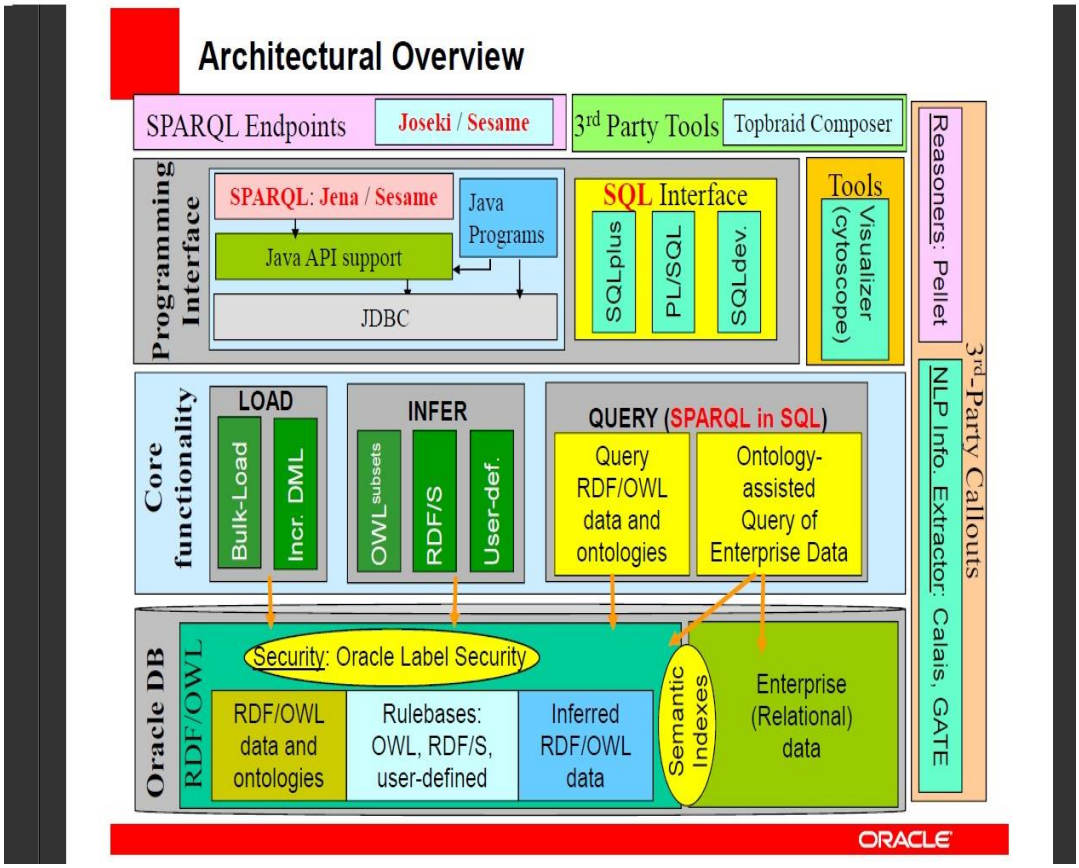
- Απόδοση και ελεξιμότητα:
    1. Τα συμπεράσματα, συμπεριλαμβανομένων των επαυξητικών και παράλληλων τρόπων, που γίνονται πριν από τις επερωτήσεις, μεγιστοποιούν την απόδοση.
    2. Οι επερωτήσεις RDF ωφελούνται από την σελίδα-ισόπεδη συμπίεση(page-level compression) μειώνει το I/O, της Oracle για να μεγιστοποιήσει την απόδοση.

4.2 Βασικές Ικανότητες της Oracle

Η Oracle 11g είναι το κορυφάιο πρόγραμμα, αποθήκευσης και επεξεργασίας δεδομένων , με αποθήκευση των δεδομένων σε τοπική βάση δεδομένων. Παρακάτω μπορούμε να δούμε τις βασικές ικανότητες της Oracle 11g.

1. Φόρτωση και Αποθήκευση:
    - Τοπική αποθήκευση στοιχείων γραφικών παραστάσεων RDF
    - Διαχειρίζεται δισεκατομμύρια τριπλετών
    - Βελτιστοποιημένη αρχιτεκτονική αποθήκευσης
2. Επερώτηση
    - Επερώτηση μέσω SPARQL-Jena/Joseki, Sesame
    - Επερώτηση μέσω SQL/graph, ευρετήριο b-tree

3. Εξαγωγή Συμπερασμάτων
    - RDFS, OWL2 RL, EL+, SKOS
    - Επαυξητικός, παράλληλος συλλογισμός
    - Ενσωματωμένη Αρχιτεκτονική



Εικόνα 7-Αρχιτεκτονική απεικόνιση επερωτήσεων στην Oracle

Στην εικόνα 7 μπορούμε να δούμε την αρχιτεκτονική απεικόνιση της Oracle. Στο υψηλότερο επίπεδο βλέπουμε της διεπαφές οπου ο προγραμματιστής έχει αλληλεπίδραση με το πρόγραμμα. Στο επόμενο επίπεδο έχουμε τις διάφορες λειτουργίες που εκτελούνται απο το jena adapter. Στο τελευταιο επίπεδο έχουμε

την βάση δεδομένων της Oracle με ενεργοποιημένες τις τεχνολογίες Spatial και Partitioning.Τέλος, δεξιά έχουμε τα προγράμματα που μπορούν να εγκατασταθούν ξεχωριστά απο την Oracle αλλά δουλεύουν μαζί με αυτήν.

4.3 Διεπαφές της Oracle

Όπως είδαμε στην εικόνα 7 στο πρώτο επίπεδο υπάρχουν οι διαπαφές και αυτές παρουσιάζονται παρακάτω.

- SQL και PL/SQL

Η **SQL** είναι μία γλώσσα υπολογιστών στις βάσεις δεδομένων, που σχεδιάστηκε για τη διαχείριση δεδομένων, σε ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων . Η **PL/SQL** (Procedural Language/SQL) επεκτείνει την SQL προσθέτοντας δομές που βρίσκονται συνήθως σε διαδικαστικές γλώσσες προγραμματισμού (Pascal, PL/I)

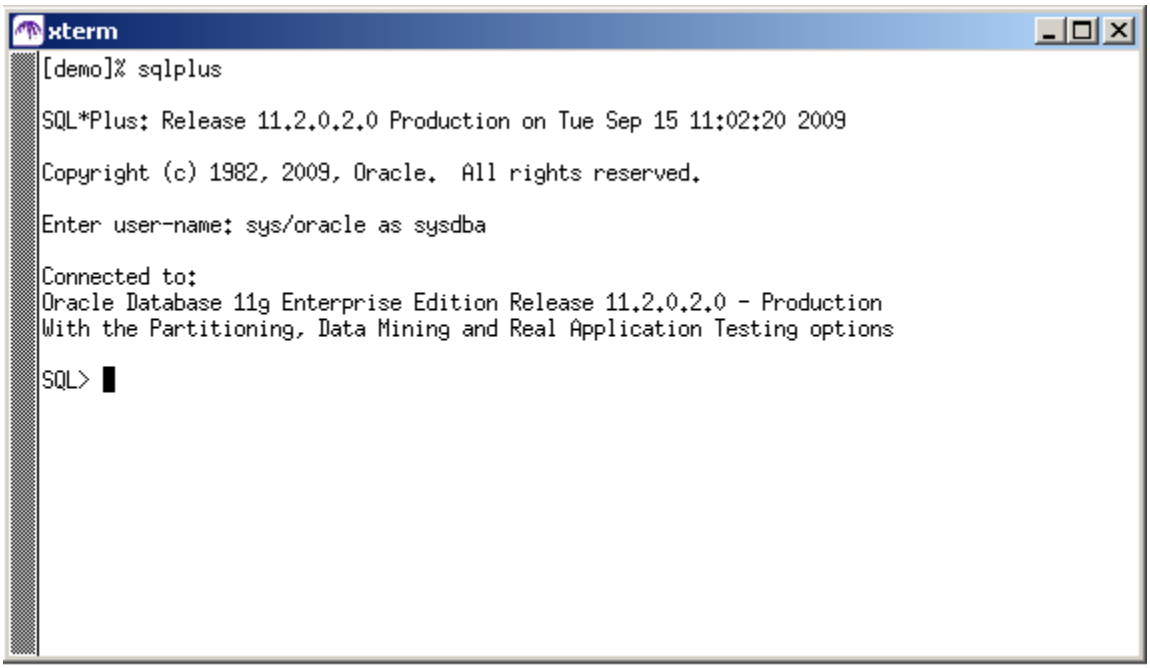
- Java-based
  - Jena (Jena Adapter για Oracle)
  - Sesame (Sesame Adapter για Oracle)
- SPARQL Endpoints
  - Joseki
  - OpenRDF Workbench

Το Joseki και το OpenRDFWorkbench είναι SPARQL Endpoint οπου μπορείς να εκτελέσεις SPARQL ερωτήματα μέσω ενός HTTP Endpoint.

4.4 Προετοιμασία Oracle και δημιουργία Semantic Network.

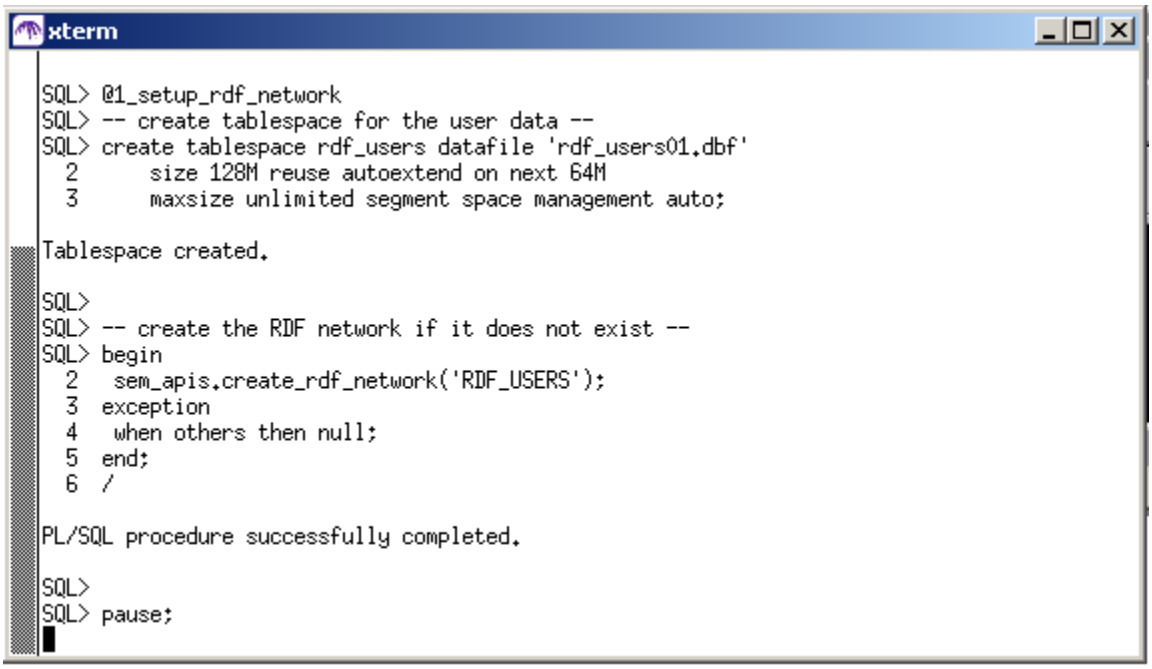
1.Ξεκινήστε το SQL*Plus και εισάγεται το όνομα χρήστη με τα κατάλληλα δικαιώματα.

sys/oracle as sysdba



2.Δημιουργία tablespace και semantic network με της ακόλουθες εντολές.

```
create tablespace rdf_users datafile 'rdf_users01.dbf'
size 128M reuse autoextend on next 64M
maxsize unlimited segment space management auto;

exec sem_apis.create_rdf_network('RDF_USERS');
```



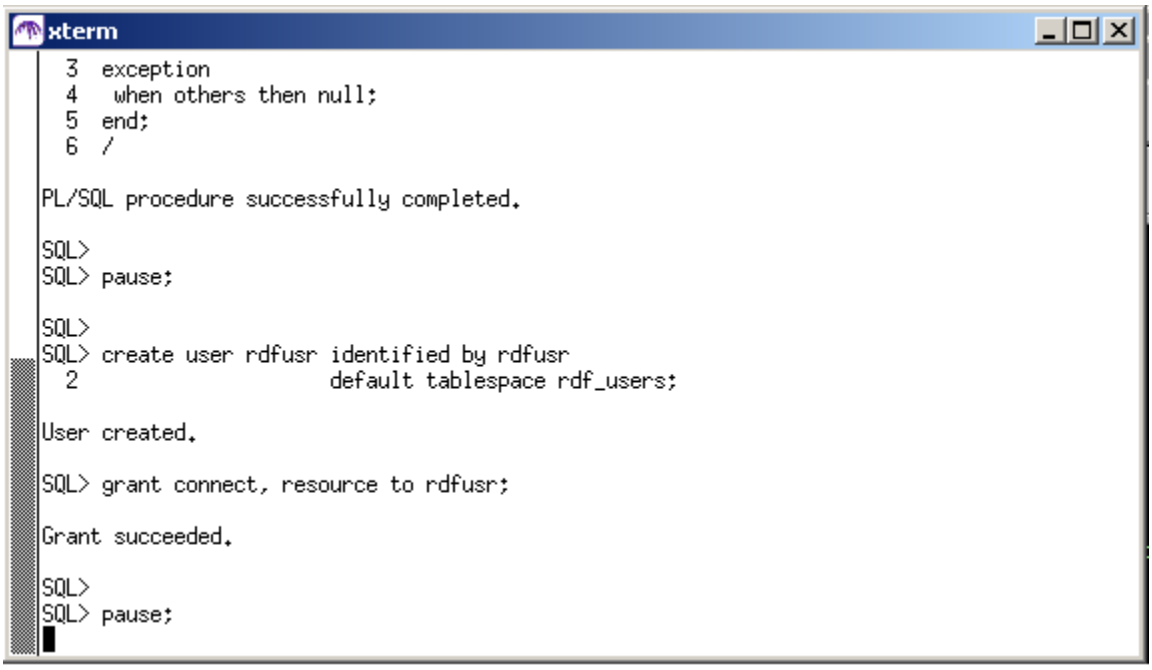
3.Δημιουργία Χρήστη rdf_user.

create user rdfusr identified by rdfusr

default tablespace rdf_users;

4. Χορήγηση δικαιωμάτων στον χρήστη.

grant connect, resource to rdfusr;



5 Πλαίσια Διαχείρισης Εγγράφων RDF

ΕΙΣΑΓΩΓΗ

Το γεγονός ότι το W3C δεν παρέχει κάποια προδιαγραφή για τη διαχείριση RDF, οδήγησε στην ανάπτυξη πολλών τέτοιων πλαισίων, σε διάφορες γλώσσες προγραμματισμού. Η Java φαίνεται να είναι η πιο δημοφιλής γλώσσα για την υλοποίηση εφαρμογών στο Σημασιολογικό Ιστό και έτσι στις μέρες μας διατίθενται πλέον αρκετά συστήματα αποθήκευσης και διατήρησης εγγράφων RDF υλοποιημένα σε αυτή τη γλώσσα. Στην παρούσα εργασία γίνεται αναφορά στις τρεις πιο ευρέως χρησιμοποιούμενες: στο Sesame, στο Jena, και στο Jena Adapter. Στο τέλος ακολουθεί συγκριτική αξιολόγηση των εργαλείων.

5.1 Jena

Η Jena δημιουργήθηκε και αναπτύχθηκε από την the HP Labs Semantic Web Program της Hewlett-Packard και είναι ένα από τα ποιο ευρέως πλαίσια διαχείρισης εγγράφων. Όπως αναφέρεται και στον δικτυακό τόπο της Jena " Το Jena είναι ένα ολοκληρωμένο εργαλείο για την ανάπτυξη εφαρμογών στο Σημασιολογικό Ιστό. Παρέχει ένα προγραμματιστικό περιβάλλον για το RDF, το RDF Schema, την OWL και τη γλώσσα επερωτήσεων SPARQL ."

Η δεύτερη έκδοση του Jena είναι αυτή στην οποία θα αναφερθούμε, καθώς αποτελεί βελτιωμένη έκδοση της πρώτης και συμφωνεί με τις τελευταίες προδιαγραφές του RDF. Η ουσιαστική συνεισφορά του Jena2 είναι το πλούσιο σύνολο βιβλιοθηκών που παρέχει για τη διαχείριση τέτοιου είδους δεδομένων. Ως κεντρική, εσωτερική διεπαφή το σύστημα αυτό προσφέρει μια αφαιρετική υλοποίηση του γραφήματος RDF η οποία χρησιμοποιείται σε κάθε περίπτωση

γραφημάτων, είτε αυτοί αναπτύσσονται εσωτερικά στη μνήμη, είτε μέσα σε μια βάση δεδομένων ή ακόμα και αν προκύπτουν μέσα από διαδικασίες εξαγωγής συμπερασμών. Το Jena υποστηρίζει συλλογισμό για το RDFS, την OWL-Lite και το DIG 1.1 ,προσφέρει συντακτικούς αναλυτές και εγγραφείς για τους μορφότυπους RDF/XML, N-NTriples, N3 και Turtle ενώ ως προς τις μεθόδους αποθήκευσης, ο χρήστης μπορεί να επιλέξει ανάμεσα στο να χρησιμοποιήσει τη μνήμη, είτε να προτιμήσει ένα σύστημα βάσης δεδομένων, προκειμένου να επιτύχει μόνιμη και ασφαλή διατήρηση των εγγράφων του. Τέλος, το Jena2 υποστηρίζει πλήρως τη γλώσσα επερωτήσεων SPARQL, τη χρήση της οποίας προτείνει στους χρήστες του, ενώ διατηρεί ακόμα υποστήριξη για μια πιο παλιά και λιγότερο δυναμική γλώσσα, την RDQL.

Ως προς την αρχιτεκτονική του, το Jena διέπεται από δύο βασικούς στόχους: Αρχικά, για τους προγραμματιστές εφαρμογών, στόχο αποτελεί η παρουσίαση των γραφημάτων RDF με πολλαπλές και συνάμα ευέλικτες μεθόδους. Με αυτό τον τρόπο διευκολύνεται η προσπέλαση στα γραφήματα αλλά και η διαχείρισή τους, επιτρέποντας στους συγκεκριμένους χρήστες να πλοηγούνται σε ολόκληρη τη δομή των τριπλετών. Απ’ την άλλη, για τους προγραμματιστές συστήματος, που επιθυμούν να εκθέτουν τα δεδομένα σαν τριπλέτες, επιδιώκεται μια πιο απλή, μινιμαλιστική οπτική των γραφημάτων. Ουσιαστικά, οι εφαρμογές που αναπτύσσονται σε ένα τέτοιο σύστημα αλληλεπιδρούν με ένα αφαιρετικό μοντέλο όπου μεταφράζει υψηλού επιπέδου λειτουργίες σε λειτουργίες χαμηλότερου επιπέδου κι αυτές με τη σειρά τους εφαρμόζονται πάνω σε τριπλέτες, αποθηκευμένες σε έναν γράφημα RDF.

5.2 Sesame

Το Sesame είναι ένα πλαίσιο ανοικτού κώδικα που προσφέρεται για την αποδοτική αποθήκευση και επερώτηση εγγράφων RDF, παρέχοντας ταυτοχρόνως πλήρη υποστήριξη στο RDF Schema. Επιπρόσθετα, παρέχει ένα σύνολο βιβλιοθηκών (API) για την διαχείριση δεδομένων RDF ενώ μπορεί να λειτουργήσει και ως εξυπηρετητής βάσεων δεδομένων για την προσπέλαση απομακρυσμένων αποθεμάτων. Μερικά από τα χαρακτηριστικά του είναι η υλοποίηση μηχανισμών εξαγωγής συμπερασμών (inferences), η υποστήριξη γνωστών γλωσσών επερωτήσεων για RDF καθώς και δημοφιλών πρωτοκόλλων για την υλοποίηση του μοντέλου επικοινωνίας πελάτη- εξυπηρετητή. Οι δυνατότητες αποθήκευσης που προσφέρει ποικίλουν και μπορεί να είναι εσωτερικά στη μνήμη, σε βάση δεδομένων ή σε αρχείο. Ωστόσο, ο σχεδιασμός και η υλοποίηση του δεν εξαρτώνται από την εκάστοτε χρησιμοποιούμενη μέθοδο αποθήκευσης, συνεπώς το Sesame μπορεί να αναπτυχθεί πάνω από οποιοδήποτε αποθηκευτικό μέσο χωρίς να χρειάζεται να τροποποιήσει το μηχανισμό επερωτήσεων ή κάποιο άλλο λειτουργικό του κομμάτι. Τέλος, το εν λόγω πλαίσιο, επειδή είναι υλοποιημένο σε Java, μπορεί να τρέξει πάνω από οποιαδήποτε πλατφόρμα ή λειτουργικό σύστημα που διαθέτει εικονική μηχανή της Java.

Η συγκεκριμένη έκδοση του Sesame εμφανίζεται αρκετά βελτιωμένη σε σχέση με την προκάτοχό της, την έκδοση 1.0, συμβαδίζοντας έτσι με τις τρέχουσες εξελίξεις στο πεδίο του RDF και του Σημασιολογικού Ιστού. Μερικά από τα χαρακτηριστικά του Sesame 2.0, τα οποία το καθιστούν ένα από τα πιο δημοφιλή συστήματα στη διαχείριση, διατήρηση και επερώτηση εγγράφων RDF περιγράφονται παρακάτω:

- **Ποικίλες μέθοδοι αποθήκευσης**

Το Sesame, όπως προαναφέρθηκε, υποστηρίζει διάφορες μεθόδους αποθήκευσης δίνοντας τη δυνατότητα στους χρήστες του να επιλέξουν αυτή που ανταποκρίνεται καλύτερα στις απαιτήσεις των εφαρμογών τους. Αν επιδιώκεται η βέλτιστη απόδοση προτιμάται η υλοποίησή του εσωτερικά μέσα στη μνήμη (*in-memory*) ή τοπικά (*native store*). Αν στόχο αποτελεί η διαχείριση μεγάλου όγκου δεδομένων RDF και η μόνιμη, ασφαλής διατήρησή τους, δίνεται αποθήκευσης σε βάση δεδομένων, χαρακτηριστικό που έχει ήδη ξεκινήσει να αναπτύσσεται και βρίσκεται σε ικανοποιητικό επίπεδο.

- **Υποστήριξη συμπερασμάτων**

Το Sesame 2.0 υποστηρίζει πλήρως τη λειτουργία εξαγωγής συμπερασμών για το RDF Schema. Επίσης, με τη βοήθεια του σημασιολογικού αποθέματος OWLIM, που έχει σχεδιαστεί κατάλληλα ώστε να μπορεί να ενσωματωθεί στην αρχιτεκτονική του Sesame, το τελευταίο μπορεί να προσφέρει μηχανισμούς εξαγωγής συμπερασμών και για την γλώσσα OWL.

- **Γλώσσες Επερωτήσεων**

Το Sesame 2.0 υποστηρίζει δύο αρκετά δημοφιλείς και εκφραστικές γλώσσες γι αυτό το σκοπό, τη SeRQL και τη SPARQL . Η SeRQL (**Se**same **R**DF **Q**uery **L**anguage) σχεδιάστηκε με στόχο για βελτιωμένη έκδοση των RQL και RDQL.
Όσον αφορά στη SPARQL, είναι η πλέον δημοφιλής γλώσσα σε αυτό τον πεδίο καθώς διανύει ήδη το τελικό στάδιο ανακήρυξής της σε πρότυπο του W3C.

- **Εισαγωγή και εξαγωγή δεδομένων σε διάφορους μορφότυπους**

Το Sesame 2.0 μπορεί να δεχθεί ή να εξάγει δεδομένα σε RDF/XML, N-Triples, Turtle, N3, Trig και TriX. Οι Μορφότυποι των αποτελεσμάτων επερώτησης

για RDF που υποστηρίζονται είναι η SPARQL/XML, η SPARQL/JSON καθώς και η δυαδική μορφή.

5.3 Jena Adapter

Το Jena Adapter για την βάση δεδομένων της Oracle παρέχει μία διεπαφή βασισμένη σε Java ενσωματώνοντας τα γνωστά στοιχεία της Jena και τα APIs.Για να χρησιμοποιήσεις το Jena Adapter πρέπει να έχεις εγκατεστημένη την Oracle Database 11*g* με ενεργοποιημένες της επιλογές Spatial, Partitioning και Semantic Technologies support και εγκατεστημένο το Jena Adapter επεκτείνοντας την διαχείριση των σημασιολογικών δεδομένων.

Χαρακτηριστικά και βελτιώσεις του Jena Adapter

- Υποστήριξη SPARQL , SPARUL
- Τα δεδομένα δεν αποθηκεύονται στην cache.
- Τα SPARQL ερωτήματα μετατρέπονται σε SQL και εκτελούνται μέσω της Oracle.
- Ποικίλοι τρόποι εισαγωγείς δεδομένων. Bulk/ Batch / Επαυξητικός.
- Εξωτερικά εργαλεία εξαγωγής συμπερασμάτων (Π.χ Pellet).

6. Η ΓΛΩΣΣΑ SPARQL

6.1 ΕΙΣΑΓΩΓΗ

Η SPARQL είναι η ποιο ευρέως διαδιδόμενη γλώσσα επερωτήσεων που υλοποιείται από όλα τα δημοφιλή συστήματα διαχείρισης δεδομένων που αναφέραμε προηγούμενος. Η SPARQL είναι μια SQL-like γλώσσα ερωτήσεων, και αποτελεί επίσημη πρόταση του W3C από τις 15 Ιανουαρίου 2008. Οι ερωτήσεις σε SPARQL αποτελούνται από γράφους εκφρασμένους σε τριάδες περιέχοντας ενώσεις, τομές και προαιρετικές εκφράσεις. Υπάρχουν πολλές επιτυχείς γλώσσες αναζήτησης, συμπεριλαμβανομένων προτύπων όπως SQL και XQuery. Αυτά είχαν σχεδιαστεί αρχικά για αναζητήσεις περιορισμένες σε ένα προϊόν, μια μορφή, τύπο πληροφοριών ή σε τοπικά αποθηκευμένα δεδομένα. Παραδοσιακά, ήταν απαραίτητο να δημιουργηθεί η ίδια αναζήτηση υψηλού επιπέδου διαφορετικά, ανάλογα με την εφαρμογή ή τη συγκεκριμένη διάταξη που επιλέγεται για τη βάση δεδομένων. Και όταν αναζητούνται πολλοί πόροι δεδομένων ήταν απαραίτητο να γραφτεί λογική για το συνδυασμό των αποτελεσμάτων. Αυτοί οι περιορισμοί έχουν επιφέρει υψηλότερο κόστος ανάπτυξης και δημιούργησαν φραγμούς για την ενσωμάτωση νέων πόρων.

Ο στόχος του Σημασιολογικού Ιστού είναι να δώσει τη δυνατότητα στους ανθρώπους να μοιραστούν, να ενσωματώσουν και να επαναχρησιμοποιήσουν τα δεδομένα παγκοσμίως. Η SPARQL έχει σχεδιαστεί για χρήση στον Παγκόσμιο Ιστό και επιτρέπει τις αναζητήσεις σε κατανεμημένους πόρους δεδομένων, ανεξαρτήτως μορφής. Δημιουργώντας μια μοναδική αναζήτηση σε εύρος δεδομένων, είναι ευκολότερο από το να υπάρχουν πολλές αναζητήσεις. Επίσης, στοιχίζει λιγότερο και παρέχει πλουσιότερα αποτελέσματα.

Επειδή η SPARQL δεν έχει περιορισμούς για συγκεκριμένη μορφή βάσης δεδομένων, μπορεί να χρησιμοποιηθεί από το κύμα δεδομένων του Web 2.0 και να το συνδυάσει με άλλους πόρους του Σημασιολογικού Ιστού. Ακόμα, επειδή οι απομακρυσμένοι πόροι μπορεί να μην έχουν το ίδιο 'σχήμα' ή να μοιράζονται τις ίδιες ιδιότητες, η SPARQL έχει σχεδιαστεί για αναζήτηση μη ομοιόμορφων δεδομένων.

6.2 Χαρακτηριστικά της SPARQL

Ως προς την σύνταξή της, η SPARQL βασίζεται στην Turtle , μια γλώσσα περιγραφής γραφημάτων RDF. Αυτό σημαίνει ότι χρησιμοποιεί κεφαλίδες ως βάσεις(*header bases*), προθέματα (*prefixes*) και την απλοποιημένη μέθοδο της N-Triples για τις περιγραφές RDF.

Η SPARQL υποστηρίζει τέσσερις διαφορετικούς τύπους επερωτήσεων: SELECT, DESCRIBE, ASK και CONSTRUCT. Και οι τέσσερις τύποι πραγματοποιούν επερωτήσεις που επιτελούν μόνο ανάγνωση στα δεδομένα και όχι εγγράψιμες ενημερώσεις. Η SPARQL μπορεί να απευθύνει επερωτήσεις πάνω σε αποθέματα, που είτε έχουν αποθηκευμένα γραφήματα RDF, είτε δεδομένα που εμφανίζονται στον τελικό χρήστη ως RDF με τη χρήση ενδιάμεσων εφαρμογών. Οι απαντήσεις σε μια επερώτηση SPARQL μπορεί να έχουν τη μορφή γραφημάτων RDF ή να αποτελούν ένα σύνολο αποτελεσμάτων (*result set*).

6.2.1 Μορφή Ερωτήσεων SPARQL

Οι απαντήσεις που αντιστοιχούν σε μια επερώτηση SPARQL, ανάλογα με το είδος της, έχουν την εξής μορφή:

- SELECT – Ταιριάζει τον γράφο που δίνεται στο τμήμα WHERE με τον συνολικό γράφο της βάσης μας και επιστρέφει τις τιμές που λαμβάνουν οι μεταβλητές σε κάθε ταίριασμα. Δεν επιστρέφονται οι τιμές των όλων μεταβλητών αλλά μόνο αυτές που αναφέρονται μετά την λέξη SELECT. Αν μετά την λέξη SELECT τοποθετήσουμε τον χαρακτήρα "*" τότε επιλέγονται όλες οι μεταβλητές.
- CONSTRUCT – Δημιουργεί ένα γράφημα RDF αντικαθιστώντας τις μεταβλητές με τα κατάλληλα δεδομένα, όπως αυτά προκύπτουν από τη δομή των μοτίβων τριπλετών μέσα στην επερώτηση.
- DESCRIBE - Οι ερωτήσεις αυτής της μορφής, επιστέφουν έναν RDF γράφο ο οποίος περιλαμβάνει RDF δεδομένα τα οποία “περιγράφουν” τις πηγές (Resource). Αντίθετα με τις CONSTRUCT ερωτήσεις η μορφή του γράφου δεν προσδιορίζεται από την ερώτηση αλλά ορίζεται από το SPARQL query engine. Η σχηματομορφή (Pattern) του ερωτήματος χρησιμοποιείται για να δημιουργηθεί ένα σύνολο αποτελεσμάτων. Οι ερωτήσεις αυτής της μορφής για κάθε πηγή που εμφανίζεται στα αποτελέσματα ή για κάθε πηγή που προσδιορίζεται από IRI, δημιουργούν μια “Περιγραφή”(Description) βασισμένη στα RDF δεδομένα. Τα αποτέλεσμα που παράγει μια ερώτηση αυτής της μορφής δεν είναι καθορισμένα από την προδιαγραφή της γλώσσας, αλλά καθορίζονται από την υλοποίηση του SPARQL query engine που εκτελεί την ερώτηση.
- ASK – Αυτό το ερώτημα επιστρέφει απλά την απάντηση τύπου Boolean true ή false, ανάλογα με το αν το ζητούμενο μοτίβο τριπλέτας έχει λύση ή όχι.

Επιπλέον, ενδιαφέρον παρουσιάζουν και ορισμένες εκφράσεις που μπορούν να μπουν σε μια επερώτηση SPARQL, χωρίς όμως να παίζουν ουσιαστικό ρόλο στην επεξεργασία των αποτελεσμάτων, καθώς ρυθμίζουν απλά τον τρόπο προβολής τους. Ενημερωτικά, πρόκειται για τις εξής:

• DISTINCT - αποτρέπει την επιστροφή της ίδιας απάντησης πολλές φορές

• ORDER BY – Ο τροποποιητής ORDER BY προσδιορίζει την σειρά της ακολουθίας των λύσεων. Η δομή ORDER BY, ακολουθείται από ένα σύνολο μεταβλητών με βάση των οποίων θα πραγματοποιηθεί η ταξινόμηση της ακολουθίας των λύσεων Προαιρετικά υπάρχει η δυνατότητα, να προσδιοριστεί η φορά της ταξινόμησης, δηλώνοντας ASC ή DESC για αύξουσα ή φθίνουσα ταξινόμηση αντίστοιχα, στην περίπτωση που δεν προσδιορίζεται η φορά ταξινόμησης, εφαρμόζεται αύξουσα.

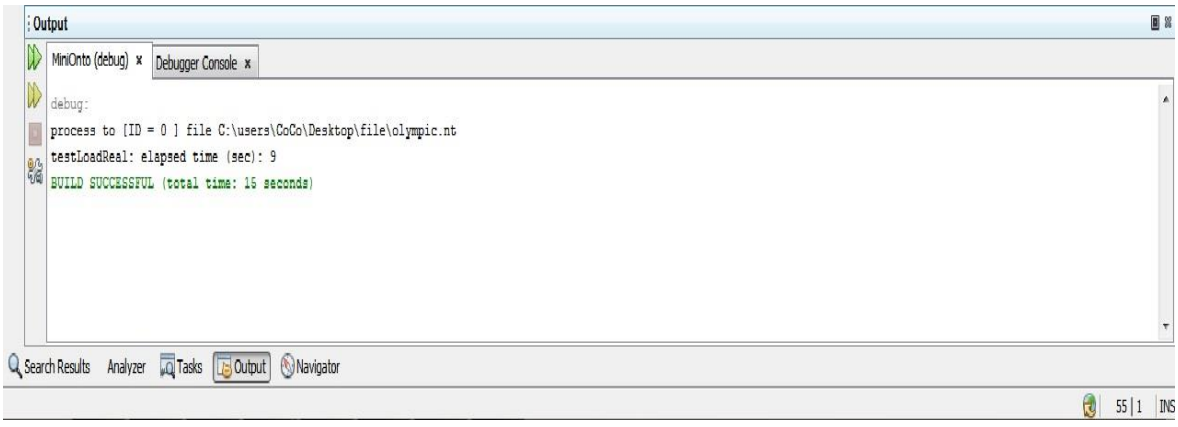
• LIMIT *n* – περιορίζει τον αριθμό των απαντήσεων σε *n*

• OFFSET *m* – προβάλλονται οι απαντήσεις ξεκινώντας από το αντικείμενο *m*

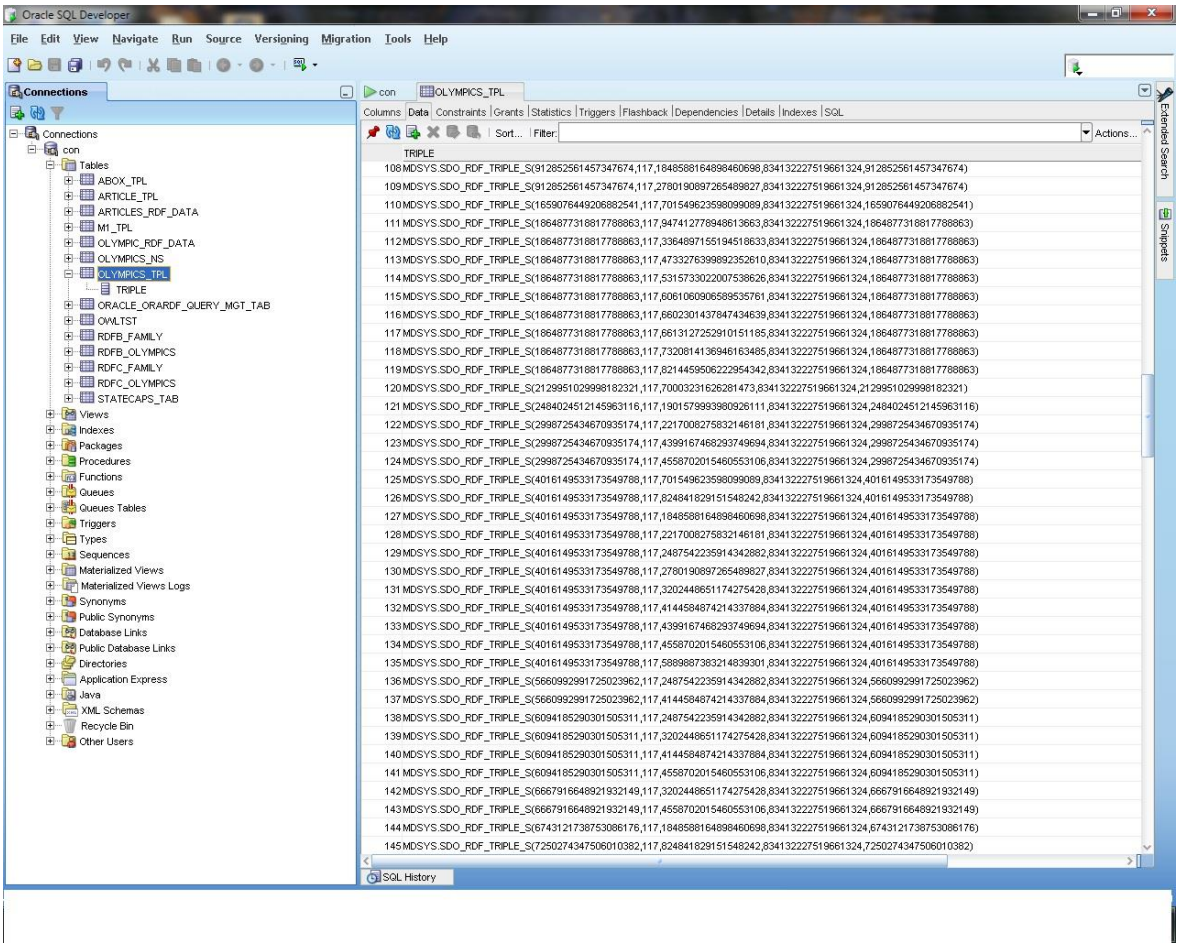
6.3 Σύνταξη SPARQL με Jena Adapter για Oracle

6.3.1 Εισαγωγή δεδομένων στην Oracle

Για την εισαγωγή των δεδομένων μας στην βάση δεδομένων ,Oracle 11g, χρησιμοποιήσαμε την μέθοδο εισαγωγής bulk load όπως φαίνεται στο παράρτημα 1. Για την εισαγωγή των δεδομένων χρησιμοποιήσαμε τα δεδομένα, που είναι σε N-Triples μορφή, που βρίσκονται στο παράρτημα 15.



Εικόνα 8-Αποτελέσματα φόρτωσης δεδομένων



Εικόνα 9-Κωδικοποιημένα δεδομένα στην Oracle

6.3.2 Select επερώτηση

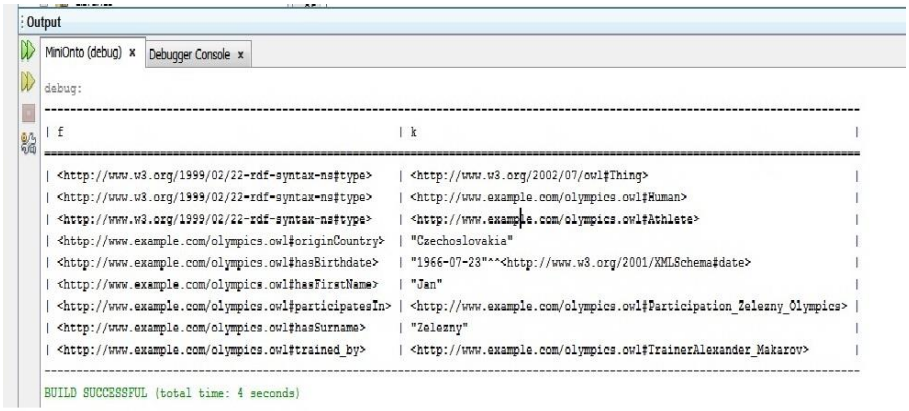
Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select ερώτημα όπως φαίνεται στο παράρτημα 2

```
select        ?f    ?k
WHERE  {          <http://www.example.com/olympics.owl#AthleteJan_Zelezny>
        ?f        ?k   .
```

}

Στο συγκεκριμένο ερώτημα θέλουμε να μάθουμε πληροφορίες για τον αθλητή Jan Zelezny.Δηλαδή το f και το k θα μας επιστρέψουν τα χαρακτηριστικά του αθλητή.

Όπως φαίνεται και στην Εικόνα 8 παίρνουμε πληροφορίες για τον αθλητή όπως το όνομα, επώνυμο, ημερομηνία γέννησης, χώρα καταγωγής , τον προπονητή του και σε ποιά διοργάνωση συμμετέχει .Επίσης περνούμε πληροφορίες ότι ο Jan Zelezny είναι τύπου Human και τύπου Athlete.



Εικόνα 10-Αποτελέσματα ερώτησης Select

6.3.3 Select επερώτηση με OPTIONAL

Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση OPTIONAL ερώτημα όπως φαίνεται στο παράρτημα 3.

```
select ?athlere_name  ?trainer_name  ?comes_from

WHERE  {  ?athlere_name  <http://www.example.com/olympics.owl#trained_by>
?trainer_name .

OPTIONAL {

?trainer_name  <http://www.example.com/olympics.owl#originCountry>
?comes_from } }
```

Στο συγκεκριμένο ερώτημα θέλουμε να μάθουμε πληροφορίες για τους αθλητές τους προπονητές τους καθώς και από ποια χώρα κατάγονται οι προπονητές τους. Για να λειτουργήσει το OPTIONAL στο παράδειγμα μας πρέπει να υπαρχεί αθλητής που να έχει προπονητή, και ο προπονητής να έχει όνομα, αλλιώς δεν μας επιστρέφει αποτέλεσμα το ερώτημα μας.

Εικόνα 11-Αποτελέσματα ερώτησης Select με OPTIONAL

6.3.4 Select επερώτηση με FILTER

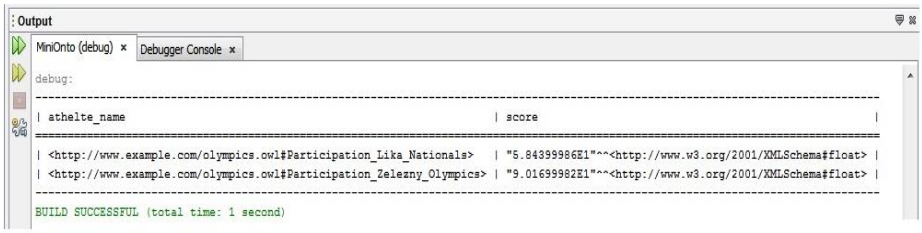
Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση FILTER ερώτημα όπως φαίνεται στο παράρτημα 4.

"SELECT ?athelte_name ?score "

+ " WHERE { ?athelte_name <http://www.example.com/olympics.owl#hasScore> ?score . "

+ " FILTER (?score > 50) }

Στο συγκεκριμένο ερώτημα θέλουμε να μάθουμε τους αθλητές που έχουν επίδοση στο άθλημα μεγαλύτερο από 50. Στο πεδίο Filter μπορούμε να χρησιμοποιήσουμε οτιδήποτε τύπου μεταβλητή.Π.χ(Integer, float, String).



Εικόνα 12-Αποτελέσματα ερώτησης Select με FILTER

6.3.5 Select επερώτηση με FILTER και OPTIONAL

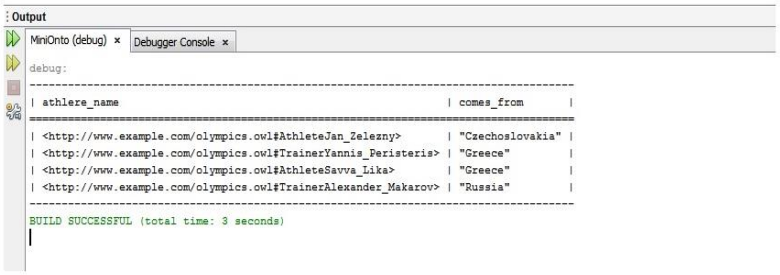
Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση FILTER και OPTIONAL ερώτημα όπως φαίνεται στο παράρτημα 5.

"select ?athlere_name ?comes_from "

+ "WHERE { ?athlere_name <http://www.example.com/olympics.owl#originCountry> ?comes_from . "

+ " OPTIONAL { ?comes_from <http://www.example.com/olympics.owl#hasBirthdate> ?age . "

+ " FILTER (bound(?age)) }} "



Εικόνα 13-Αποτελέσματα ερώτησης Select με FILTER και OPTIONAL

Στο συγκεκριμένο ερώτημα θέλουμε να μάθουμε πληροφορίες για την καταγωγή των αθλητών. Εδώ βλέπουμε και την λειτουργία bound() μέσα στο FILTER. Συντάσσεται με αυτόν τον τρόπο bound(?var) και στο συγκεκριμένο παράδειγμα θα μας επιστρέψει τους αθλητές που τους έχει καταχωρηθεί ηλικία.

6.3.6 Select επερώτηση με UNION

Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση UNION ερώτημα όπως φαίνεται στο παράρτημα 6.

Εικόνα 14-Αποτελέσματα ερώτησης Select με UNION

```
select ?athlete ?trainer "
        + "WHERE { "
        + "{?athlete <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.example.com/olympics.owl#Athlete> . } "
        + "UNION "
        + " { ?trainer <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.example.com/olympics.owl#Trainer> } } "
```

Το παράδειγμα αυτό το έχουμε χωρίσει σε 2 ξεχωριστά ερωτήματα. Στο πρώτο παίρνουμε τους αθλητές και στο δεύτερο τους προπονητές. Στο παράδειγμα με UNION έχουμε τα αποτελεσματα των δύο ερωτημάτων σε δύο ξεχωστους πίνακες και πάνω σε αυτούς μπορούμε να τρέξουμε ξεχωριστά ερωτήματα

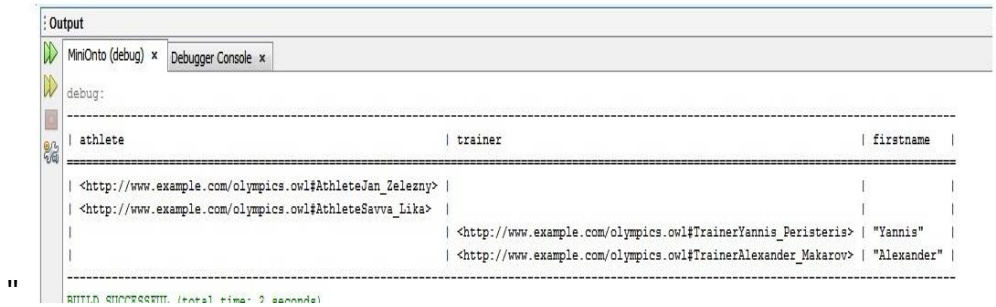
6.3.7 Select επερώτηση με UNION 2

Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση UNION ερώτημα όπως φαίνεται στο παράρτημα 7.

```
"select ?athlete ?trainer ?firstname "
        + "WHERE { "
        + "{?athlete <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.example.com/olympics.owl#Athlete> . } "
        + "UNION "
```

+ " { ?trainer <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.example.com/olympics.owl#Trainer> . "

+ " ?trainer <http://www.example.com/olympics.owl#hasFirstName> ?firstname } }



"

Εικόνα 15 - Αποτελέσματα ερώτησης Select με UNION

Το παράδειγμα 7 το έχουμε χωρίσει σε 2 ξεχωριστά ερωτήματα. Στο πρώτο παίρνουμε τους αθλητές και στο δεύτερο τους προπονητές. Επίσης έχουμε πάρει τα μικρά ονόματα μόνο των προπονητών.

6.3.8 Select επερώτηση με Order by

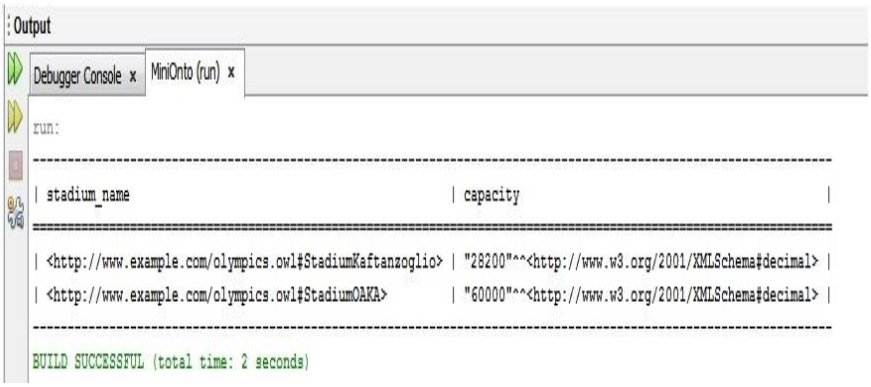
Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση UNION ερώτημα όπως φαίνεται στο παράρτημα 8.

"select ?stadium_name ?capacity"

+ " WHERE {"

+ " ?stadium_name <http://www.example.com/olympics.owl#hasSpectatorCapacity> ?capacity }"

+ "ORDER BY ?capacity"



Εικόνα 16-Αποτελέσματα ερώτησης Select με Order by

Στο παράρτημα 8 το έχουμε πάρει τα στάδια που έχουμε στην βάση μας και τα έχουμε ταξινομήσει σύμφωνα με την χωρητικότητα τους.Το Order by συντάσσεται με την Select και μετατρέπει την σειρά που θα έχει το αποτέλεσμα χωρίς να επηρεάζει τα δεδομένα.

.

6.3.9 Select επερώτηση με DISTINCT

Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση DISTINCT ερώτημα όπως φαίνεται στο παράρτημα 9.

"select DISTINCT ?Type WHERE {<http://www.example.com/olympics.owl#StadiumOAKA> ?Type ?o ."

+" } ");

Εικόνα 17-Αποτελέσματα ερώτησης Select με DISTINCT

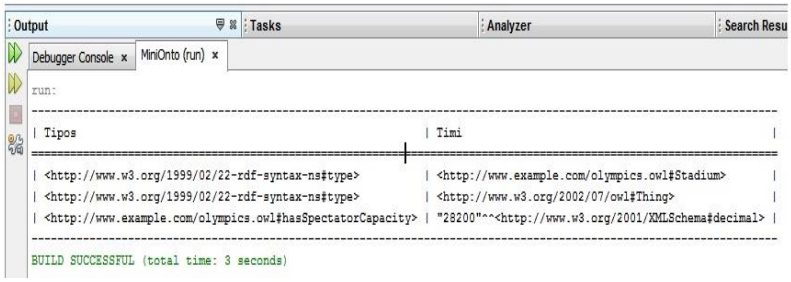
Στο παράδειγμα 9 παίρνουμε σαν αποτέλεσμα τους τύπους χωρίς επανάληψη που έχουν ανατεθεί στο στάδιο OAKA. Εδώ να προσθέσουμε ότι ο τελεστής REDUCED μπορεί να αντικαταστήσει τον τελεστή DISTINCT ακριβώς με τα ίδια αποτελέσματα.

6.3.10 Select επερώτηση με LIMIT

Στο συγκεκριμένο παράδειγμα εκτελούμε ένα Select με χρήση LIMIT ερώτημα όπως φαίνεται στο παράρτημα 10.

"select ?Tipos ?Timi WHERE {<http://www.example.com/olympics.owl#StadiumKaftanzoglio> ?Tipos ?Timi . }"

+ "LIMIT 3");



Εικόνα 18-Αποτελέσματα ερώτησης Select με LIMIT

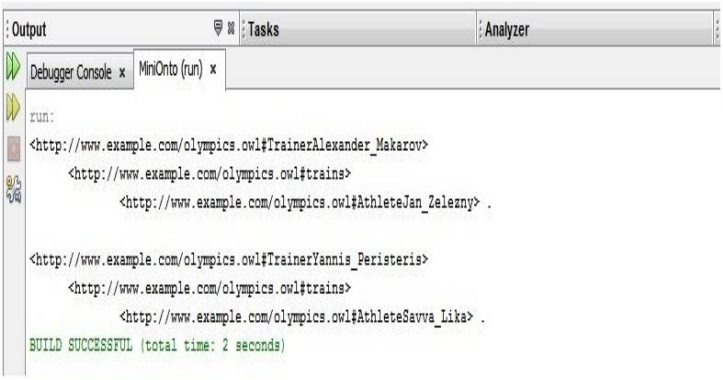
Στο παράρτημα 10 παίρνουμε σαν αποτέλεσμα τους τύπους που έχουν ανατεθεί στο στάδιο Καυτατζογλειο και έχουμε περιορίσει τα αποτελέσματα που θα πάρουμε ανάλογα με τον αριθμό που θα γράψουμε δίπλα στην λέξη LIMIT

6.3.11 Construct

Όπως αναφέραμε και προηγούμενος το construct δημιουργεί ένα γράφημα RDF αντικαθιστώντας τις μεταβλητές με τα κατάλληλα δεδομένα, όπως αυτά προκύπτουν από τη δομή των μοτίβων τριπλετών μέσα στην επερώτηση.

" CONSTRUCT { ?athlete <http://www.example.com/olympics.owl#trains> ?trainer } " +

" WHERE { ?athlete <http://www.example.com/olympics.owl#trains> ?trainer } " ;



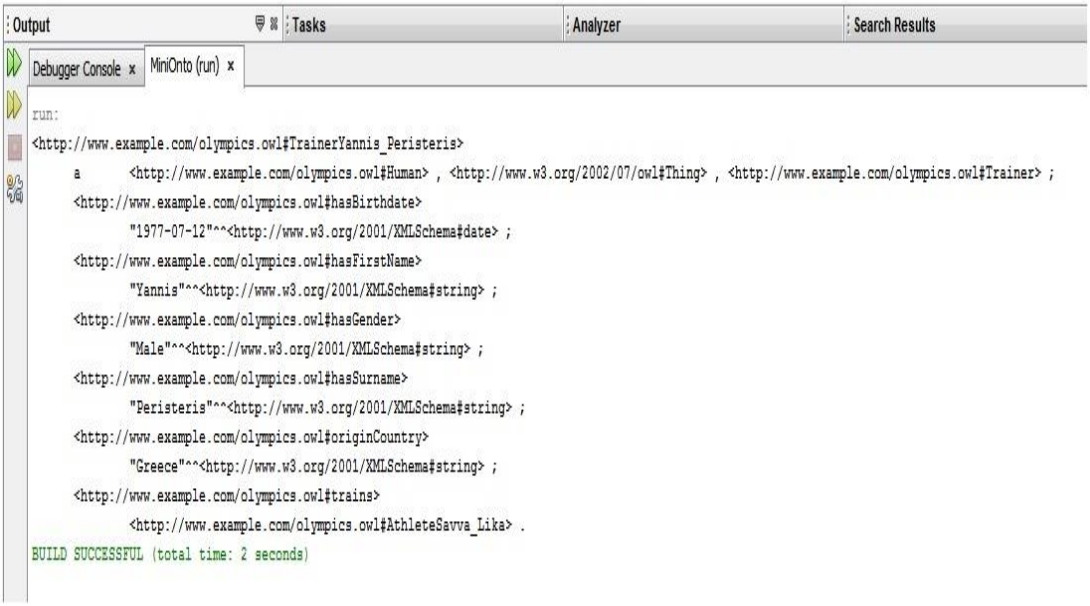
Εικόνα 19-Αποτελέσματα CONSTRUCT

Ενώ το Select ερώτημα επιστρέφει μία λύση το construct μας επιστρέφει σαν αποτέλεσμα ένα γράφο που του έχουμε ορίσει μέσα στην ερώτηση. Στο παράρτημα 11 βλέπουμε τους αθλητές και τους προπονητές τους. Στο Construct μπορούμε να έχουμε απο ένα μέχρι δυο αγνωστους μέσα σε ενα ερώτημα. Εάν ειναι και οι τρεις άγνωστοι θα μας επιστρέψει όλο το γράφο.

6.3.12 Describe

Οι ερωτήσεις αυτής της μορφής, επιστέφουν έναν RDF γράφο ο οποίος περιλαμβάνει RDF δεδομένα τα οποία “περιγράφουν” τις πηγές (Resource). Αντίθετα με τις CONSTRUCT ερωτήσεις η μορφή του γράφου δεν προσδιορίζεται από την ερώτηση αλλά ορίζεται από το SPARQL query engine.

```
String queryString =

    " DESCRIBE ?y  " +

    "  WHERE      { ?y <http://www.example.com/olympics.owl#hasFirstName>
'Yannis' } ";
```



Εικόνα 20-Αποτελέσματα Describe

Στο παράρτημα 11 βλέπουμε όλες της πληροφορίες που σχετίζονται με το όνομα "Yannis". Στο Describe ερώτημα όπου η SPARQL query engine κάνει έλεγο μέσα στον γράφο έχουμε τουλάχιστον ένα άγνωστο στο ερώτημα του Describe και απο εκεί και πέρα μας εμφανίζει τον γράφω που είναι σχετικός με το ερώτημα μας.

6.3.13 Ask

Αυτό το ερώτημα επιστρέφει απλά την απάντηση τύπου Boolean true ή false, ανάλογα με το αν το ζητούμενο μοτίβο τριπλέτας έχει λύση ή όχι.

String queryString =

" ASK { ?x <http://www.example.com/olympics.owl#inGame> <http://www.example.com/olympics.owl#GameNationals2009> .} ";

Εικόνα 21-Αποτέλεσμα Ask

Στο παράρτημα 13 κάνουμε θέλουμε να δούμε εάν υπάρχει αθλητής που συμμετείχε στους εθνικούς αγώνες του 2009 και η απάντηση είναι TRUE.

6.3.14 Ask 2

Σε αυτό το σημείο να τονίσουμε ότι η SPARQL είναι case sensitive.

String queryString =

" ASK { ?x <http://www.example.com/olympics.owl#hasFirstName> 'Kostas' .} ";

Εικόνα 22-Αποτέλσμα Ask

Στο παράρτημα 14 ρωτάμε εάν υπάρχει άτομο με μικρό όνομα "Kostas" και η απάντηση είναι FALSE.

ΕΠΙΛΟΓΟΣ

Η σύγχρονη τάση είναι να προσδιορίζεται και εννοιολογικά η πληροφορία δίνοντας με αυτόν τον τρόπο τη δυνατότητα στο χρήστη ενός φιλτραρίσματος της πληροφορίας και επιτρέποντας την καλύτερη ανάδραση με το σύστημα. Για να επιτευχθεί αυτό τα δεδομένα μας πρέπει να έχουν την κατάλληλη σύνταξη. Έτσι για να αναπαραστήσουμε τα δεδομένα χρησιμοποιούμε την RDF. Για την αναπαράσταση των δεδομένων χρηρισιμοποιούμε URIs για να περιγράψουμε τις οντολογίες. Έχοντας μια τριπλέτα απο URIs που μπορούν να γραφούν σε N-Triples,Turtle, RDF/XML περιγράφοντας μία οτνολογία ή να περιγράφουν δεδομένα για τα δεδομένα (metadata).Το Sesame, το Jena και το Jena Adapter για την Oracle τα τρία πιο γνωστά πλαίσια διαχείρισης εγγράφων προσφέρουν δυνατότητες για αποθήκευση και επεξεργασία των δεδομένων. Το Jena Adapter για Oracle, οπου ασχοληθήκαμε στην συγκεκριμένη πτυχιακή εργασία, προσφέρει ενα ασφαλές περιβάλλον για αποθήκευση των δεδομένων. Με την JAVA μπορούμε να εκτελέσουμε SPARQL ερωτήματα μέσω του Jena Adapter, που είναι προέκταση του Jena API. Δημιουργώντας στο μέλλον μία "έξυπνη" εφαρμογή αναζήτησης που να εκτελεί τα SPARQL επερωτήματα θα μειώσει τον χρόνο αναζήτησης και έχουμε απ'ευθειας το αποτέλεσμα που επιθυμούμε.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Bill Beauregard S.P.P.M., Souri Das P.D., Oracle, Matthew Perry P.D., Oracle, Karl Rieb O., Seema Sundara O. (2011) Oracle Database Semantic Technologies:Understanding How to Install, Load, Query and Inference, in: Oracle (Ed.).

Dean Allemang J.A.H. (2008) Semantic web for the working ontologist: modeling in RDF, RDFS and OWL.

Dokulil J. (2006) Evaluation of SPARQL Queries Using Relational Databases Department of Software Engineering,Faculty of Mathematics and Pgysics, Charles University in Prague, Czech Republic.

Erling O. (2010) Implementing a SPARQL compliant RDF Triple Store using a SQL-ORDBMS.

Florian Stegmaier U.G.M.D., Harald Kosch and Gero, Baese. (2011) Evaluation of Current RDF Database Solutions:17.

Harmelen. G.A.a.F.v. (2008) A Semantic Web Primer. 2nd ed.

Jinsoo Lee M.-D.P., Jihwan Lee , Wook-Shin Han, Hune Cho, Hwanjo Yu and Jeong-Hoon Lee. (2011) Processing SPARQL queries with regular expressions in RDF databases.

Manola F. (2004) RDF Primer, in: H.-P. L. Brian McBride (Ed.), w3.

McBride B., Labs. H.-P. (2002) Jena: a semantic Web toolkit. Internet Computing, IEEE 6.

Mikhailov O.E.a.I. Business Intelligence Extensions for SPARQL. from www.OpenLinksw.com

Murray C. (2010) Oracle Database Semantic Technologies Developer's Guide, 11g Release 2 in: Oracle (Ed.), Oracle. OpenRDF.

Parsia E.S.a.B. (2006) SPARQL-DL: SPARQL Query for OWL-DL.

Pollock J.T. (2009) Semantic Web for Dummies.

Richard Cyganiak D., Wood D. (2011) RDF Concepts and Abstract Syntax, in: D. Richard Cyganiak and D. Wood (Eds.).

Seaborne E.P.h.A. (2008) SPARQL Query Language for RDF, in: E. P. h. A. Seaborne (Ed.).

Shadbolt N.H., W.; Berners-Lee, T.; . (2006) The Semantic Web Revisited. Intelligent Systems, IEEE 21.

Tim Berners-Lee J.H.a.O.L. (2001) A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American: The Semantic Web.w3school.

Xavier Lopez P.D., Zhe Wu P.D., Souripriya Das P.D. (2009) Put SPARQL in Your Code: Building Applications with Oracle Semantic Technologies, Oracle.

Αριστογιάννης Δ.Γ. (2010) Διαχείριση Πληροφορίας & Τεχνικές Κατηγοριοποίησης στο Διαδίκτυο, Μεταπτυχιακό πρόγραμμα Σπουδών: "Δυνητικές Κοινότητες: Κοινωνιο-Ψυχολογικές Προσεγγίσεις και Τεχνικές Εφαρμογές. pp. 79.

Μανώλης Μ. (2008) Ανάπτυξη συστήματος παροχής εξατομικευμένων εκπαιδευτικών εμπειριών από ψηφιακές βιβλιοθήκες οπτικοακουστικού υλικού, Τμήμα Ηλεκτρονικών Μηχανικών & Μηχανικών Υπολογιστών, Πολυτεχνείο Κρήτης, Κρήτη. pp. 238.

Σολομού Γ. (2008) Δημιουργία μηχανισμού επερώτησης και διατήρηση κατανεμημένου αποθέματος εγγράφων RDF στον παγκόσμιο ιστό, Πανεπιστήμιο Πατρών Πολυτεχνική Σχόλη, πανεπιστήμιο Πατρών, πάτρα. σελ. 132.

Τζουβάρας Β. (2006) Σημασιολογικός Ιστός, Εθνικό Μετσόβιο Πολυτεχνείο.

ΙΣΤΟΣΕΛΙΔΕΣ

http://blog.garethj.com/2010/01/11/using-jena-as-a-sparql-endpoint/

http://clarkparsia.com/pellet

http://en.wikipedia.org/wiki/Jena_(framework)

http://en.wikipedia.org/wiki/N-Triples

http://en.wikipedia.org/wiki/Resource_Description_Framework

http://en.wikipedia.org/wiki/Sesame_(framework)

http://en.wikipedia.org/wiki/Web_Ontology_Language

http://jena.sourceforge.net/ontology/index.html

http://wiki.dbpedia.org/Ontology

http://www.hpl.hp.com/semweb/

http://www.hpl.hp.com/semweb/

http://www.joseki.org/

http://www.linkeddatatools.com/introducing-rdfs-owl

http://www.linkeddatatools.com/querying-semantic-data

http://www.w3.org/2004/03/trix/

http://www.w3.org/MarkUp/SGML/

ΠΑΡΑΡΤΗΜΑΤΑ

Το παράρτημα περιέχει τον κώδικα σε Java που χρησιμοποιήθηκε για φόρτωση τον δεδομένων στην Oracle, τον κώδικα για για επερώτηση και τέλος τα δεδομένα που είναι γραμμένα σε N-Triple μορφή.

Για τα παρακάτω παραδείγματα χρησιμοποιήθηκε το η Oracle 11g release 2 και το NetBeans 6.9.1 με τις παρακάτω βιβλιοθήκες:

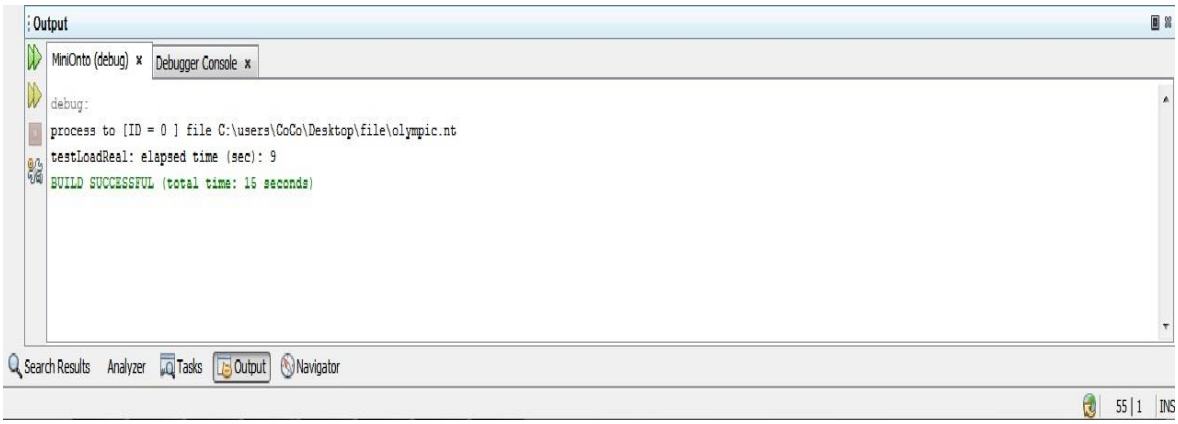
jena-2.6.3.jar,arq-2.8.4.jar, sdordf.jar, ojdbc6.jar, sdordfclient.jar, slf4j-api-1.5.8.jar, slf4-log4j12-1.5.8.jar, log4j-1.2.13.jar, xercesImpl-2.7.1jar, iri-0.8.jar, icu4j-3.4.4.jar και το JDK 1.6

ΠΑΡΑΡΤΗΜΑ 1. Εισαγωγή δεδομένων στην Oracle με bulk loader

```
import java.io.*;

import com.hp.hpl.jena.query.*;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.rdf.model.*;

import com.hp.hpl.jena.util.*;

import oracle.spatial.rdf.client.jena.*;

public class Bulk {

    public static void main(String[] args) throws Exception{

long startTime = System.currentTimeMillis();
```

```
String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl"; //URL για την βάση μας

String szUser    = "kostas"; //Όνομα χρήστη

String szPasswd  = "welcome"; //Κωδικός

String szModelName = "Olympics"; //Μοντέλο που δημιουργούμε ή υπάρχει

Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);

GraphOracleSem graph = new GraphOracleSem(oracle, szModelName); //Σύνδεση στην βάση

PrintStream psOut = System.out;

String dirname = "C:\\users\\CoCo\\Desktop\\file"; //Directory που έχουμε τα αρχεία μας

File fileDir = new File(dirname);

String[] szAllFiles = fileDir.list();

// Σάρωση για παραπάνω από ένα αρχεία

for (int idx = 0; idx < szAllFiles.length; idx++) {

  String szIndFileName = dirname + File.separator + szAllFiles[idx];

  psOut.println("process to [ID = " + idx + " ] file " + szIndFileName);

  psOut.flush();

  try {

    InputStream is = new FileInputStream(szIndFileName);
```

```
graph.getBulkUpdateHandler().prepareBulk(
  is,                  // input stream
      "http://example.com",  // base URI
      "N-TRIPLE",            // data file type: can be RDF/XML, N-TRIPLE, etc.
      "SEMTS",               // tablespace
      null,                  // flags
      null,                  // listener
      null                   // staging table name.
      );
 is. close();  }
 catch (Throwable t) {
   psOut.println("Hit exception " + t.getMessage());  }
graph.getBulkUpdateHandler().completeBulk(null, null);
  psOut.println("testLoadReal: elapsed time (sec): " +
     ((System.currentTimeMillis() - startTime) / 1000));
}}}
```

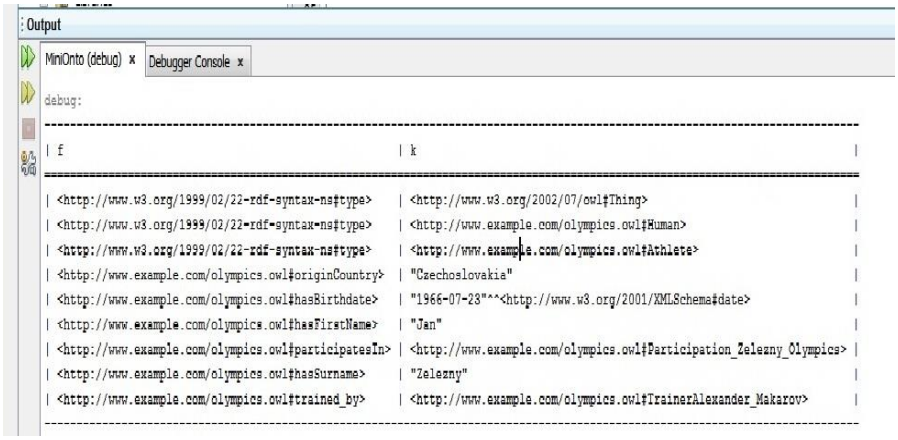


Εικόνα 8-Αποτελέσματα φόρτωσης δεδομένων

ΠΑΡΑΡΤΗΜΑ 2.Select επερώτηση

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class Test {

  public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";

    String szModelName = "Olympics";
```

```
    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
    Model model = ModelOracleSem.createOracleSemModel(
      oracle, szModelName);
    Query query = QueryFactory.create(
            "select ?f ?k WHERE
{<http://www.example.com/olympics.owl#AthleteJan_Zelezny> ?f ?k .}" );
QueryExecution qexec = QueryExecutionFactory.create(query, model);
    ResultSet results = qexec.execSelect();
    ResultSetFormatter.out(System.out, results, query);


    model.close();
    oracle.dispose();
  }
}
```



Εικόνα 10-Αποτελέσματα ερώτησης Select

ΠΑΡΑΡΤΗΜΑ 3. Select επερώτηση με OPTIONAL

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class Optional {

  public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";

    String szModelName = "Olympics";

    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
```

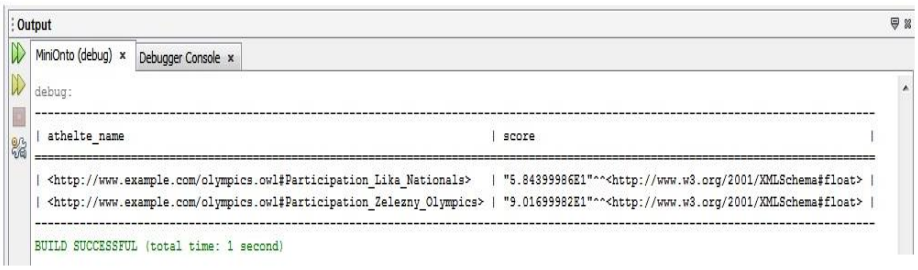
```
    Model model = ModelOracleSem.createOracleSemModel(
      oracle, szModelName);
       Query query = QueryFactory.create(
       "select ?athlere_name ?trainer_name ?comes_from WHERE { ?athlere_name <http://www.example.com/olympics.owl#trained_by> ?trainer_name . "
              + " OPTIONAL {?trainer_name <http://www.example.com/olympics.owl#originCountry> ?comes_from } }" );
    QueryExecution qexec = QueryExecutionFactory.create(query, model);
    ResultSet results = qexec.execSelect();
    ResultSetFormatter.out(System.out, results, query);
    model.close();
 oracle.dispose();
  }
}
```

Εικόνα 11-Αποτελέσματα ερώτησης Select με OPTINAL

ΠΑΡΑΡΤΗΜΑ 4. Select επερώτηση με FILTER

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class filter {

  public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";

    String szModelName = "Olympics";

    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);

    Model model = ModelOracleSem.createOracleSemModel(

      oracle, szModelName);

        Query query = QueryFactory.create(

        " SELECT ?athelte_name ?score "

     +          "          WHERE          {          ?athelte_name

<http://www.example.com/olympics.owl#hasScore>  ?score . "
```

```
    + " FILTER ( ?score > 50 ) } ");

   QueryExecution qexec = QueryExecutionFactory.create(query, model);

   ResultSet results = qexec.execSelect();

   ResultSetFormatter.out(System.out, results, query);

   model.close();

   oracle.dispose();

 }}
```



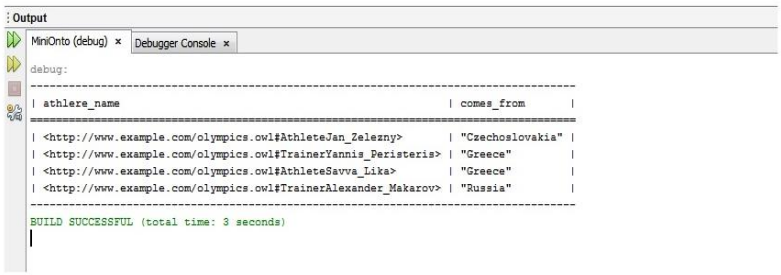
Εικόνα 12-Αποτελέσματα ερώτησης Select με FILTER

ΠΑΡΑΡΤΗΜΑ 5. Select επερώτηση με FILTER και OPTIONAL

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;
```

```
public class FilterOptional {
  public static void main(String[] args) throws Exception
  {
    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";
    String szUser    = "kostas";
    String szPasswd  = "welcome";
    String szModelName = "Olympics";
    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
    Model model = ModelOracleSem.createOracleSemModel(
      oracle, szModelName);
       Query query = QueryFactory.create(
       "select ?athlere_name ?comes_from   "
            +          "WHERE          {          ?athlere_name
<http://www.example.com/olympics.owl#originCountry> ?comes_from . "
            +          "          OPTIONAL          {          ?comes_from
<http://www.example.com/olympics.owl#hasBirthdate> ?age .   "
            + " FILTER (bound(?age)) }} ");
    QueryExecution qexec = QueryExecutionFactory.create(query, model);
    ResultSet results = qexec.execSelect();
```

```
  ResultSetFormatter.out(System.out, results, query);

  model.close();

  oracle.dispose();

}}
```



Εικόνα 13-Αποτελέσματα ερώτησης Select με FILTER και OPTIONAL

ΠΑΡΑΡΤΗΜΑ 6. Select επερώτηση με Union

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class Union {

  public static void main(String[] args) throws Exception

  {
```

```
String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";
String szUser    = "kostas";
String szPasswd  = "welcome";
String szModelName = "Olympics";
Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
Model model = ModelOracleSem.createOracleSemModel(
  oracle, szModelName);
   Query query = QueryFactory.create(
   "select ?athlete ?trainer "
         + "WHERE { "
         +      "{?athlete      <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.example.com/olympics.owl#Athlete> . } "
         + "UNION "
         +   "   {   ?trainer   <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.example.com/olympics.owl#Trainer> } } " );
QueryExecution qexec = QueryExecutionFactory.create(query, model);
ResultSet results = qexec.execSelect();
ResultSetFormatter.out(System.out, results, query);
model.close();
```

```
    oracle.dispose();
  }
}
```

Εικόνα 14-Αποτελέσματα ερώτησης Select με UNION

ΠΑΡΑΡΤΗΜΑ 7. Select επερώτηση με Union 2

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class Union {

  public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";
```

```
String szModelName = "Olympics";
Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
Model model = ModelOracleSem.createOracleSemModel(
  oracle, szModelName);
    Query query = QueryFactory.create(
    "select ?athlete ?trainer ?firstname "
          + "WHERE { "
          +     "{?athlete     <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.example.com/olympics.owl#Athlete> . } "
          + "UNION "
          +  "  {  ?trainer  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.example.com/olympics.owl#Trainer> . "
          +  "  ?trainer  <http://www.example.com/olympics.owl#hasFirstName>
?firstname } } " );
QueryExecution qexec = QueryExecutionFactory.create(query, model);
ResultSet results = qexec.execSelect();
ResultSetFormatter.out(System.out, results, query);
model.close();
oracle.dispose();
 }
```

}

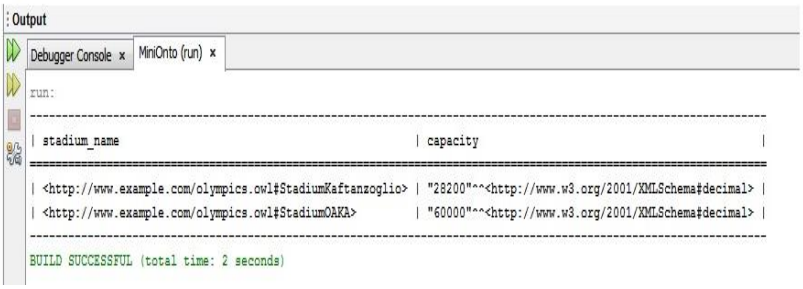
Εικόνα 15-Αποτελέσματα ερώτησης Select με UNION 2

ΠΑΡΑΡΤΗΜΑ 8. Select επερώτηση με ORDER BY

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;


public class Order {


  public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";
```

```
String szPasswd  = "welcome";


String szModelName = "Olympics";


Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
Model model = ModelOracleSem.createOracleSemModel(
  oracle, szModelName);


   Query query = QueryFactory.create(
   "select ?stadium_name ?capacity"
         + " WHERE {"
         +                    "                    ?stadium_name
<http://www.example.com/olympics.owl#hasSpectatorCapacity>  ?capacity  }"
         + "ORDER BY ?capacity" );
QueryExecution qexec = QueryExecutionFactory.create(query, model);
ResultSet results = qexec.execSelect();
ResultSetFormatter.out(System.out, results, query);


model.close();
```

```
    oracle.dispose();
  }
}
```



Εικόνα 16-Αποτελέσματα ερώτησης Select με Order by

ΠΑΡΑΡΤΗΜΑ 9. Select επερώτηση με DISTINCT

```
import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.graph.*;
import com.hp.hpl.jena.query.*;
import oracle.spatial.rdf.client.jena.*;
public class Distinct {
  public static void main(String[] args) throws Exception
  {
    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";
```

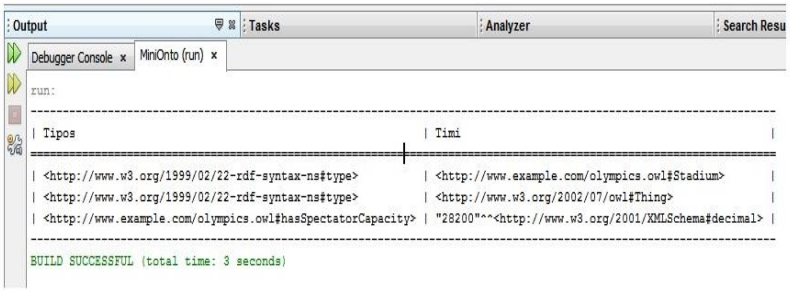
```
String szUser    = "kostas";

String szPasswd  = "welcome";

String szModelName = "Olympics";

Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);

Model model = ModelOracleSem.createOracleSemModel(

  oracle, szModelName);

   Query query = QueryFactory.create(

   "select DISTINCT ?Type WHERE
{<http://www.example.com/olympics.owl#StadiumOAKA>  ?Type  ?o ."

          +" } ");



QueryExecution qexec = QueryExecutionFactory.create(query, model);

ResultSet results = qexec.execSelect();

ResultSetFormatter.out(System.out, results, query);

model.close();

oracle.dispose();  }}
```

Εικόνα 17-Αποτελέσματα ερώτησης Select με DISTINCT

ΠΑΡΑΡΤΗΜΑ 10. Select επερώτηση με LIMIT

```
import com.hp.hpl.jena.rdf.model.Model;

import com.hp.hpl.jena.graph.*;

import com.hp.hpl.jena.query.*;

import oracle.spatial.rdf.client.jena.*;

public class Limit {

  public static void main(String[] args) throws Exception  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";

    String szModelName = "Olympics";

    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
```

```
Model model = ModelOracleSem.createOracleSemModel(
 oracle, szModelName);
  Query query = QueryFactory.create(
 "select ?Tipos ?Timi WHERE {<http://www.example.com/olympics.owl#StadiumKaftanzoglio> ?Tipos ?Timi . } LIMIT 3");
QueryExecution qexec = QueryExecutionFactory.create(query, model);
 ResultSet results = qexec.execSelect();
 ResultSetFormatter.out(System.out, results, query);
 model.close();
 oracle.dispose(); }}
```



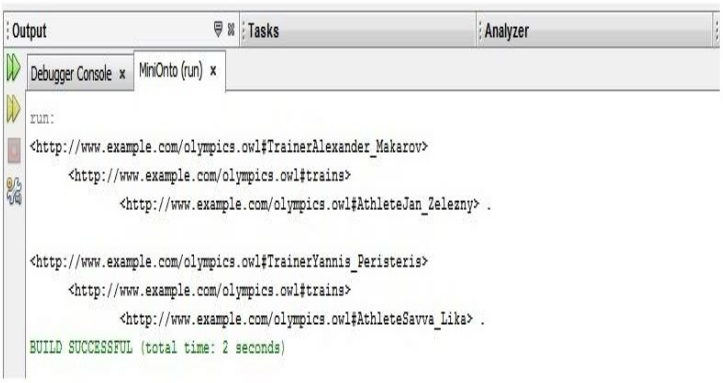
Εικόνα 18-Αποτελέσματα ερώτησης Select με LIMIT

ΠΑΡΑΡΤΗΜΑ 11. Construct

```
import oracle.spatial.rdf.client.jena.*;
import com.hp.hpl.jena.query.Query;
```

```
import com.hp.hpl.jena.query.QueryExecution;

import com.hp.hpl.jena.query.QueryExecutionFactory;

import com.hp.hpl.jena.query.QueryFactory;

import com.hp.hpl.jena.rdf.model.Model;

public class contrust {

    public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

 String szPasswd  = "welcome";

    String szModelName = "Olympics";

    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);

    GraphOracleSem graph = new GraphOracleSem(oracle, szModelName, false);

    ModelOracleSem model = new ModelOracleSem(graph);

    {

      String queryString =

        " CONSTRUCT    { ?athlete   <http://www.example.com/olympics.owl#trains>
?trainer } " +
```

```
        " WHERE { ?athlete <http://www.example.com/olympics.owl#trains>
?trainer }" ;
     Query query = QueryFactory.create(queryString);
     QueryExecution qexec = QueryExecutionFactory.create(query, model);
     Model m = qexec.execConstruct();
          m.write(System.out, "TURTLE");
}
   graph.close();
   oracle.dispose();
  }
}
```

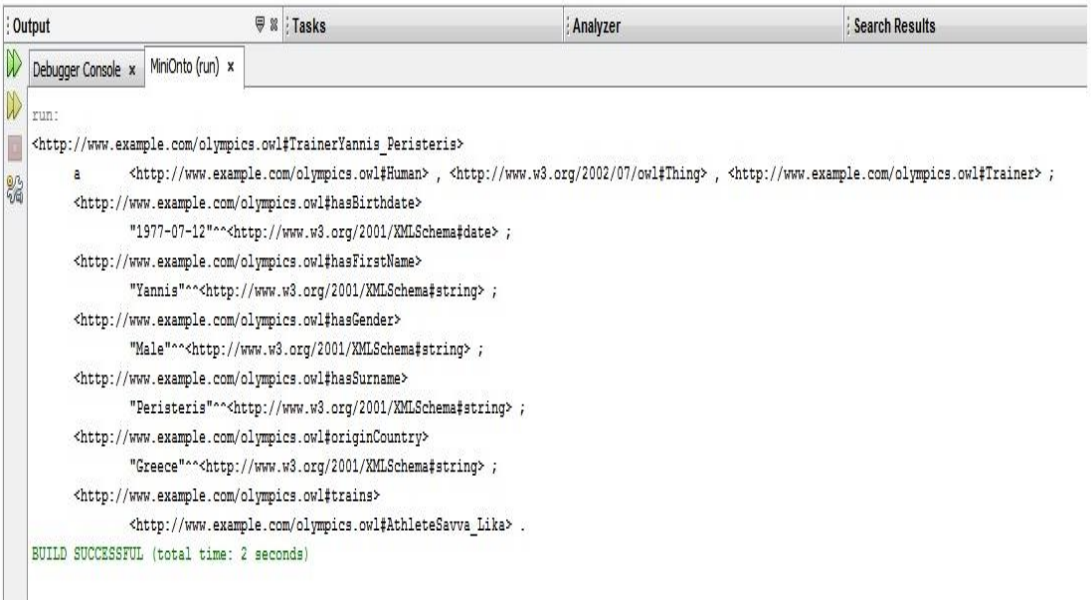


Εικόνα 19-Αποτελέσματα CONSTRUCT

ΠΑΡΑΡΤΗΜΑ 12. Describe

```
import oracle.spatial.rdf.client.jena.*;




import com.hp.hpl.jena.graph.Node;

import com.hp.hpl.jena.graph.Triple;



import com.hp.hpl.jena.query.Query;

import com.hp.hpl.jena.query.QueryExecution;

import com.hp.hpl.jena.query.QueryExecutionFactory;

import com.hp.hpl.jena.query.QueryFactory;

import com.hp.hpl.jena.query.ResultSet;

import com.hp.hpl.jena.query.ResultSetFormatter;

import com.hp.hpl.jena.rdf.model.Model;

public class describe {

   public static void main(String[] args) throws Exception

 {

   String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";
```

```
    String szUser    = "kostas";

    String szPasswd  = "welcome";

    String szModelName = "Olympics";

    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);

    GraphOracleSem graph = new GraphOracleSem(oracle, szModelName, false);

    ModelOracleSem model = new ModelOracleSem(graph);

String queryString =

        " DESCRIBE ?y  " +

        "  WHERE        {  ?y  <http://www.example.com/olympics.owl#hasFirstName>
'Yannis' } ";

      Query query = QueryFactory.create(queryString);

      QueryExecution qexec = QueryExecutionFactory.create(query, model);

           Model m = qexec.execDescribe();

           m.write(System.out, "TURTLE");



    graph.close();

    oracle.dispose();

    }

  }
```



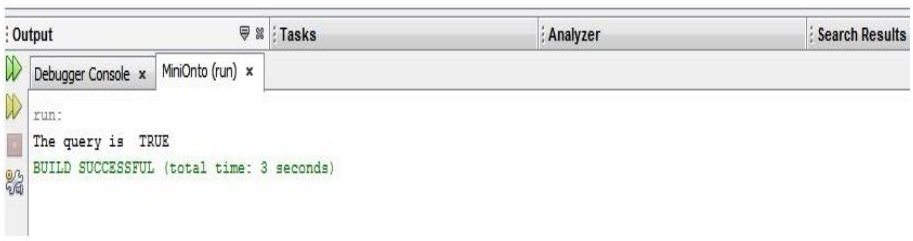
Εικόνα 20-Αποτελέσματα Describe

ΠΑΡΑΡΤΗΜΑ 13.ASK

```
import oracle.spatial.rdf.client.jena.*;

import com.hp.hpl.jena.query.Query;

import com.hp.hpl.jena.query.QueryExecution;

import com.hp.hpl.jena.query.QueryExecutionFactory;

import com.hp.hpl.jena.query.QueryFactory;

import com.hp.hpl.jena.rdf.model.Model;

import java.lang.String;

public class ask {

    public static void main(String[] args) throws Exception
```

```
{
  String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";
  String szUser    = "kostas";
  String szPasswd  = "welcome";
  String szModelName = "Olympics";
  Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
  GraphOracleSem graph = new GraphOracleSem(oracle, szModelName, false);
  ModelOracleSem model = new ModelOracleSem(graph);
  {
    String queryString =
   "      ASK      {      ?x      <http://www.example.com/olympics.owl#inGame>
<http://www.example.com/olympics.owl#GameNationals2009> .} ";
    Query query = QueryFactory.create(queryString);
    QueryExecution qexec = QueryExecutionFactory.create(query, model);
    boolean b = qexec.execAsk();
    System.out.println("The query is  " + ((b)?"TRUE":"FALSE"));
  }
   graph.close();
  oracle.dispose();
```

```
    }
}
```



Εικόνα 21-Αποτελέσματα Ask

ΠΑΡΑΡΤΗΜΑ 14.ASK 2

```
import oracle.spatial.rdf.client.jena.*;

import com.hp.hpl.jena.query.Query;

import com.hp.hpl.jena.query.QueryExecution;

import com.hp.hpl.jena.query.QueryExecutionFactory;

import com.hp.hpl.jena.query.QueryFactory;

public class ask2 {

    public static void main(String[] args) throws Exception

  {

    String szJdbcURL = "jdbc:oracle:thin:@192.168.2.3:1521:orcl";

    String szUser    = "kostas";

    String szPasswd  = "welcome";
```

```
    String szModelName = "Olympics";
    Oracle oracle = new Oracle(szJdbcURL, szUser, szPasswd);
    GraphOracleSem graph = new GraphOracleSem(oracle, szModelName, false);
    ModelOracleSem model = new ModelOracleSem(graph);
    {
      String queryString =
" ASK { ?x <http://www.example.com/olympics.owl#hasFirstName> 'Kostas' .} ";
      Query query = QueryFactory.create(queryString);
      QueryExecution qexec = QueryExecutionFactory.create(query, model);
      boolean b = qexec.execAsk();
      System.out.println("The query is  " + ((b)?"TRUE":"FALSE"));
    }
     graph.close();
    oracle.dispose();
    }
}
```

Εικόνα 22-Αποτέλσμα Ask

ΠΑΡΑΡΤΗΜΑ 15 Οντολογία Ολυμπιακών Αγώνων σε N-Triple

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.example.com/olympics.owl#hasScore>

"58.44"^^<http://www.w3.org/2001/XMLSchema#float> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.example.com/olympics.owl#hasScore>

"90.17"^^<http://www.w3.org/2001/XMLSchema#float> .

<http://www.example.com/olympics.owl#SportMaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Sport> .

<http://www.example.com/olympics.owl#SportFemaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Sport> .

<http://www.example.com/olympics.owl#StadiumKaftanzoglio>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#StadiumOAKA>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#aboutSport>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#participatesIn>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#playedIn>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#trained_by>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#trains> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#inGame> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.w3.org/2002/07/owl#topObjectProperty>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://www.example.com/olympics.owl#hasScore>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#FunctionalProperty> .

<http://www.example.com/ontology/olympic-games-inferred.owl>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Ontology> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#StadiumKaftanzoglio>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#StadiumOAKA>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#SportMaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#SportFemaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Thing> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Trainer> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Trainer> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#SportMaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#MaleSport> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#hasScore>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasLevel>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#originCountry>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasBirthdate>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasSpectatorCapacity>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#inCity> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasQualification>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasFirstName>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasStadiumName>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasSurname>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.w3.org/2002/07/owl#topDataProperty>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#sportName>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#hasGender>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#DatatypeProperty> .

<http://www.example.com/olympics.owl#Sport> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Stadium>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Participation>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Class> .

<http://www.w3.org/2002/07/owl#Thing> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Trainer> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Human> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Athlete> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#MaleSport>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#Game> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#FemaleSport>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2002/07/owl#Class> .

<http://www.example.com/olympics.owl#SportFemaleJavelin>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.example.com/olympics.owl#FemaleSport> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.example.com/olympics.owl#AthleteJan_Zelezny> .

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.example.com/olympics.owl#AthleteSavva_Lika> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.example.com/olympics.owl#hasLevel>

"Final"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.example.com/olympics.owl#hasLevel>

"Final"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.example.com/olympics.owl#hasLevel> "Semi-Final"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasScore>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasLevel>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#originCountry>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasBirthdate>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasSpectatorCapacity>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#inCity> <http://www.w3.org/2000/01/rdf-schema#subPropertyOf> <http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasQualification>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasFirstName>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasStadiumName>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasSurname>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#sportName>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#hasGender>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topDataProperty> .

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#aboutSport>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#participatesIn>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#playedIn>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#trained_by>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#trains> <http://www.w3.org/2000/01/rdf-schema#subPropertyOf> <http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#inGame> <http://www.w3.org/2000/01/rdf-schema#subPropertyOf> <http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.w3.org/2000/01/rdf-schema#subPropertyOf>

<http://www.w3.org/2002/07/owl#topObjectProperty> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.example.com/olympics.owl#originCountry>

"Czechoslovakia"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.example.com/olympics.owl#originCountry>

"Greece"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#originCountry>

"Greece"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.example.com/olympics.owl#originCountry>

"Russia"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>
<http://www.example.com/olympics.owl#hasBirthdate> "1970-06-27"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://www.example.com/olympics.owl#AthleteJan_Zelezny>
<http://www.example.com/olympics.owl#hasBirthdate> "1966-07-23"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>
<http://www.example.com/olympics.owl#hasBirthdate> "1973-03-19"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>
<http://www.example.com/olympics.owl#hasBirthdate> "1977-07-12"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://www.example.com/olympics.owl#participatesIn>
<http://www.w3.org/2002/07/owl#inverseOf>
<http://www.example.com/olympics.owl#participationOfAthlete> .
<http://www.example.com/olympics.owl#trained_by>
<http://www.w3.org/2002/07/owl#inverseOf>
<http://www.example.com/olympics.owl#trains> .
<http://www.example.com/olympics.owl#inGame>
<http://www.w3.org/2002/07/owl#inverseOf>
<http://www.example.com/olympics.owl#consistsOfParticipation> .
<http://www.example.com/olympics.owl#StadiumKaftanzoglio>
<http://www.example.com/olympics.owl#hasSpectatorCapacity>
"28200"^^<http://www.w3.org/2001/XMLSchema#int> .

<http://www.example.com/olympics.owl#StadiumOAKA>

<http://www.example.com/olympics.owl#hasSpectatorCapacity>

"60000"^^<http://www.w3.org/2001/XMLSchema#int> .

<http://www.example.com/olympics.owl#StadiumKaftanzoglio>

<http://www.example.com/olympics.owl#inCity>

"Thessaloniki"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.example.com/olympics.owl#aboutSport>

<http://www.example.com/olympics.owl#SportMaleJavelin> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.example.com/olympics.owl#aboutSport>

<http://www.example.com/olympics.owl#SportFemaleJavelin> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.example.com/olympics.owl#hasQualification>

"Q"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.example.com/olympics.owl#hasQualification>

"Q"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.example.com/olympics.owl#hasFirstName>

"Yannis"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.example.com/olympics.owl#hasFirstName>

"Alexander"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#hasFirstName>

"Savva"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.example.com/olympics.owl#hasFirstName>

"Jan"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#sportName>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Sport> .

<http://www.example.com/olympics.owl#hasSpectatorCapacity>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#inCity> <http://www.w3.org/2000/01/rdf-schema#domain> <http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#hasStadiumName>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#hasScore>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#hasQualification>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#inGame> <http://www.w3.org/2000/01/rdf-schema#domain> <http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#trains> <http://www.w3.org/2000/01/rdf-schema#domain> <http://www.example.com/olympics.owl#Trainer> .

<http://www.example.com/olympics.owl#originCountry>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#hasBirthdate>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#hasFirstName>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#hasSurname>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#hasGender>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Human> .

<http://www.example.com/olympics.owl#participatesIn>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#trained_by>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#hasLevel>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#aboutSport>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#playedIn>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.w3.org/2000/01/rdf-schema#domain>

<http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.example.com/olympics.owl#participatesIn>

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#participatesIn>

<http://www.example.com/olympics.owl#Participation_Lika_Nationals> .

<http://www.example.com/olympics.owl#MaleSport>

<http://www.w3.org/2000/01/rdf-schema#subClassOf>

<http://www.example.com/olympics.owl#Sport> .

<http://www.example.com/olympics.owl#FemaleSport>
<http://www.w3.org/2000/01/rdf-schema#subClassOf>
<http://www.example.com/olympics.owl#Sport> .
<http://www.example.com/olympics.owl#Sport> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.w3.org/2002/07/owl#Thing> .
<http://www.example.com/olympics.owl#Stadium>
<http://www.w3.org/2000/01/rdf-schema#subClassOf>
<http://www.w3.org/2002/07/owl#Thing> .
<http://www.example.com/olympics.owl#Participation>
<http://www.w3.org/2000/01/rdf-schema#subClassOf>
<http://www.w3.org/2002/07/owl#Thing> .
<http://www.example.com/olympics.owl#Human> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.w3.org/2002/07/owl#Thing> .
<http://www.example.com/olympics.owl#Game> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.w3.org/2002/07/owl#Thing> .
<http://www.example.com/olympics.owl#Trainer> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.example.com/olympics.owl#Human> .
<http://www.example.com/olympics.owl#Athlete> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.example.com/olympics.owl#Human> .
<http://www.example.com/ontology/olympic-games-inferred.owl>
<http://www.w3.org/2000/01/rdf-schema#comment> "An example ontology based on a subset of Olympic Games."@en .
<http://www.example.com/olympics.owl#GameOlympics2004>
<http://www.example.com/olympics.owl#playedIn>
<http://www.example.com/olympics.owl#StadiumOAKA> .

<http://www.example.com/olympics.owl#StadiumOAKA>
<http://www.example.com/olympics.owl#hasStadiumName> "OAKA Stadium"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://www.example.com/olympics.owl#GameOlympics2004>
<http://www.example.com/olympics.owl#playedIn>
<http://www.example.com/olympics.owl#StadiumKaftanzoglio> .
<http://www.example.com/olympics.owl#GameNationals2009>
<http://www.example.com/olympics.owl#playedIn>
<http://www.example.com/olympics.owl#StadiumKaftanzoglio> .
<http://www.example.com/olympics.owl#StadiumKaftanzoglio>
<http://www.example.com/olympics.owl#hasStadiumName> "Kaftanzoglio Stadium"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>
<http://www.example.com/olympics.owl#hasSurname>
"Peristeris"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://www.example.com/olympics.owl#AthleteJan_Zelezny>
<http://www.example.com/olympics.owl#hasSurname>
"Zelezny"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>
<http://www.example.com/olympics.owl#hasSurname>
"Makarov"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://www.example.com/olympics.owl#AthleteSavva_Lika>
<http://www.example.com/olympics.owl#hasSurname>
"Lika"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#trained_by>

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris> .

<http://www.example.com/olympics.owl#AthleteJan_Zelezny>

<http://www.example.com/olympics.owl#trained_by>

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov> .

<http://www.example.com/olympics.owl#hasScore>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#float> .

<http://www.example.com/olympics.owl#aboutSport>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Sport> .

<http://www.example.com/olympics.owl#playedIn>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Stadium> .

<http://www.example.com/olympics.owl#participatesIn>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Participation> .

<http://www.example.com/olympics.owl#hasLevel>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#originCountry>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#inCity> <http://www.w3.org/2000/01/rdf-schema#range> <http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasQualification>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasFirstName>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasStadiumName>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasSurname>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#sportName>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#hasGender>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#trained_by>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Trainer> .

<http://www.example.com/olympics.owl#hasSpectatorCapacity>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#int> .

<http://www.example.com/olympics.owl#participationOfAthlete>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#trains> <http://www.w3.org/2000/01/rdf-schema#range> <http://www.example.com/olympics.owl#Athlete> .

<http://www.example.com/olympics.owl#inGame> <http://www.w3.org/2000/01/rdf-schema#range> <http://www.example.com/olympics.owl#Game> .

<http://www.example.com/olympics.owl#hasBirthdate>

<http://www.w3.org/2000/01/rdf-schema#range>

<http://www.w3.org/2001/XMLSchema#date> .

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.example.com/olympics.owl#trains>

<http://www.example.com/olympics.owl#AthleteJan_Zelezny> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.example.com/olympics.owl#trains>

<http://www.example.com/olympics.owl#AthleteSavva_Lika> .

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics>

<http://www.example.com/olympics.owl#inGame>

<http://www.example.com/olympics.owl#GameOlympics2004> .

<http://www.example.com/olympics.owl#Participation_Lika_Nationals>

<http://www.example.com/olympics.owl#inGame>

<http://www.example.com/olympics.owl#GameNationals2009> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#inGame>

<http://www.example.com/olympics.owl#GameNationals2009> .

<http://www.example.com/olympics.owl#SportMaleJavelin>

<http://www.example.com/olympics.owl#sportName>

"Javelin"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#SportFemaleJavelin>

<http://www.example.com/olympics.owl#sportName>

"Javelin"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#GameOlympics2004>

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.example.com/olympics.owl#Participation_Zelezny_Olympics> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.example.com/olympics.owl#Participation_Lika_Nationals> .

<http://www.example.com/olympics.owl#GameNationals2009>

<http://www.example.com/olympics.owl#consistsOfParticipation>

<http://www.example.com/olympics.owl#AthleteSavva_Lika> .

<http://www.example.com/olympics.owl#TrainerYannis_Peristeris>

<http://www.example.com/olympics.owl#hasGender>

"Male"^^<http://www.w3.org/2001/XMLSchema#string>

<http://www.example.com/olympics.owl#TrainerAlexander_Makarov>

<http://www.example.com/olympics.owl#hasGender>

"Male"^^<http://www.w3.org/2001/XMLSchema#string> .

<http://www.example.com/olympics.owl#AthleteSavva_Lika>

<http://www.example.com/olympics.owl#hasGender>

"Female"^^<http://www.w3.org/2001/XMLSchema#string> .