**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**
**ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**
**ΣΧΟΛΗ ΝΑΥΤΙΛΙΑΣ ΚΑΙ ΒΙΟΜΗΧΑΝΙΑΣ**
**ΤΜΗΜΑΤΟΣ ΒΙΟΜΗΧΑΝΙΚΗΣ ΔΙΟΙΚΗΣΗΣ & ΤΕΧΝΟΛΟΓΙΑΣ**

ΔΙΕΠΙΣΤΗΜΟΝΙΚΟ – ΔΙΑΠΑΝΕΠΙΣΤΗΜΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
«ΤΕΧΝΟ-ΟΙΚΟΝΟΜΙΚΑ ΣΥΣΤΗΜΑΤΑ»

# Διαδραστικός Δημιουργός Ερωτημάτων SPARQL

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Δημήτριος, Ι. Μπεκιάρης

**Επιβλέπων :** Ιωάννης Αναγνωστόπουλος
Καθηγητής, Παν. Θεσσαλίας

Αθήνα, Φεβρουάριος 2025

**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**
**ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**
**ΣΧΟΛΗ ΝΑΥΤΙΛΙΑΣ ΚΑΙ ΒΙΟΜΗΧΑΝΙΑΣ**
**ΤΜΗΜΑΤΟΣ ΒΙΟΜΗΧΑΝΙΚΗΣ ΔΙΟΙΚΗΣΗΣ & ΤΕΧΝΟΛΟΓΙΑΣ**

ΔΙΕΠΙΣΤΗΜΟΝΙΚΟ – ΔΙΑΠΑΝΕΠΙΣΤΗΜΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
«ΤΕΧΝΟ-ΟΙΚΟΝΟΜΙΚΑ ΣΥΣΤΗΜΑΤΑ»

# Διαδραστικός Δημιουργός Ερωτημάτων SPARQL

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Δημήτριος, Ι. Μπεκιάρης

**Επιβλέπων :** Ιωάννης Αναγνωστόπουλος
Καθηγητής, Παν. Θεσσαλίας

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13η Φεβρουαρίου 2025.

| | | |
|---|---|---|
| ..... | ..... | ..... |
| Ασκούνης Δημήτριος | Μάριος Κόνιαρης | Ραζής Γεράσιμος |
| Καθηγητής ΕΜΠ | ΕΔΙΠ ΕΜΠ | Δόκτωρ Παν. Θεσσαλίας |

Αθήνα, Φεβρουάριος 2025

Δημήτριος, Ι. Μπεκιάρης
Διπλωματούχος μεταπτυχιακού προγράμματος (ΔΠΜΣ) "Τεχνο-Οικονομικά Συστήματα"
της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών ΕΜΠ

Copyright © Δημήτριος, Μπεκιάρης, 2025
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

# Περίληψη

Η εξαγωγή και διαχείριση δεδομένων από σύνθετες βάσεις γνώσεων είναι ένα πολύπλοκο και απαιτητικό πρόβλημα, καθώς απαιτεί όχι μόνο ανάλυση δεδομένων, αλλά και την κατανόηση των σημασιολογικών δομών που τις διέπουν. Επιπλέον, η χρήση εξειδικευμένων μεθόδων για τη σύνταξη ερωτημάτων είναι κρίσιμη για την αποτελεσματική αλληλεπίδραση με αυτές τις βάσεις. Στην παρούσα διπλωματική εργασία, αναπτύχθηκε ένα εργαλείο για τη σύνταξη και διαχείριση ερωτημάτων SPARQL, το οποίο επιτρέπει στους χρήστες να δημιουργούν και να εκτελούν ερωτήματα σε βάσεις δεδομένων RDF μέσω ενός εύχρηστου γραφικού περιβάλλοντος.

Η εφαρμογή που δημιουργήθηκε, «SPARQL Query Builder», προσφέρει ένα απλό και φιλικό περιβάλλον που επιτρέπει ακόμα και σε χρήστες χωρίς ιδιαίτερη εμπειρία με την γλώσσα SPARQL να διαμορφώσουν και να τρέξουν τα ερωτήματά τους. Η ανάπτυξη της εφαρμογής έγινε με τη χρήση του .NET Framework και της γλώσσας προγραμματισμού VB, με κύριο στόχο την ευχρηστία και την ευχάριστη εμπειρία χρήστη.

Στην εργασία αναλύονται και άλλα εργαλεία που προσφέρουν παρόμοιες δυνατότητες, με συγκρίσεις στα πλεονεκτήματα και τα σημεία όπου αυτά ενδέχεται να υστερούν σε σχέση με την εφαρμογή που αναπτύχθηκε. Παράλληλα, δίνεται έμφαση στα χαρακτηριστικά που καθιστούν το «SPARQL Query Builder» καινοτόμο και χρήσιμο, όπως και στους τρόπους βελτίωσης του στο μέλλον.

Τέλος σημειώνεται είναι ότι η εφαρμογή αυτή είναι λογισμικό ανοιχτού κώδικα (https://github.com/bekdim072/SparqlQueryBuilder.git), επιτρέποντας στην κοινότητα προγραμματιστών να συνεισφέρει στην ανάπτυξή της και να προσθέσει νέες δυνατότητες και επεκτάσεις, ενισχύοντας έτσι την ευχρηστία και τις δυνατότητές της μελλοντικά.

**Λέξεις Κλειδιά:**

SPARQL, RDF, Βάσεις Γνώσεων, Σύνταξη Ερωτημάτων, Γραφικό Περιβάλλον, .NET Framework, VB, Διαχείριση Δεδομένων, Λογισμικό Ανοιχτού Κώδικα, Ανάπτυξη Εφαρμογών, Σημασιολογικός Ιστός.

# Abstract

Extracting and managing data from complex knowledge bases is a challenging and demanding task, as it requires not only data analysis but also an understanding of the semantic structures that govern them. Additionally, the use of specialized methods for query formulation is crucial for effective interaction with these databases.

In this thesis, a tool for composing and managing SPARQL queries was developed, allowing users to create and execute queries on RDF databases through an intuitive graphical interface. The application, named **"SPARQL Query Builder"**, provides a simple and user-friendly environment that enables even users with limited experience in SPARQL to construct and run their queries with ease. The development of the application was carried out using the **.NET Framework** and the **VB programming language**, with a primary focus on usability and an enjoyable user experience.

The thesis also analyzes other tools offering similar functionalities, comparing their advantages and potential shortcomings relative to the developed application. Emphasis is placed on the features that make the **"SPARQL Query Builder"** innovative and useful, as well as on possible future improvements.

Finally, it is worth noting that this application is **open-source software** (**https://github.com/bekdim072/SparqlQueryBuilder.git**), allowing the developer community to contribute to its development by adding new features and extensions. This enhances its usability and capabilities for the future.

**Keywords:**

SPARQL, RDF, Knowledge Bases, Query Formulation, Graphical User Interface, .NET Framework, VB Programming, Data Management, Open-Source Software, Semantic Web, Application Development.

## Πίνακας περιεχομένων

Κεφάλαιο 1: Εισαγωγή ..... 14

Κεφάλαιο 2: Υφιστάμενα Εργαλεία ..... 15

2.1 SPARQL Builder ..... 15

2.2 Sparnatural ..... 17

2.3 Sparklis ..... 19

Κεφάλαιο 3: Σύνταξη Ερωτημάτων SPARQL ..... 22

3.1 Σύνθεση Ερωτημάτων SPARQL ..... 22

3.2 Είδη Ερωτημάτων SPARQL ..... 23

3.3 Χρησιμοποίηση Φίλτρων και Σύνθετων Ερωτημάτων ..... 23

3.4 Βελτιστοποίηση και Παράλληλη Επεξεργασία Ερωτημάτων ..... 23

3.5 Εφαρμογές και Σημασία της SPARQL ..... 24

Κεφάλαιο 4: Εγχειρίδιο Εφαρμογής ..... 25

4.1 Αρχικό Μενού ..... 25

4.2 Επιλογή Data Source ..... 27

4.3 Πάνελ Ιεραρχίας ..... 29

4.4 Εμφάνιση Αποτελεσμάτων ..... 32

Κεφάλαιο 5: Δημιουργία Ερωτημάτων SPARQL ..... 34

Κεφάλαιο 6: Τεκμηρίωση Κώδικα ..... 44

6.1 Φόρτωση Πηγών Δεδομένων ..... 44

6.1.1 Ανάκτηση Γράφων ..... 44

6.1.2 Ανάκτηση Subject ..... 45

6.1.3 Ανάκτηση Ιδιοτήτων (Predicates) ..... 46

6.2 Δημιουργία Ερωτημάτων SPARQL ..... 46

6.2.1 Εκτέλεση Ερωτήματος ..... 47

6.2.2 Φόρμα Results ..... 48

6.2.3 Άνοιγμα Αρχείου Excel ..... 50

6.3 Δυναμικό TreeView ..... 51

6.4 Module Query Manager ..... 57

6.5 Κώδικας GroupBy ..... 61

6.6 Διαχείριση Ρυθμίσεων ..... 63

Κεφάλαιο 7: Συμπεράσματα και Μελλοντικές Επεκτάσεις ..... 67

Βιβλιογραφία ..... 69

## Πίνακας Εικόνων

Εικόνα 1.  SPARQLBuilder ....16

Εικόνα 2.  Sparnatural ....18

Εικόνα 3.  Sparklis ....20

Εικόνα 4. Επιλογή end point ....25

Εικόνα 5. Διαθέσιμα end points ....26

Εικόνα 6. Προσθήκη end point ....26

Εικόνα 7. Διαγραφή end point ....26

Εικόνα 8. Επιλογή end point και άνοιγμα Query Builder για να μεταφερθούμε στο περιβάλλον όπου γράφουμε τα ερωτήματά μας ....27

Εικόνα 9. Επιλογή "Default" end point ....27

Εικόνα 10. Γράφοι σε δεντρική μορφή. ....28

Εικόνα 11. Επιλογή Γράφου. ....29

Εικόνα 12. Επιλογή Subject. ....30

Εικόνα 13. Ερώτημα που αφορά σε ένα Subject ....30

Εικόνα 14. Επιλογή Predicate. ....30

Εικόνα 15. Εισαγωγή επιπλέον δηλώσεων ....31

Εικόνα 16. Επιλογή και προσθήκη μετβλητών ....31

Εικόνα 17. Εκτέλεση ερωτήματος ....32

Εικόνα 18. Φόρμα "Results" ....32

Εικόνα 19. Data grid ....33

Εικόνα 20. Excel ....33

Εικόνα 21. Επιλογή Data Source ..... 34

Εικόνα 22. Επιλογή Subject ..... 35

Εικόνα 23. Προσθήκη Statement ..... 35

Εικόνα 24. Επιλογή Predicate ..... 36

Εικόνα 25. Ορισμός μεταβλητής ..... 36

Εικόνα 26. Φόρμα Φιλτραρίσματος ..... 37

Εικόνα 27. Κώδικας φιλτραρίσματος ..... 37

Εικόνα 28. Εισαγωγή κώδικα φιλτραρίσματος στο Ερώτημα ..... 38

Εικόνα 29. Επιλογή μεταβλητών ..... 38

Εικόνα 30. Αποτελέσματα ..... 39

Εικόνα 31. Επιλογή Subject ..... 40

Εικόνα 32. Προσθήκη Statement ..... 40

Εικόνα 33.Φόρμα Group By ..... 41

Εικόνα 34. Τελικός κώδικας ερωτήματος ..... 41

Εικόνα 35. Αποτελέσματα ..... 42

Εικόνα 36. Αποτελέσματα σε Data Grid ..... 42

Εικόνα 37. Αποτελέσματα σε Excel ..... 43

Εικόνα 38. Λίστα Data Source ..... 44

Εικόνα 39. Λίστα Γράφων ..... 44

Εικόνα 40. Λίστα Subject ..... 45

Εικόνα 41. Λίστα Predicates ..... 46

Εικόνα 42. Πλήκτρο Insert Where ..... 46

Εικόνα 43. Πλήκτρο εκτέλεσης ερωτήματος ..... 47

Εικόνα 44. Φόρμα Αποτελεσμάτων ..... 48

Εικόνα 45. Εξαγωγή σε Excel ..... 50

Εικόνα 46. Tree View ..... 51

Εικόνα 47. Επέκταση Γράφου ..... 54

Εικόνα 48. Αποθηκευμένα Εωτήματα ..... 57

Εικόνα 49. Φόρμα Group By ..... 61

Εικόνα 50. Μορφή αρχείου ρυθμίσεων ..... 63

# Κεφάλαιο 1: Εισαγωγή

Η σύγχρονη εποχή χαρακτηρίζεται από την εκρηκτική αύξηση των δεδομένων σε όλους σχεδόν τους τομείς της ανθρώπινης δραστηριότητας, όπως η επιστήμη, η τεχνολογία, η οικονομία, η υγεία και οι κοινωνικές επιστήμες. Η σωστή διαχείριση, επεξεργασία και αξιοποίηση αυτών των δεδομένων δεν αποτελεί απλώς πρόκληση, αλλά και μια μοναδική ευκαιρία για την ανάπτυξη νέας γνώσης και καινοτόμων εφαρμογών που μπορούν να επιφέρουν σημαντικές αλλαγές σε παγκόσμιο επίπεδο. Κεντρικό στοιχείο σε αυτή τη διαδικασία είναι η σημασιολογική αναπαράσταση των δεδομένων, η οποία καθιστά δυνατή την αποθήκευση και την αναζήτηση πληροφοριών όχι μόνο βάσει των δεδομένων αυτών, αλλά και των σχέσεων και του νοήματος που φέρουν.

Το RDF (Resource Description Framework) έχει καθιερωθεί ως το πιο διαδεδομένο πρότυπο για τη σημασιολογική αναπαράσταση δεδομένων, επιτρέποντας τη δημιουργία γραφημάτων γνώσης που συνδέουν διάφορες πληροφορίες με τρόπο που διευκολύνει την ανακάλυψη νέας γνώσης. Παράλληλα, η γλώσσα SPARQL (SPARQL Protocol and RDF Query Language) αποτελεί το βασικό εργαλείο που χρησιμοποιείται για την αναζήτηση και εξαγωγή πληροφοριών από βάσεις δεδομένων RDF. Μέσω της SPARQL, οι χρήστες μπορούν να διατυπώνουν ερωτήματα για να εξετάσουν τις σχέσεις μεταξύ των δεδομένων και να ανακτήσουν πολύπλοκες πληροφορίες με ακρίβεια και ευχέρεια.

Ωστόσο, η σύνταξη ερωτημάτων SPARQL απαιτεί συνήθως προγραμματιστικές γνώσεις και βαθιά κατανόηση της δομής των δεδομένων, κάτι που περιορίζει τη χρήση της από χρήστες που δεν έχουν τεχνική κατάρτιση. Εδώ έρχεται η ανάγκη για εργαλεία "query builders", τα οποία επιτρέπουν στους χρήστες να δημιουργούν ερωτήματα χωρίς να χρειάζονται εξειδικευμένες γνώσεις προγραμματισμού. Τα εργαλεία αυτά προσφέρουν φιλικές και κατανοητές διεπαφές, μέσω των οποίων οι χρήστες μπορούν να δημιουργούν τα ερωτήματά τους με χρήση απλών γραφικών εργαλείων, όπως πίνακες, φόρμες ή ακόμη και φυσική γλώσσα. Έτσι, η δημιουργία ερωτημάτων γίνεται πιο προσιτή σε ένα ευρύτερο κοινό, από ερευνητές και φοιτητές μέχρι επαγγελματίες του τομέα των δεδομένων και των επιχειρήσεων.

Η συνεισφορά της τρέχουσας διπλωματικής σε αυτό το πλαίσιο, ήταν η ανάπτυξη ενός εργαλείου που διευκολύνει τη σύνταξη και εκτέλεση ερωτημάτων SPARQL, το οποίο επιτρέπει στους χρήστες να αλληλεπιδρούν με τα δεδομένα RDF με τρόπο απλό και κατανοητό. Η διεπαφή του εργαλείου έχει σχεδιαστεί με τέτοιον τρόπο ώστε να μπορεί να χρησιμοποιηθεί χωρίς τεχνικές

γνώσεις, προσφέροντας ταυτόχρονα ισχυρές δυνατότητες αναζήτησης και εξαγωγής δεδομένων. Ένα από τα πιο σημαντικά πλεονεκτήματα αυτής της προσέγγισης είναι η δυνατότητα χρήσης έτοιμων λιστών και γραφικού περιβάλλοντος, που διευκολύνει ακόμα περισσότερο την επικοινωνία του χρήστη με το σύστημα.

Η συνεισφορά αυτή, εκτός από το ότι διευκολύνει τη ζωή των χρηστών, προάγει επίσης την ευρύτερη πρόσβαση στη γνώση. Η πρόσβαση στη σημασιολογική ανάλυση δεδομένων και στην τεχνολογία RDF γίνεται πιο προσιτή και κατανοητή για ένα ευρύτερο κοινό, ανεξαρτήτως τεχνικών γνώσεων ή επαγγελματικής εξειδίκευσης. Αυτή η εξέλιξη είναι ιδιαίτερα σημαντική, καθώς ανοίγει νέους ορίζοντες για ερευνητές, φοιτητές, επιχειρηματίες και αναλυτές δεδομένων που μέχρι σήμερα δεν είχαν άμεση πρόσβαση σε αυτά τα εργαλεία. Επιπλέον, με την εξέλιξη αυτών των εργαλείων, αναμένεται ότι η χρήση τους θα γίνει ακόμα πιο αποδοτική, ευέλικτη και φιλική προς τον χρήστη, συμβάλλοντας έτσι στην ευρύτερη χρήση της τεχνολογίας RDF και στη βελτίωση της ανάλυσης δεδομένων στον σύγχρονο κόσμο.

# Κεφάλαιο 2: Υφιστάμενα Εργαλεία

Στο πλαίσιο αυτής της διπλωματικής εργασίας, πραγματοποιήθηκε μια ανασκόπηση και ανάλυση υφιστάμενων εργαλείων που έχουν σχεδιαστεί για να διευκολύνουν τη διαχείριση και τη χρήση της γλώσσας SPARQL από χρήστες με διαφορετικά επίπεδα γνώσης και τεχνολογικής εμπειρίας. Ιδιαίτερη έμφαση δόθηκε στα εργαλεία SPARQL Builder, Sparnatural και Sparklis, τα οποία ακολουθούν διαφορετικές προσεγγίσεις για την αντιμετώπιση της πολυπλοκότητας που συνδέεται με τη σύνθεση και εκτέλεση ερωτημάτων SPARQL.

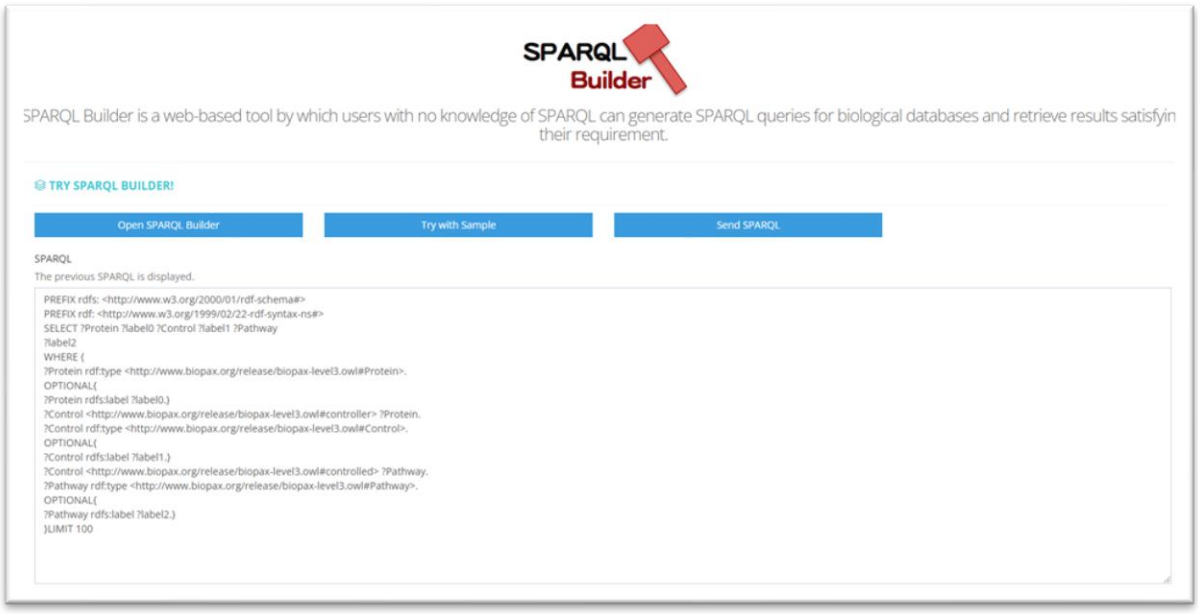
## 2.1 SPARQL Builder

Το **SPARQL Builder**[1] είναι ένα εργαλείο που έχει αναπτυχθεί με στόχο να διευκολύνει τη σύνταξη και την εκτέλεση ερωτημάτων SPARQL, ειδικά για χρήστες που δεν διαθέτουν εξειδικευμένες γνώσεις προγραμματισμού ή τη σύνταξη της γλώσσας SPARQL. Η βασική του φιλοσοφία είναι να προσφέρει μια γραφική διεπαφή που επιτρέπει στους χρήστες να δημιουργούν

[1] https://www.sparqlbuilder.org/

ερωτήματα μέσω μιας απλής, φιλικής προς τον χρήστη διαδικασίας, η οποία αφαιρεί την ανάγκη για χειροκίνητη συγγραφή σύνθετων εντολών.

Παρέχει ένα περιβάλλον εργασίας στο οποίο οι χρήστες μπορούν να επιλέγουν και να καθορίζουν διάφορα στοιχεία του ερωτήματος, όπως τα επιθυμητά πεδία (variables), τους όρους αναζήτησης (conditions), καθώς και τις σχέσεις μεταξύ των δεδομένων, μέσω της χρήσης γραφικών στοιχείων και επιλογών από μενού. Με αυτόν τον τρόπο, η διαδικασία δημιουργίας ερωτημάτων SPARQL γίνεται πιο προσβάσιμη, καθώς οι χρήστες δεν χρειάζεται να ανησυχούν για την ακριβή σύνταξη ή τη σωστή τοποθέτηση των εντολών.




Εικόνα 1. SPARQLBuilder

Η διεπαφή του SPARQL Builder είναι πολύ χρήσιμη καθώς ενσωματώνει διάφορα εργαλεία, όπως αναπτυσσόμενα μενού, κουμπιά και πίνακες, επιτρέποντας στους χρήστες να επιλέξουν τα δεδομένα που θέλουν να αναζητήσουν και να τα ενσωματώσουν εύκολα στο ερώτημά τους, χωρίς να χρειάζεται να γράψουν καθόλου κώδικα. Αυτό βοηθάει ιδιαίτερα όσους δεν είναι εξοικειωμένοι με τη γλώσσα SPARQL και την επεξεργασία δεδομένων RDF, καθώς μειώνει τη δυσκολία στην κατανόηση και την εφαρμογή αυτών των τεχνολογιών.

Τα βασικά πλεονεκτήματα του SPARQL Builder είναι τα εξής:

1. **Φιλικό προς τον χρήστη περιβάλλον**: Η γραφική διεπαφή κάνει τη διαδικασία δημιουργίας ερωτημάτων πιο προσιτή, ακόμη και για εκείνους που δεν έχουν προηγούμενη εμπειρία με τη SPARQL ή την επεξεργασία δεδομένων RDF.

2. **Μείωση λαθών σύνταξης**: Το εργαλείο περιορίζει σημαντικά τα λάθη κατά τη σύνταξη των ερωτημάτων, καθώς παρέχει προκαθορισμένα πεδία και επιλογές, εξασφαλίζοντας ότι το τελικό ερώτημα είναι σωστό συντακτικά.
3. **Εξοικονόμηση χρόνου**: Η διαδικασία σύνταξης ερωτημάτων γίνεται πιο γρήγορη, καθώς οι χρήστες δεν χρειάζεται να γράφουν τις εντολές SPARQL χειροκίνητα ή να ανατρέχουν σε τεκμηρίωση για βοήθεια.
4. **Προσαρμοστικότητα σε διάφορους χρήστες**: Το εργαλείο είναι κατάλληλο για χρήστες με διάφορα επίπεδα εμπειρίας, είτε είναι αρχάριοι είτε πιο έμπειροι χρήστες που θέλουν να διευκολύνουν τη διαδικασία σύνταξης ερωτημάτων.

Ωστόσο, παρά τα πλεονεκτήματά του, το SPARQL Builder έχει και κάποια περιορισμένα σημεία. Ειδικότερα, μπορεί να μην καλύπτει όλα τα πιο προηγμένα χαρακτηριστικά της γλώσσας SPARQL. Χρήστες με πιο εξειδικευμένες ανάγκες ή εκείνοι που έχουν καλή γνώση της γλώσσας μπορεί να βρουν τη γραφική διεπαφή περιοριστική, καθώς δεν προσφέρει την ίδια ευχέρεια που δίνει η χειροκίνητη σύνταξη κώδικα SPARQL. Επίσης, κάποιες προηγμένες λειτουργίες, όπως οι σύνθετες υποερωτήσεις ή οι εξατομικευμένες συνθήκες φιλτραρίσματος, ίσως να μην υποστηρίζονται πάντα, ανάλογα με την έκδοση του εργαλείου.

## 2.2 Sparnatural

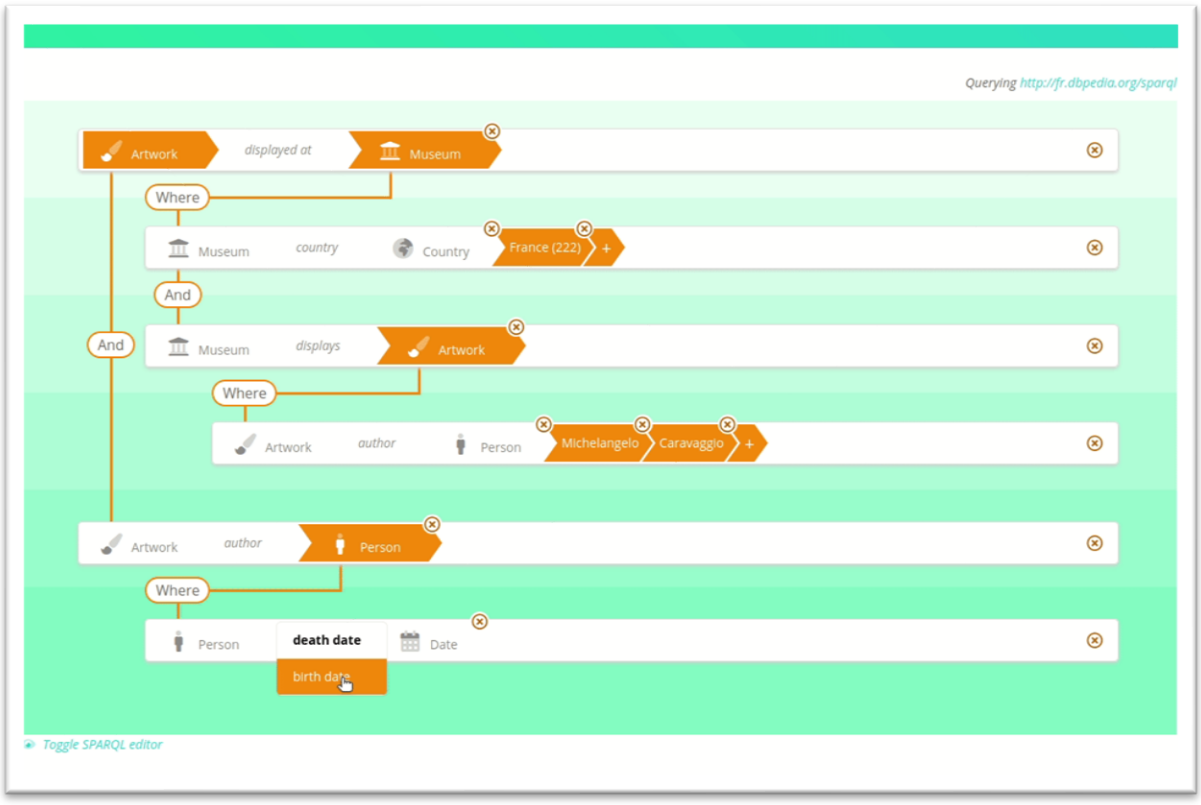
Το **Sparnatural**[2] είναι ένα καινοτόμο εργαλείο που έχει αναπτυχθεί για να διευκολύνει τη σύνταξη ερωτημάτων SPARQL, χρησιμοποιώντας φυσική γλώσσα. Η βασική του καινοτομία έγκειται στην ικανότητά του να επιτρέπει στους χρήστες να διατυπώνουν τα ερωτήματά τους με όρους καθημερινής γλώσσας, μειώνοντας την ανάγκη για εξειδικευμένες γνώσεις της γλώσσας SPARQL. Αυτό το καθιστά ιδιαίτερα χρήσιμο για χρήστες που δεν έχουν τεχνική κατάρτιση, αλλά θέλουν να εκμεταλλευτούν τα δεδομένα που υπάρχουν σε βάσεις RDF.

### Χαρακτηριστικά και Λειτουργίες

Το **Sparnatural** αξιοποιεί τεχνικές επεξεργασίας φυσικής γλώσσας (Natural Language Processing - NLP) για να «κατανοήσει» τις ερωτήσεις που διατυπώνονται από τους χρήστες και να τις μεταφράσει σε έγκυρα ερωτήματα SPARQL. Ο χρήστης μπορεί να πληκτρολογήσει ή να εισάγει την ερώτησή του σε μια μορφή που μοιάζει με φυσική γλώσσα (π.χ. "Ποιες είναι οι χώρες με πληθυσμό μεγαλύτερο από 10 εκατομμύρια;"), και το εργαλείο αναλαμβάνει να μετατρέψει αυτή

[2] https://sparnatural.eu/

την εισαγωγή σε ένα ερώτημα SPARQL που μπορεί να εκτελεστεί στην αντίστοιχη βάση δεδομένων RDF.




Εικόνα 2. **Sparnatural**

Η δυνατότητα αυτή βασίζεται στην αναγνώριση και ανάλυση των εννοιών, των σχέσεων και των εννοιών που περιλαμβάνονται στην εισερχόμενη ερώτηση, επιτρέποντας στο εργαλείο να κατανοήσει την πρόθεση του χρήστη και να την αντιστοιχίσει σε συγκεκριμένες εντολές της γλώσσας SPARQL.

**Πλεονεκτήματα**

1. **Εύχρηστο για μη τεχνικούς χρήστες**: Η βασική δύναμη του Sparnatural είναι ότι απομακρύνει την ανάγκη για γνώσεις της σύνταξης της γλώσσας SPARQL, επιτρέποντας στους χρήστες να διατυπώσουν ερωτήματα με φυσική γλώσσα. Αυτό διευκολύνει την πρόσβαση σε μη τεχνικούς χρήστες, όπως ερευνητές, επιχειρηματίες, και άλλους που θέλουν να αλληλεπιδράσουν με δεδομένα RDF χωρίς να έχουν προγραμματιστικές γνώσεις.
2. **Φυσική αλληλεπίδραση**: Η δυνατότητα χρήσης καθημερινής γλώσσας για τη δημιουργία ερωτημάτων καθιστά τη διαδικασία πιο φυσική και οικεία, ενισχύοντας την εμπειρία χρήστη και μειώνοντας το φράγμα της εισόδου στη χρήση τεχνολογιών RDF και SPARQL.
3. **Εξοικονόμηση χρόνου**: Οι χρήστες δεν χρειάζεται να ξοδεύουν χρόνο στη μάθηση της σύνταξης SPARQL, γεγονός που τους επιτρέπει να επικεντρωθούν στην ανάλυση και τη διερεύνηση των δεδομένων. Η διαδικασία είναι πιο άμεση και γρήγορη, αφού οι ερωτήσεις διατυπώνονται με τον ίδιο τρόπο που κάποιος θα ρωτούσε άλλους ανθρώπους.
4. **Αντιστοίχιση σε σύνθετα ερωτήματα**: Παρά την απλότητα της φυσικής γλώσσας, το Sparnatural μπορεί να επεξεργαστεί σύνθετα ερωτήματα που απαιτούν την αντιστοίχιση πολυάριθμων εννοιών, σχέσεων και συνθηκών, επιτρέποντας στους χρήστες να

διατυπώνουν περίπλοκες αναζητήσεις χωρίς να ανησυχούν για τη σύνθεση της γλώσσας SPARQL.

**Περιορισμοί**

1. **Περιορισμός στα δεδομένα RDF**: Η χρήση του Sparnatural περιορίζεται στις δυνατότητες των δεδομένων RDF. Αν το σύστημα δεδομένων δεν είναι επαρκώς οργανωμένο ή το RDF γραφικό δεν είναι σωστά διαρθρωμένο, η μετάφραση φυσικής γλώσσας σε SPARQL μπορεί να μην αποδώσει τα αναμενόμενα αποτελέσματα.

Το εργαλείο αυτό επιτυγχάνει να γεφυρώσει το χάσμα μεταξύ της τεχνικής φύσης των SPARQL ερωτημάτων και της καθημερινής γλώσσας που χρησιμοποιούν οι χρήστες. Χρησιμοποιεί έναν συνδυασμό γραφικών διεπαφών και μηχανισμών κατανόησης φυσικής γλώσσας για να παρέχει μια διαδραστική εμπειρία. Αυτή η λειτουργία είναι ιδιαίτερα χρήσιμη σε περιβάλλοντα όπου η συνεργασία μεταξύ τεχνικών και μη τεχνικών ομάδων είναι κρίσιμη.

## 2.3 Sparklis

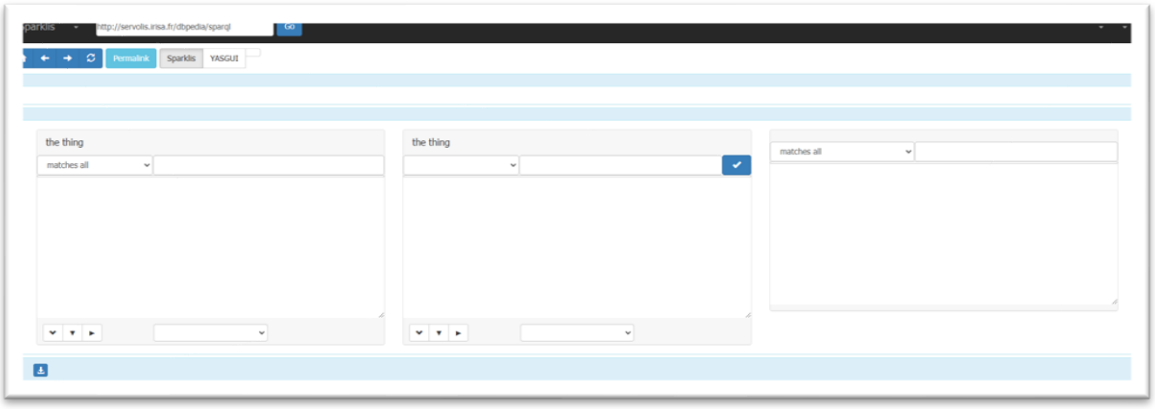
Το Sparklis[3] είναι ένα εργαλείο που βοηθά τους χρήστες να δημιουργούν, να εκτελούν και να αναλύουν ερωτήματα SPARQL με έναν πιο απλό και οπτικό τρόπο. Αντί να γράφουν τον κώδικα SPARQL, χρησιμοποιούν γραφικά στοιχεία, όπως πίνακες και κουμπιά επιλογής, για να διαμορφώσουν τα ερωτήματά τους. Είναι ιδανικό για άτομα που δεν γνωρίζουν τη σύνταξη SPARQL, αλλά θέλουν να δουλέψουν με δεδομένα RDF.

Η διεπαφή του Sparklis είναι οπτική και επιτρέπει στους χρήστες να δημιουργούν ερωτήματα με drag-and-drop επιλογές, κάνοντάς το πιο εύκολο, ακόμα και για αρχάριους. Επιπλέον, υποστηρίζει τη δημιουργία γραφημάτων γνώσης που βοηθούν στην κατανόηση των σχέσεων μεταξύ των δεδομένων. Οι χρήστες μπορούν να εξερευνούν τα δεδομένα πιο εύκολα και να εντοπίζουν μοτίβα ή σχέσεις που δεν είναι πάντα προφανείς με την απλή ανάγνωση.

Η διαδικασία δημιουργίας και εκτέλεσης των ερωτημάτων γίνεται χωρίς να χρειάζεται να γράψουν κώδικα, καθώς το εργαλείο τα εκτελεί αυτόματα και εμφανίζει τα αποτελέσματα άμεσα. Ενώ είναι πολύ φιλικό και προσβάσιμο σε όλους τους χρήστες, ακόμα και σε εκείνους χωρίς προηγούμενη εμπειρία, είναι επίσης χρήσιμο και για πιο έμπειρους χρήστες που θέλουν έναν γρήγορο και εύκολο τρόπο να δημιουργούν και να εκτελούν ερωτήματα.

Συνολικά, το Sparklis είναι ένα εργαλείο που επιτρέπει στους χρήστες να δουλέψουν με δεδομένα RDF χωρίς να χρειάζεται να κατανοήσουν την πλήρη γλώσσα SPARQL, καθιστώντας τη διαδικασία δημιουργίας ερωτημάτων πιο απλή και προσβάσιμη σε όλους.

[3] http://www.irisa.fr/LIS/ferre/sparklis/




Εικόνα 3. **Sparklis**

Παρά τις δυνατότητές τους, αυτά τα εργαλεία παρουσιάζουν ορισμένους περιορισμούς. Δεν υποστηρίζουν πάντα τη σύνθεση **federated queries**, δηλαδή ερωτημάτων που συνδυάζουν δεδομένα από πολλαπλά datasets ή end points. Επίσης, η διαχείριση και η εξερεύνηση γραφημάτων συχνά παραμένουν περιορισμένες.

Η παρούσα εργασία επικεντρώνεται στην ανάπτυξη μιας νέας εφαρμογής που υπερβαίνει αυτούς τους περιορισμούς, εισάγοντας νέες δυνατότητες για τη σύνθεση και την εκτέλεση ερωτημάτων SPARQL:

1. **Υποστήριξη πολλαπλών end points**: Ο χρήστης μπορεί να επιλέξει το end point της προτίμησής του για την εκτέλεση του ερωτήματος.
2. **Εξερεύνηση datasets και γραφημάτων**:
    - Με την επιλογή ενός end point, η εφαρμογή εμφανίζει όλα τα διαθέσιμα datasets πέραν του default.
    - Ο χρήστης μπορεί να επιλέξει dataset και να δει όλους τους διαθέσιμους γραφήματα (graphs) που περιλαμβάνονται σε αυτό.
    - Η εφαρμογή εμφανίζει όλα τα subjects ενός γραφήματος, και στη συνέχεια όλα τα διαθέσιμα predicates για το επιλεγμένο subject, διευκολύνοντας την προοδευτική σύνθεση του ερωτήματος.
3. **Υποστήριξη federated queries**: Ο χρήστης έχει τη δυνατότητα να συνδυάζει δεδομένα από πολλαπλά datasets και end points, δημιουργώντας σύνθετα ερωτήματα με στόχο την ενοποίηση πληροφοριών.
4. **Εκτέλεση ερωτημάτων σε πραγματικό χρόνο**: Τα ερωτήματα μπορούν να εκτελεστούν απευθείας στο επιλεγμένο end point, επιτρέποντας την άμεση αξιολόγηση των αποτελεσμάτων.

Η νέα εφαρμογή δεν περιορίζεται μόνο στη σύνθεση ερωτημάτων, αλλά ενισχύει τη δυνατότητα ανάλυσης δεδομένων μέσω της χρήσης δυναμικής διεπαφής και αυξημένης ευελιξίας. Αυτή η λειτουργικότητα καθιστά την εφαρμογή χρήσιμη τόσο για αρχάριους όσο και για προχωρημένους χρήστες, προσφέροντας νέες προοπτικές στη διαχείριση και τη σύνθεση δεδομένων.

# Κεφάλαιο 3: Σύνταξη Ερωτημάτων SPARQL

Η **SPARQL** (SPARQL Protocol and RDF Query Language) είναι η γλώσσα ερωτημάτων που αναπτύχθηκε από το W3C για την ανάκτηση και διαχείριση δεδομένων σε μορφή RDF (Resource Description Framework). Η SPARQL επιτρέπει την εκτέλεση πολύπλοκων ερωτημάτων σε δεδομένα που είναι αποθηκευμένα σε τυποποιημένα γραφήματα γνώσης, καθιστώντας την έναν θεμελιώδη μηχανισμό για εφαρμογές που βασίζονται στο **Σημασιολογικό Διαδίκτυο** (Semantic Web).

Η χρήση της SPARQL δεν περιορίζεται μόνο σε ακαδημαϊκές ή ερευνητικές εφαρμογές, αλλά εκτείνεται και σε εμπορικές λύσεις, όπου η αποτελεσματική διαχείριση και ανάλυση δεδομένων είναι εξίσου σημαντική. Όπως αναφέρει ο Bob DuCharme στο βιβλίο του *Learning SPARQL*, «Η κατανόηση της σύνταξης SPARQL είναι το πρώτο βήμα για την εκμετάλλευση της δύναμης του Σημασιολογικού Ιστού, καθιστώντας τα δεδομένα πιο εύκολα προσβάσιμα και επεξεργάσιμα από υπολογιστικά συστήματα».

## 3.1 Σύνθεση Ερωτημάτων SPARQL

Ένα ερώτημα SPARQL αποτελείται από τρία βασικά μέρη:

1. **Προθέματα (Prefixes)**: Αυτά τα συντομεύματα για URI (Uniform Resource Identifier) επιτρέπουν την αναφορά σε οντότητες με πιο σύντομο και κατανοητό τρόπο. Για παράδειγμα, το πρόθεμα:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
```

   Χρησιμοποιείται συχνά για οντότητες που σχετίζονται με πρόσωπα και κοινωνικές σχέσεις.

2. **Μοτίβο Ερωτήματος (Query Pattern)**: Το κύριο σώμα του ερωτήματος αποτελείται από μία τριπλέτα προτύπου (subject-predicate-object) που καθορίζει τις σχέσεις που πρέπει να πληρούν τα δεδομένα. Για παράδειγμα:

```
SELECT ?name WHERE { ?person foaf:name ?name . }
```

   Αυτό το ερώτημα αναζητά τα ονόματα των ατόμων, όπου το ?person αντιπροσωπεύει τις οντότητες και το ?name τα ονόματα τους.

3. **Τροποποιητές (Modifiers)**: Αυτά χρησιμοποιούνται για τη διαμόρφωση των αποτελεσμάτων του ερωτήματος, όπως η ταξινόμηση (ORDER BY), η ομαδοποίηση (GROUP BY) ή ο περιορισμός των αποτελεσμάτων (FILTER). Για παράδειγμα:

```
SELECT ?name WHERE { ?person foaf:name ?name . } ORDER BY ?name
```

Στην εργασία *Learning SPARQL*, ο Bob DuCharme επισημαίνει ότι «Η χρήση προθεμάτων μειώνει σημαντικά το μήκος των URI, διευκολύνοντας έτσι την κατανόηση και επεξεργασία των δεδομένων». Ο ίδιος τονίζει επίσης ότι τα μοτίβα ερωτημάτων προσφέρουν μεγάλη ευχέρεια στην αναζήτηση συγκεκριμένων σχέσεων σε δεδομένα RDF.

## 3.2 Είδη Ερωτημάτων SPARQL

Τα ερωτήματα SPARQL μπορούν να λάβουν διάφορες μορφές, ανάλογα με τον σκοπό της αναζήτησης:

- **SELECT**: Επιστρέφει συγκεκριμένα δεδομένα σε μορφή πινάκων. Είναι το πιο συνηθισμένο είδος ερωτήματος και χρησιμοποιείται για την ανάκτηση πληροφοριών από τις βάσεις δεδομένων RDF.
- **ASK**: Ελέγχει αν υπάρχει τουλάχιστον μία καταγραφή που ικανοποιεί το ερώτημα. Επιστρέφει μια boolean τιμή (true ή false).
- **CONSTRUCT**: Δημιουργεί νέα RDF γραφήματα ως αποτέλεσμα του ερωτήματος, επιτρέποντας τη δόμηση νέων συνόλων δεδομένων από την πηγή.
- **DESCRIBE**: Επιστρέφει μια περιγραφή των οντοτήτων που συμμετέχουν στο ερώτημα, παρέχοντας περισσότερες πληροφορίες για αυτές.

## 3.3 Χρησιμοποίηση Φίλτρων και Σύνθετων Ερωτημάτων

Η χρήση φίλτρων στη SPARQL είναι καθοριστική για την εξειδίκευση των αναζητήσεων και τη διαχείριση των δεδομένων. Όπως αναφέρουν οι Jorge Pérez, Marcelo Arenas και Claudio Gutiérrez στην εργασία τους *Semantics and Complexity of SPARQL*, «η βασική δομή ενός SPARQL ερωτήματος στηρίζεται στις τριπλέτες, οι οποίες είναι οι θεμελιώδεις μονάδες ενός μοτίβου. Μπορούν να συνδυαστούν μέσω λογικών τελεστών για τη δημιουργία πιο σύνθετων μοτίβων». Για παράδειγμα, το ακόλουθο ερώτημα χρησιμοποιεί φίλτρο για να περιορίσει τα αποτελέσματα στην αγγλική γλώσσα:

```
SELECT ?name

WHERE { ?person foaf:name ?name .

FILTER (lang(?name) = "en") }
```

Αυτή η δυνατότητα επιτρέπει την πιο ακριβή αναζήτηση και επεξεργασία δεδομένων, ενισχύοντας τη δύναμη της SPARQL στη διαχείριση των σημασιολογικών δεδομένων.

## 3.4 Βελτιστοποίηση και Παράλληλη Επεξεργασία Ερωτημάτων

Η απόδοση των ερωτημάτων SPARQL μπορεί να βελτιωθεί μέσω διάφορων τεχνικών βελτιστοποίησης, όπως η παράλληλη επεξεργασία, η ομαδοποίηση και η εκτέλεση κατανεμημένων συστημάτων. Στην εργασία *SPARQL Query Optimization on Top of DHTs* οι Panagiotis Bouros, Stavros Athanassopoulos και Nikos Mamoulis επισημαίνουν ότι «η χρήση κατανεμημένων συστημάτων για την εκτέλεση ερωτημάτων SPARQL μπορεί να βελτιώσει την απόδοση μέσω παράλληλης επεξεργασίας». Αυτό είναι ιδιαίτερα χρήσιμο όταν οι βάσεις δεδομένων RDF είναι μεγάλες και η αναζήτηση δεδομένων απαιτεί σημαντικούς υπολογιστικούς πόρους.

Επιπλέον, η ανάπτυξη μεθόδων για την **παράλληλη επεξεργασία** ερωτημάτων, όπως αναφέρεται στην εργασία *SPARQL Query Parallel Processing: A Survey* των Zhang και Gruenwald, είναι κρίσιμη για την υποστήριξη μεγάλων συστημάτων δεδομένων, επιτρέποντας την ταχεία και αποδοτική επεξεργασία σε κατανεμημένα περιβάλλοντα.

## 3.5 Εφαρμογές και Σημασία της SPARQL

Η **SPARQL** βρίσκεται στο επίκεντρο του οικοσυστήματος των ανοικτών συνδεδεμένων δεδομένων, με πλατφόρμες όπως το **DBpedia** και το **Wikidata** να βασίζονται στη SPARQL για την αναζήτηση και ανάκτηση δεδομένων πλούσιας σημασιολογίας. Η δυνατότητα σύνδεσης και ανάλυσης δεδομένων από πολλαπλές πηγές καθιστά τη SPARQL απαραίτητο εργαλείο για την εξερεύνηση του Σημασιολογικού Ιστού, προσφέροντας δυναμική και ευέλικτη διαχείριση δεδομένων. Όπως αναφέρει ο Zhang, «Η δυνατότητα ενσωμάτωσης και ανάλυσης δεδομένων σε πολλές γλώσσες είναι κρίσιμη για τη βελτίωση της προσβασιμότητας των ανοικτών δεδομένων».

Η SPARQL προσφέρει δυνατότητες που ξεπερνούν τη συμβατική διαχείριση βάσεων δεδομένων, επιτρέποντας την απρόσκοπτη σύνδεση δεδομένων από ετερογενείς πηγές. Στην πράξη, αυτό σημαίνει ότι τα δεδομένα μπορούν να ανακτηθούν και να αναλυθούν με έναν ενιαίο και αποτελεσματικό τρόπο, ανεξαρτήτως της μορφής ή της προέλευσής τους.

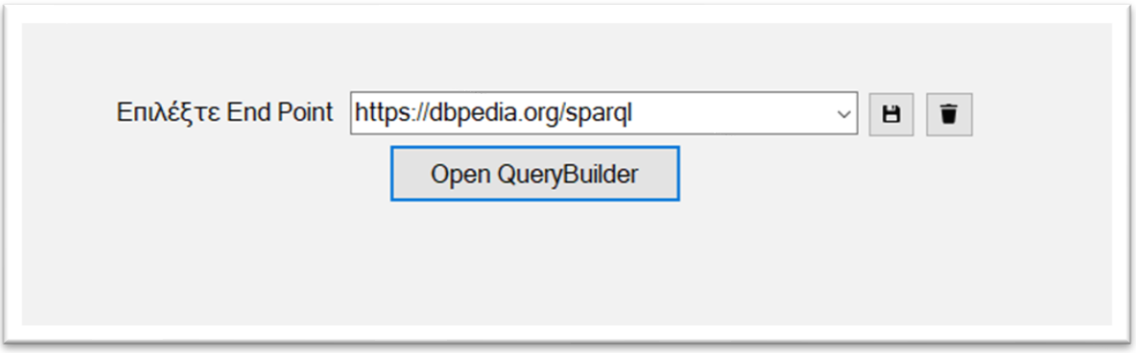
Η SPARQL αποτελεί ένα εξαιρετικά ισχυρό εργαλείο για την επεξεργασία και ανάκτηση δεδομένων RDF, με εφαρμογές που εκτείνονται από την ακαδημαϊκή έρευνα έως τις εμπορικές εφαρμογές και την ενσωμάτωση ανοικτών δεδομένων. Η κατανόηση της σύνταξης SPARQL, η χρήση της κατάλληλης βελτιστοποίησης των ερωτημάτων, καθώς και η δυνατότητα ενσωμάτωσης δεδομένων από πολλαπλές πηγές, καθιστούν τη SPARQL απαραίτητο εργαλείο για τη σύγχρονη ανάλυση δεδομένων.

# Κεφάλαιο 4: Εγχειρίδιο Εφαρμογής

Το εγχειρίδιο αυτό περιγράφει τη λειτουργία και τις δυνατότητες της εφαρμογής SPARQL Query Builder, διευκολύνοντας τους χρήστες στη διαμόρφωση και εκτέλεση ερωτημάτων SPARQL.

## 4.1 Αρχικό Μενού

Κατά την έναρξη της εφαρμογής, εμφανίζεται το **Αρχικό Μενού**, μέσω του οποίου έχετε τη δυνατότητα να επιλέξετε το **end point** στο οποίο θα εκτελούνται τα ερωτήματά σας.




Εικόνα 4. Επιλογή **end point**

### Δημιουργία Αρχείου Ρυθμίσεων

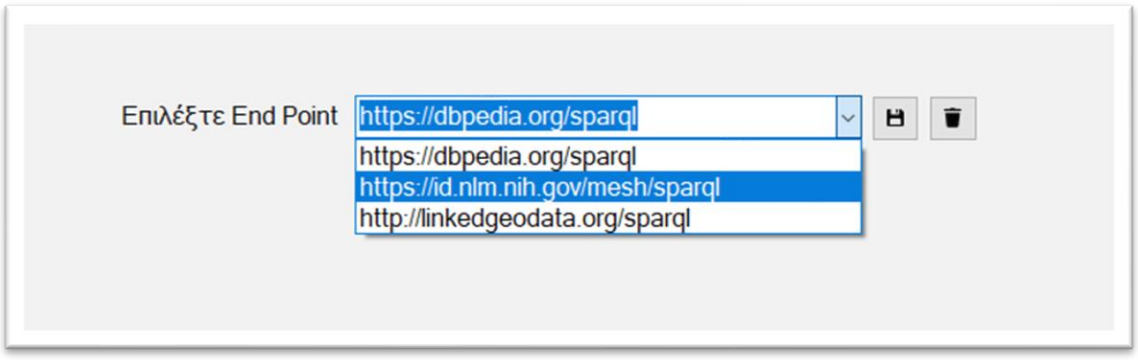
Κατά την πρώτη εκκίνηση της εφαρμογής:

- Δημιουργείται αυτόματα ένα αρχείο ρυθμίσεων (**settings**), αν δεν υπάρχει ήδη.
- Το αρχείο αυτό περιλαμβάνει βασικές ρυθμίσεις και προκαθορισμένα end points, ώστε να έχετε άμεση πρόσβαση στις αρχικές επιλογές.

### Επιλογές End Points

Στο αρχικό στάδιο:

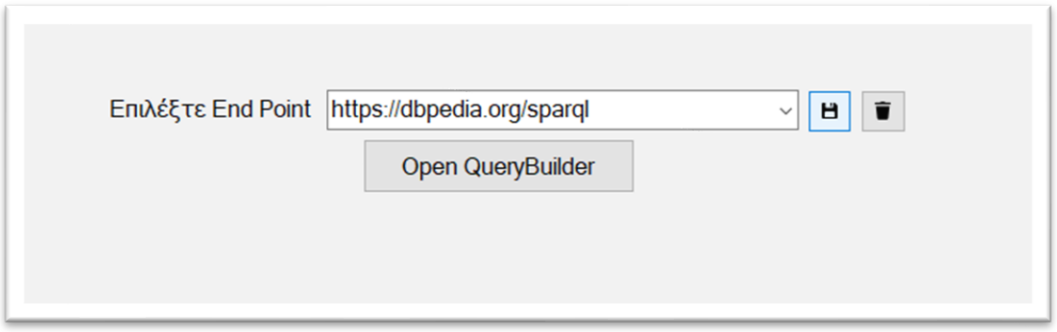
- Εμφανίζονται δύο προκαθορισμένα end points.
- Έχετε τη δυνατότητα να επιλέξετε ανάμεσα στα διαθέσιμα ή να προσθέσετε νέα end points σύμφωνα με τις ανάγκες σας.



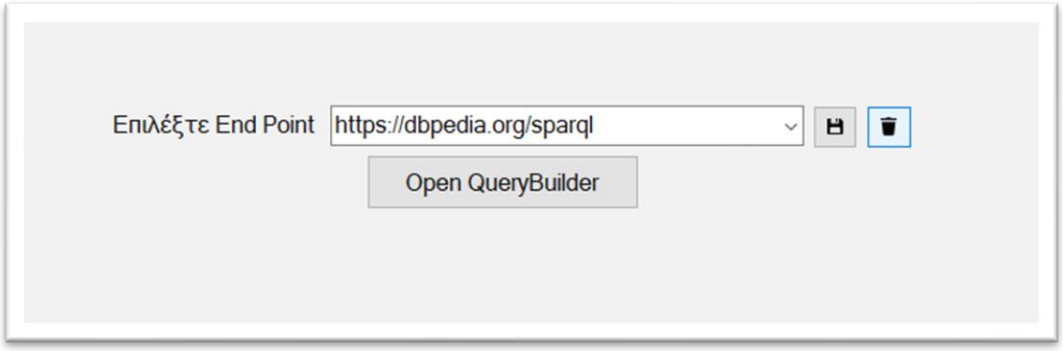

Εικόνα 5. Διαθέσιμα **end points**

## Διαχείριση End Points

1. **Προβολή Διαθέσιμων End Points**:
   - Η λίστα των αποθηκευμένων end points εμφανίζεται αυτόματα στο αρχικό μενού.
   - Περιλαμβάνει τα προκαθορισμένα end points που έχουν αποθηκευτεί στο αρχείο ρυθμίσεων.
2. **Προσθήκη Νέου End Point**:
   - Μέσα από την εφαρμογή, μπορείτε να εισάγετε νέα end points, τα οποία θα αποθηκευτούν μόνιμα στο αρχείο ρυθμίσεων.

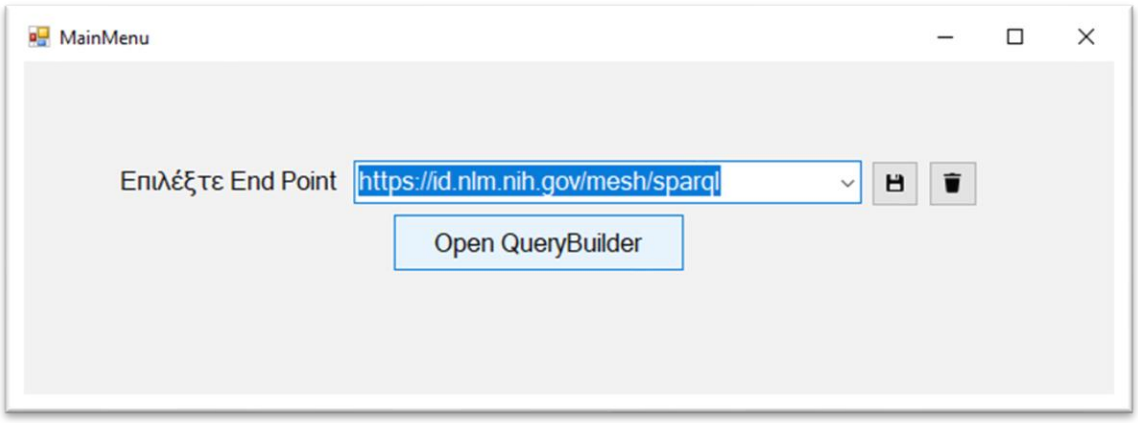



Εικόνα 6. Προσθήκη End Point




Εικόνα 7. Διαγραφή End Point

3. **Επεξεργασία Ρυθμίσεων**:
   - Το αρχείο **settings** μπορεί να επεξεργαστεί χειροκίνητα ή μέσω της εφαρμογής για να προσαρμόσετε τις ρυθμίσεις σύμφωνα με τις προτιμήσεις σας.

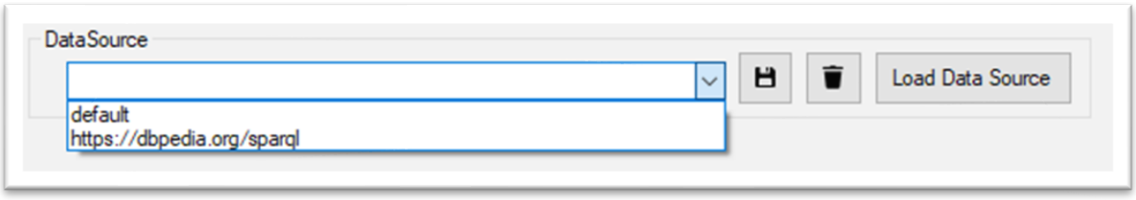



Εικόνα 8. Επιλογή End Point και άνοιγμα Query Builder για να μεταφερθούμε στο περιβάλλον όπου γράφουμε τα ερωτήματά μας

## 4.2 Επιλογή Data Source

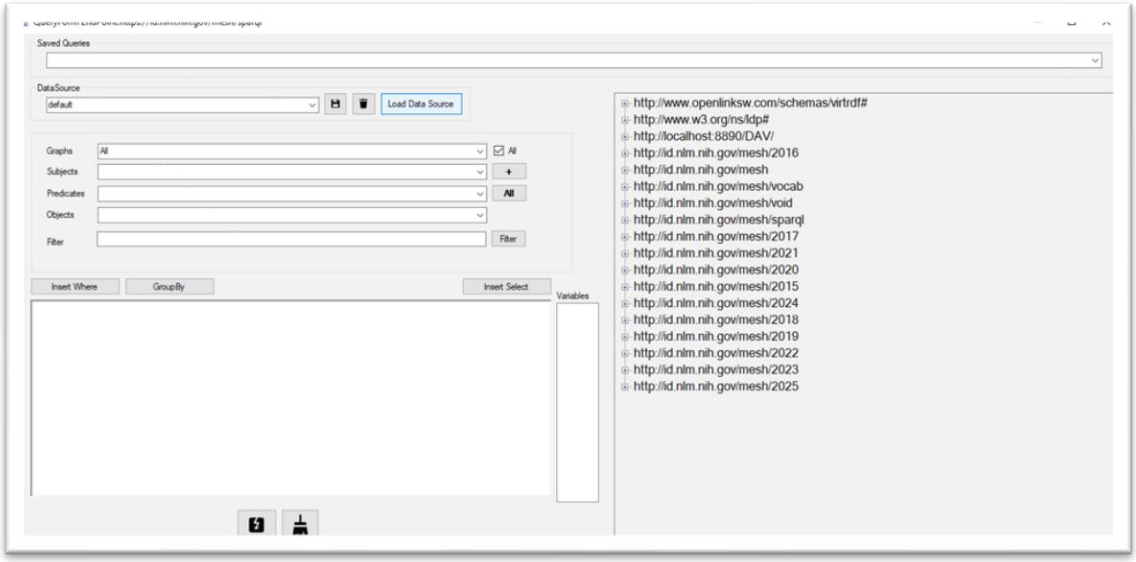
Πριν ξεκινήσετε τη σύνθεση του ερωτήματός σας:

1. **Επιλογή Data Source**:
    - Το αρχείο ρυθμίσεων σας προτείνει αποθηκευμένες επιλογές data sources.
    - Αν θέλετε να χρησιμοποιήσετε το data source του end point σας, επιλέγετε την επιλογή "Default".




Εικόνα 9. Επιλογή "Default" End Point

    - Διαφορετικά, μπορείτε να επιλέξετε ένα εξωτερικό data source για να δημιουργήσετε ένα **federated query**, εφόσον το τρέχον end point υποστηρίζει αυτή τη δυνατότητα.
2. **Φόρτωση Data Source**:
    - Πατήστε το πλήκτρο **Load Data Source** για να φορτωθούν οι γράφοι του επιλεγμένου data source.
    - Εάν η διαδικασία ολοκληρωθεί επιτυχώς, οι γράφοι εμφανίζονται στο δεξί πάνελ σε δεντρική μορφή.

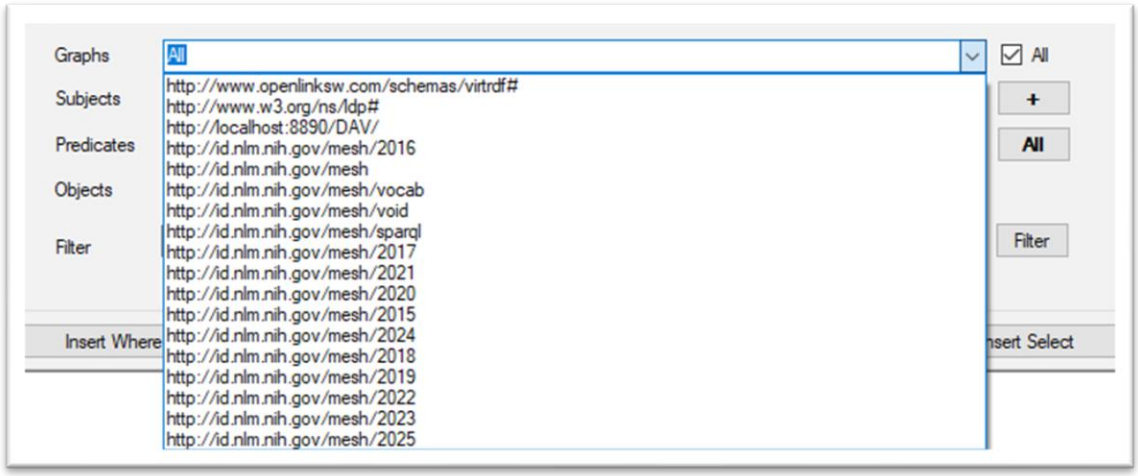



Εικόνα 10. Γράφοι σε δεντρική μορφή.

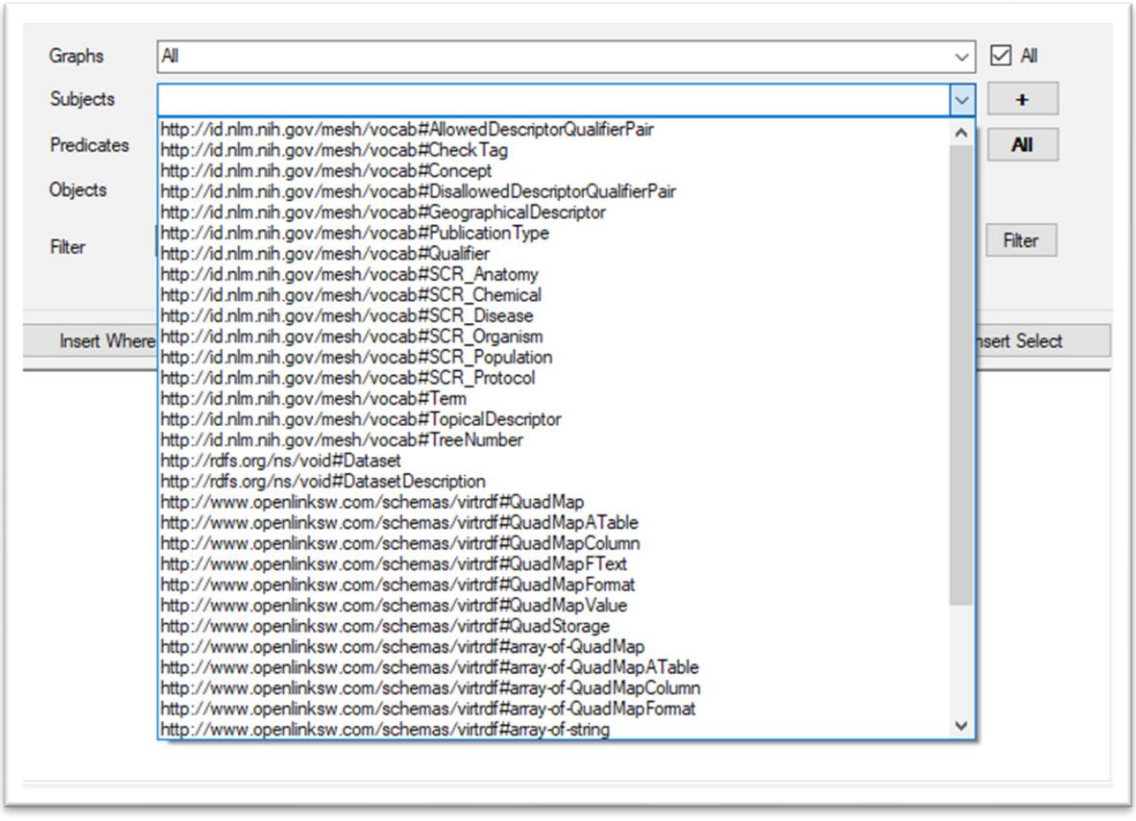
## 4.3 Πάνελ Ιεραρχίας

Μετά τη φόρτωση του data source:

- Υπάρχει ένα πάνελ με λίστες για **Γράφους**, **Subjects**, **Predicates** και **Objects**.
- Οι λίστες εμφανίζονται σε ιεραρχική διάταξη:
    1. Επιλέγοντας έναν γράφο, μπορείτε να δείτε τα διαθέσιμα subjects.
    2. Επιλέγοντας ένα subject, η λίστα γεμίζει με τα αντίστοιχα predicates (μέσω σχετικού ερωτήματος στο end point).
    3. Αν θέλετε να εκτελέσετε το ερώτημα σε ολόκληρο το data source, μπορείτε να επιλέξετε την επιλογή "All".

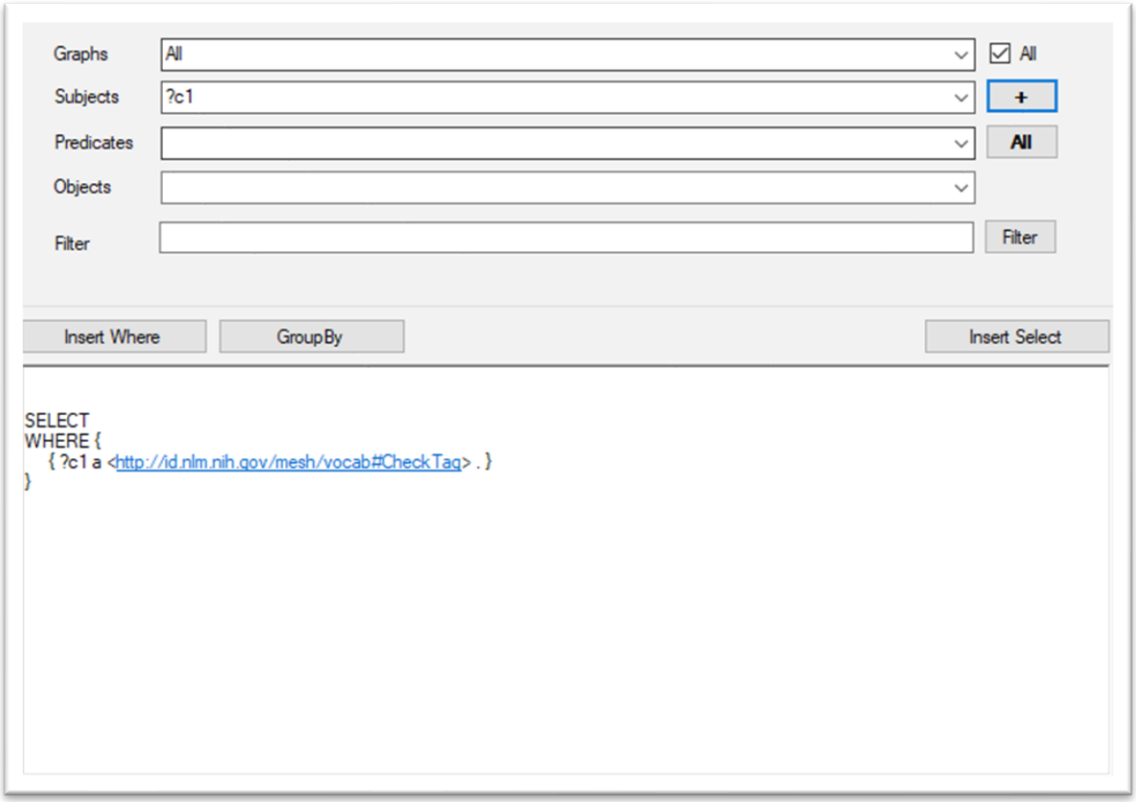



Εικόνα 11. Επιλογή Γράφου.



Εικόνα 12. Επιλογή Subject.

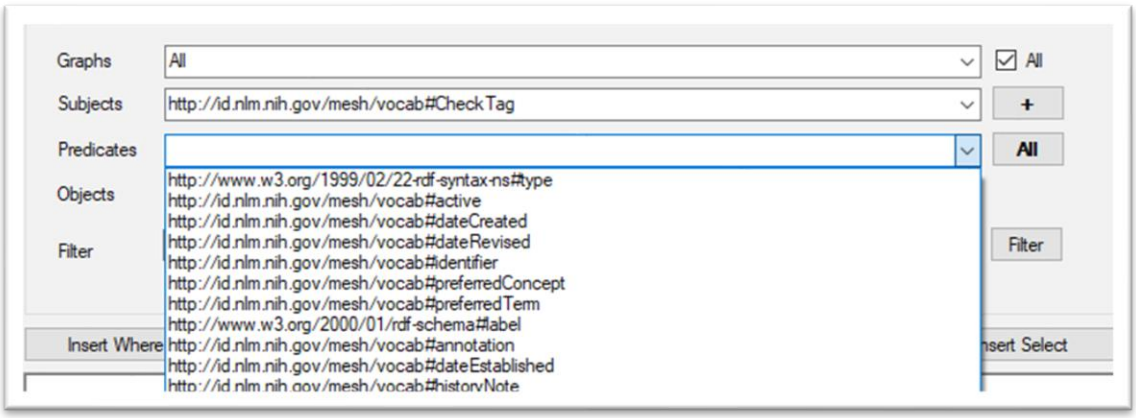
Για να δομήσουμε ένα ερώτημα που αφορά σε ένα Subject πατάμε το πλήκτρο «+».




Εικόνα 13. Ερώτημα που αφορά σε ένα Subject

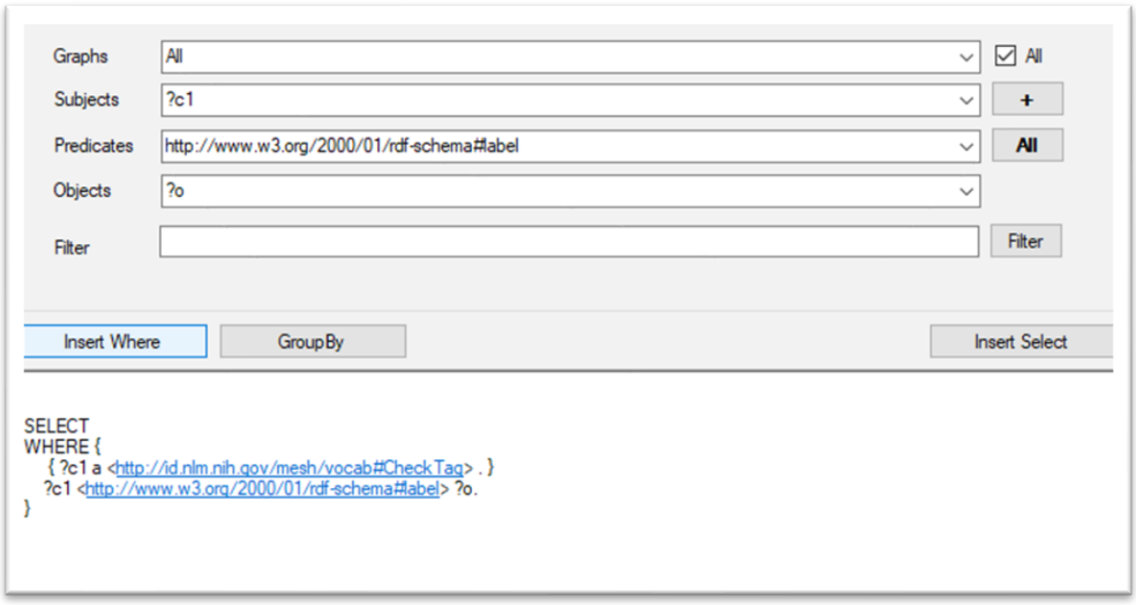
Και θα εισαχθεί στο ερώτημά μας μια δήλωση του τύπου

```
?c1 a <http://id.nlm.nih.gov/mesh/vocab#CheckTag> .
```



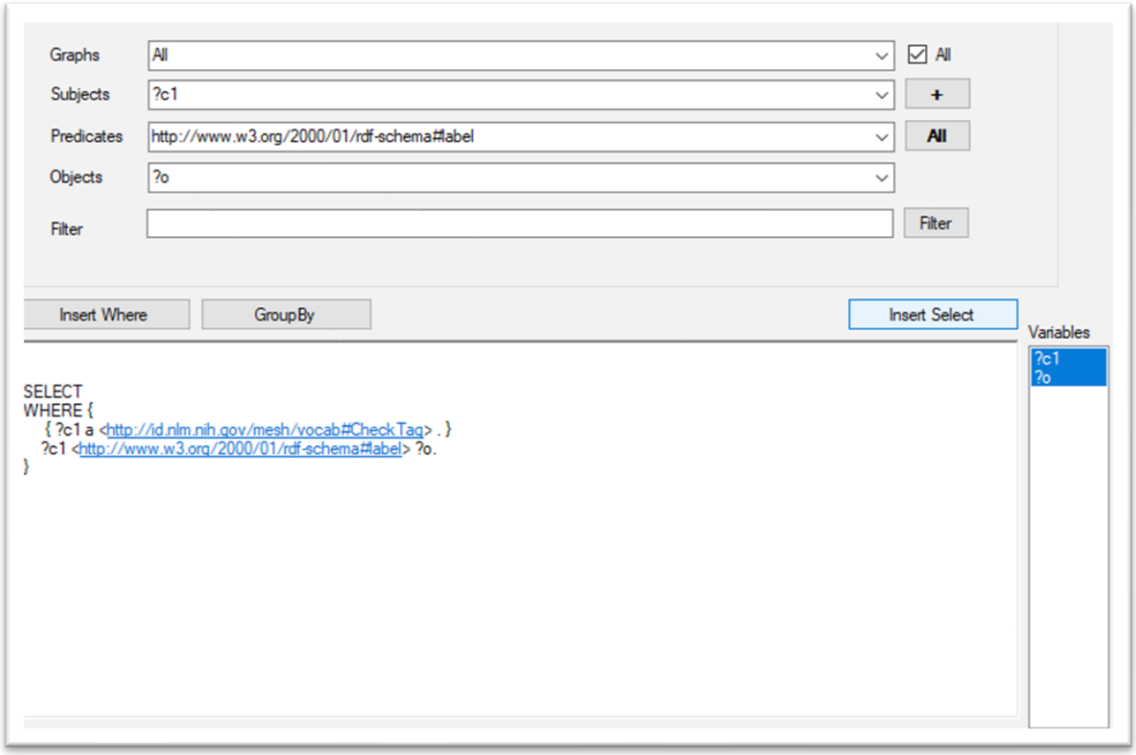

Εικόνα 14. Επιλογή Predicate.

Στη συνέχεια εφόσον πλέον έχουμε τη μεταβλητή `?c1` που είναι τύπου CheckTag μπορούμε να εισάγουμε επιπλέον δηλώσεις. Πχ αν ζητάμε όλα τα Objects για ένα συγκεκριμένο Predicate:



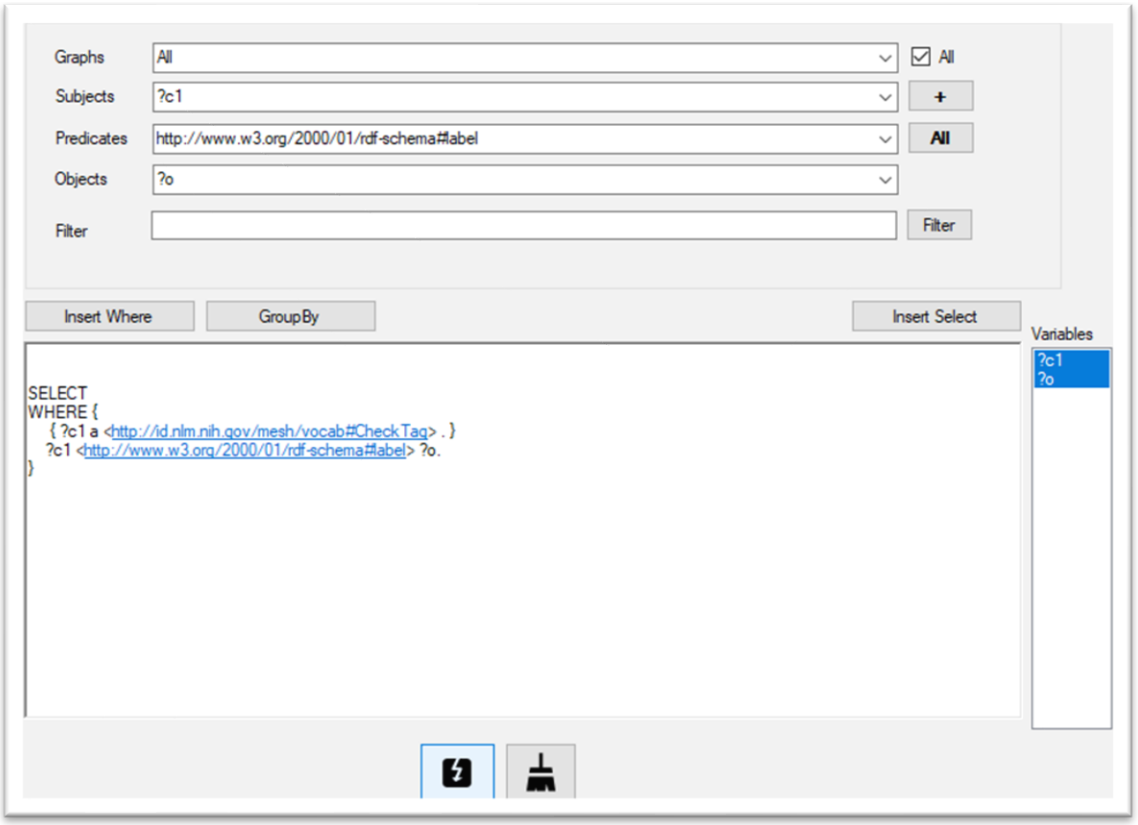

Εικόνα 15. Εισαγωγή επιπλέον δηλώσεων

Στη λίστα "Variables" εμφανίζονται οι μεταβλητές που έχουμε χρησιμοποιήσει μέχρι στιγμής στα ερωτήματά μας. Από εκεί, μπορούμε να επιλέξουμε ποιες θέλουμε να εισαχθούν στο τμήμα "Select" του ερωτήματός μας. Πατώντας το πλήκτρο "Insert Select", οι επιλεγμένες μεταβλητές θα προστεθούν στο τμήμα "Select" του ερωτήματός μας.




Εικόνα 16. Επιλογή και προσθήκη μετβλητών

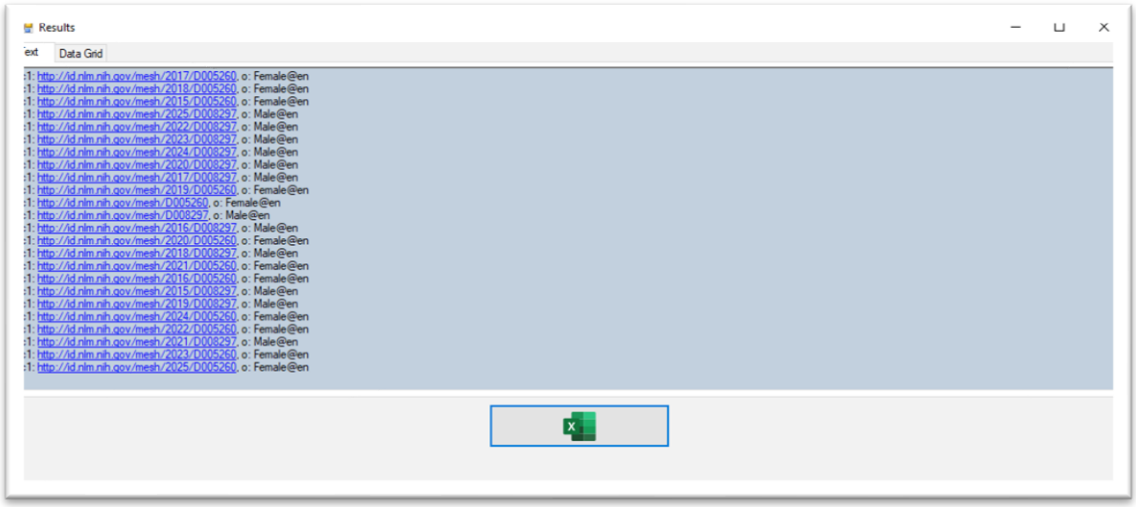
Στη συνέχεια, εκτελούμε το ερώτημα, πατώντας το κατάλληλο κουμπί για να λάβουμε τα αποτελέσματα.




Εικόνα 17. Εκτέλεση ερωτήματος

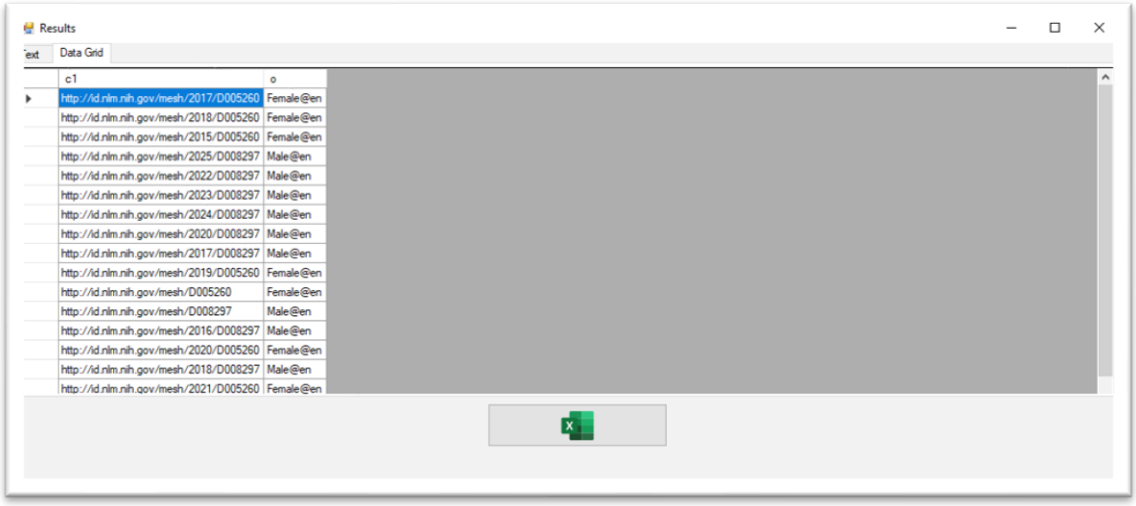
## 4.4 Εμφάνιση Αποτελεσμάτων

Μετά την εκτέλεση και προκειμένου να δούμε τα αποτελέσματα, ανοίγει η φόρμα "Results", η οποία εμφανίζει τα αποτελέσματα του ερωτήματος σε μορφή κειμένου.



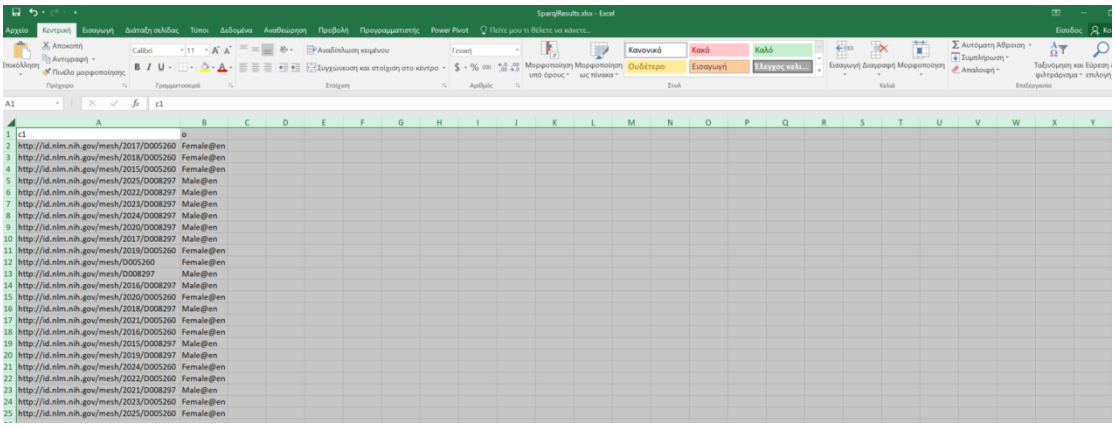

Εικόνα 18. Φόρμα "Results"

Στο δεύτερο tab, μπορούμε να δούμε τα αποτελέσματα σε μορφή data grid, που διευκολύνει την οπτική παρουσίαση και επεξεργασία των δεδομένων.




Εικόνα 19. Data grid

Πατώντας το πλήκτρο "Excel", μπορούμε να εξάγουμε τα αποτελέσματα σε αρχείο Excel, διευκολύνοντας έτσι την περαιτέρω ανάλυση ή αποθήκευση των δεδομένων σε έναν εύχρηστο μορφότυπο.

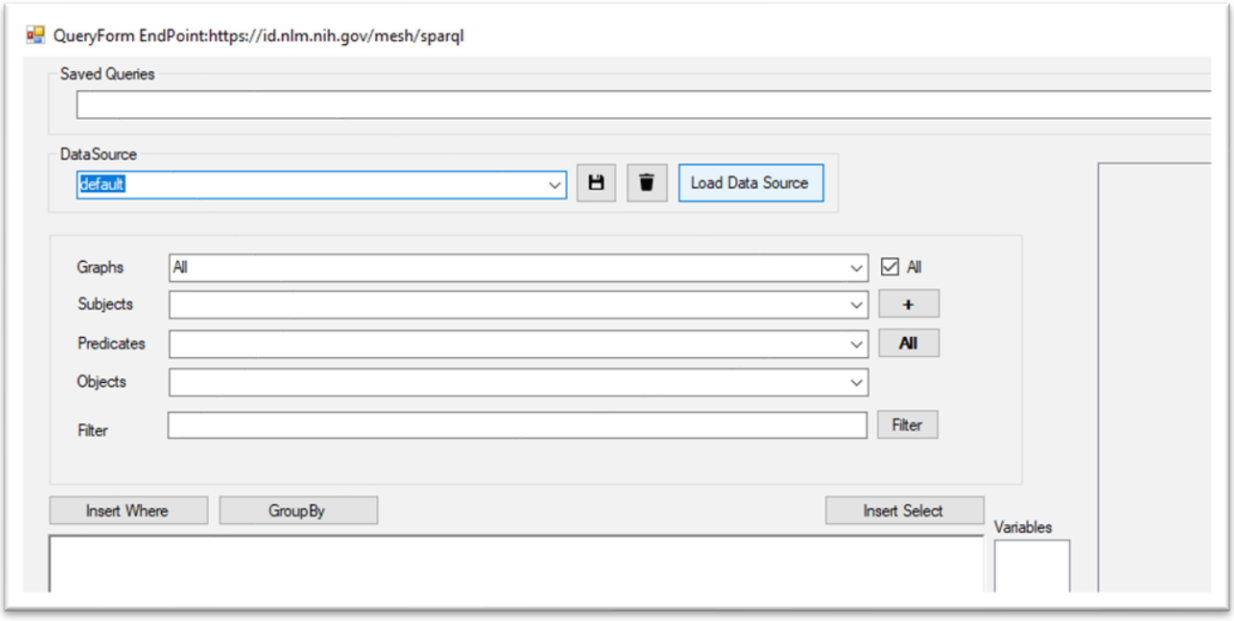


Εικόνα 20. Excel

# Κεφάλαιο 5: Δημιουργία Ερωτημάτων SPARQL

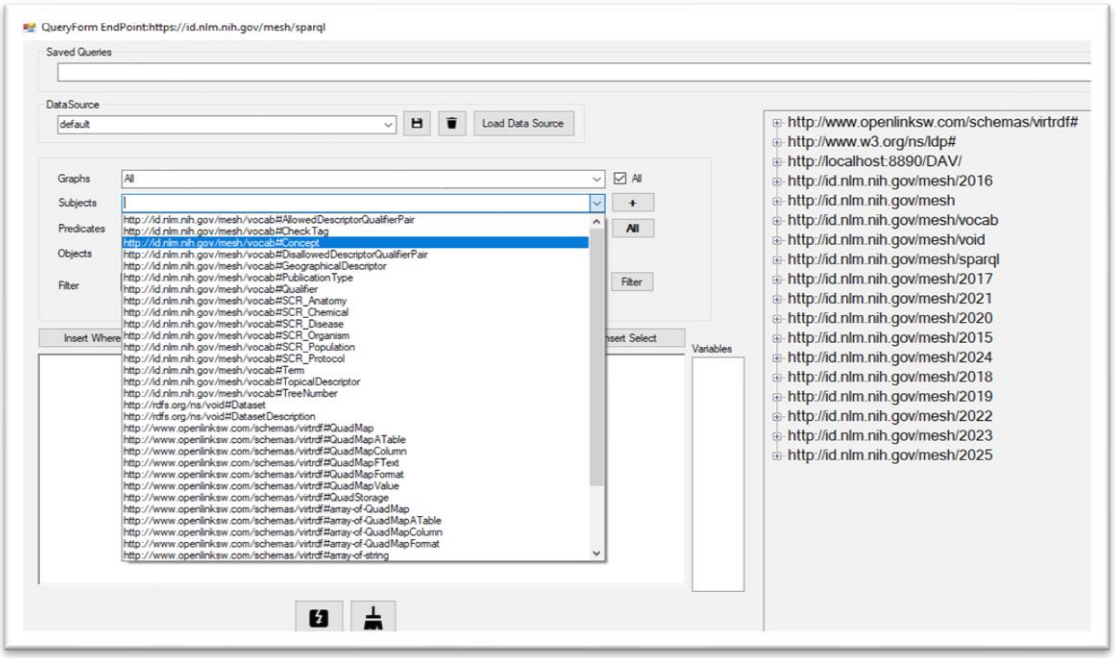
**Αναζήτηση για labels που περιέχουν τη λέξη «chloride»**

1. **Επιλέγουμε το End Point**: Επιλέξτε το end point "mesh".
2. **Επιλέγουμε το Default DataSet**: Πατήστε για να κάνετε Load το default DataSet.



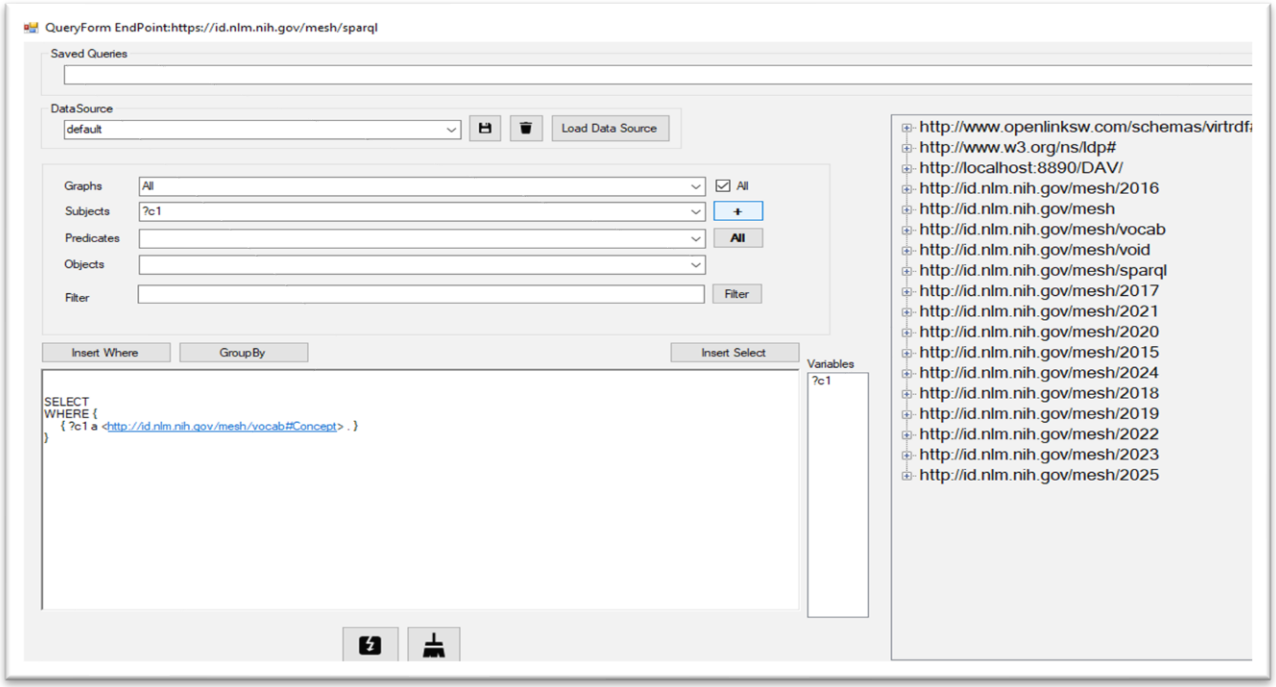

Εικόνα 21. Επιλογή Data Source

3. **Επιλέγουμε Subject**: Από τη λίστα των Subjects, επιλέξτε το Concept που σας ενδιαφέρει.




Εικόνα 22. Επιλογή Subject

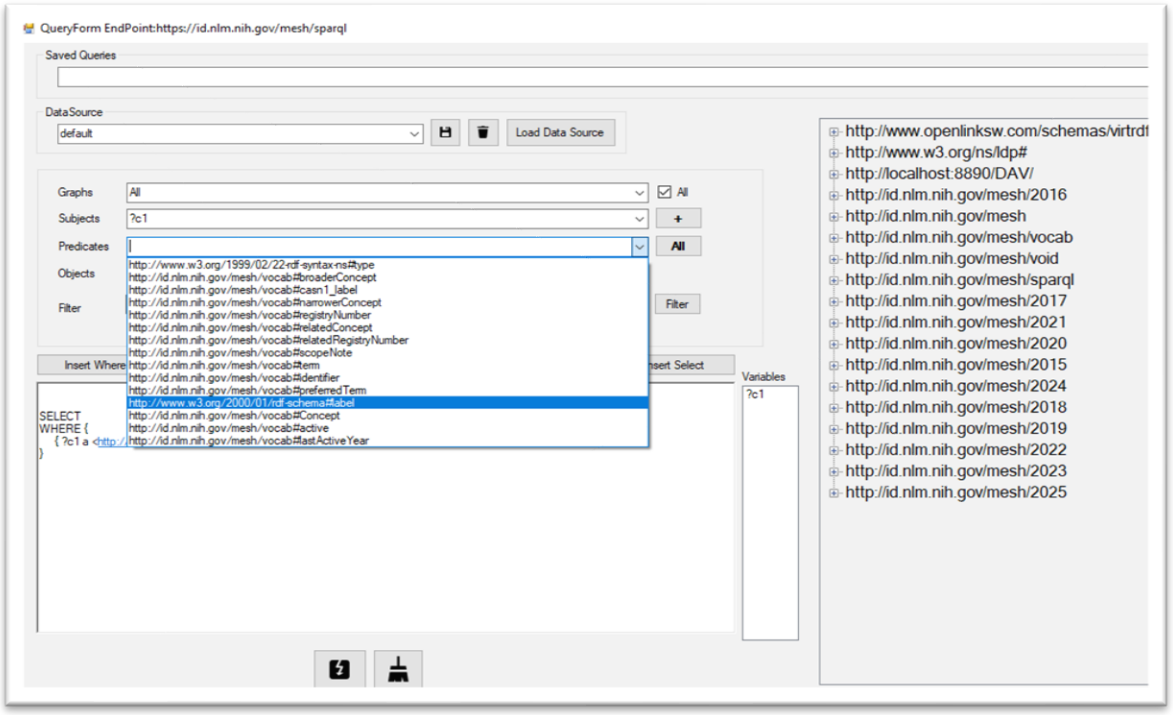
4. **Προσθήκη Statement**: Πατήστε το κουμπί "+" για να προσθέσετε το πρώτο statement.




Εικόνα 23. Προσθήκη Statement

5. **Επιλέγουμε Predicate**:

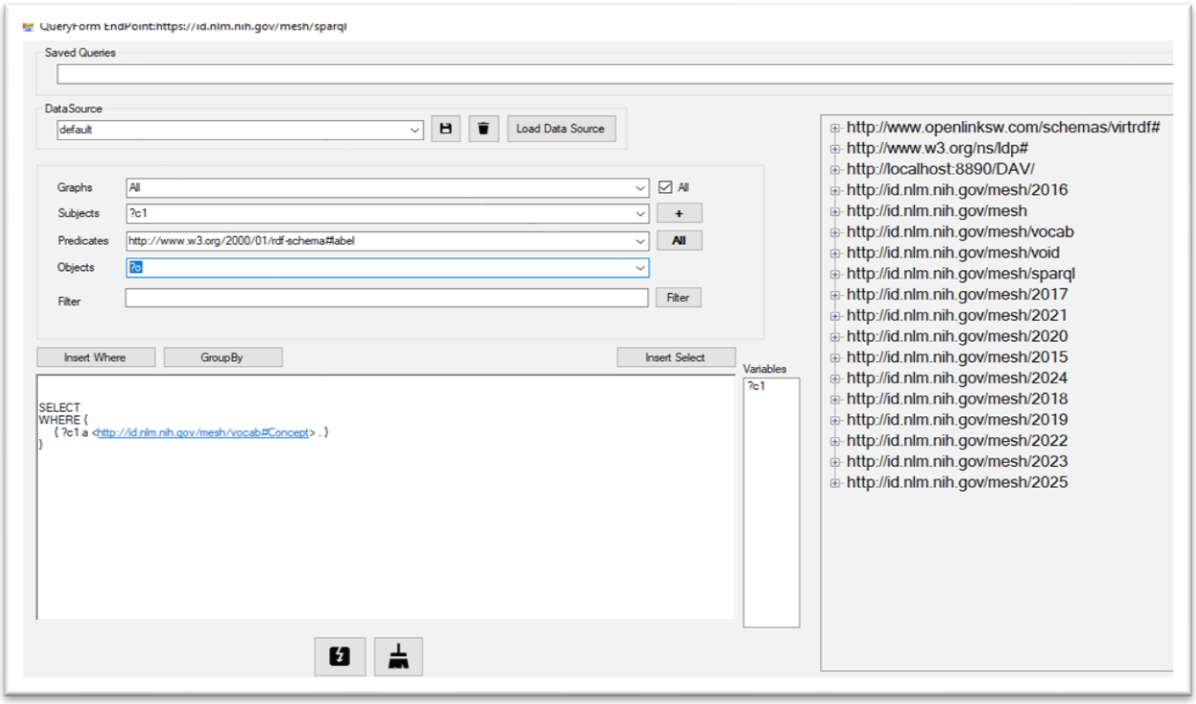
- Ανοίξτε τη λίστα με τα predicates που έχουν φορτωθεί δυναμικά βάσει της επιλογής στη λίστα Subject.
- Επιλέξτε το label.




Εικόνα 24. Επιλογή Predicate

6. **Object**:

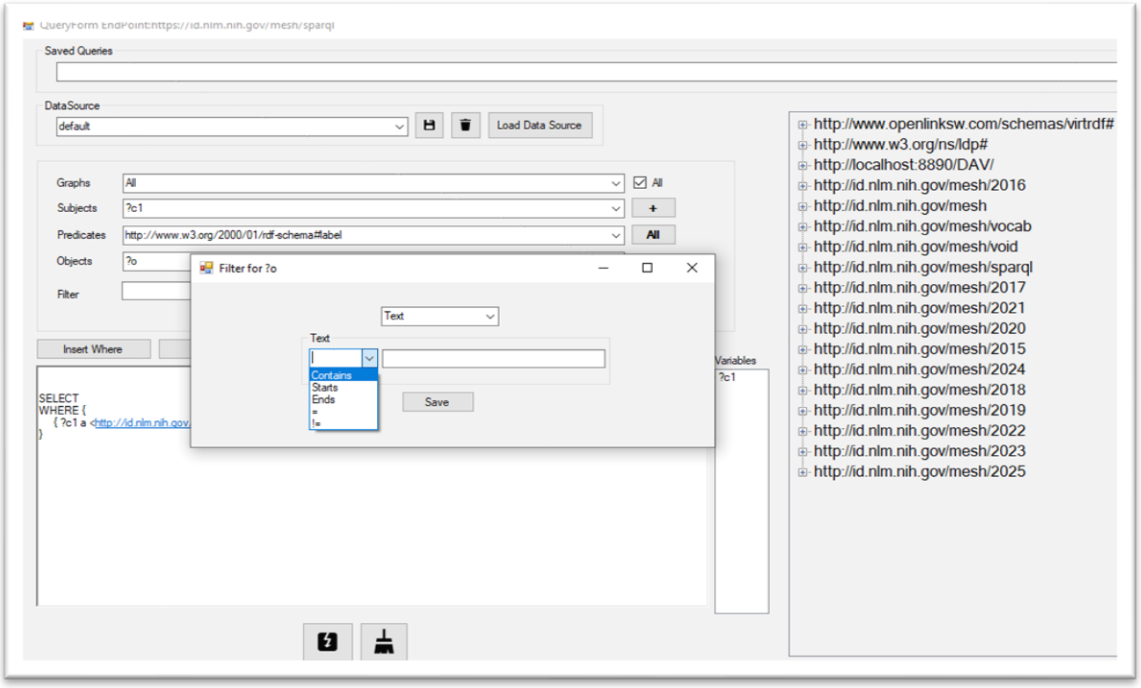
- Βάζουμε μια μεταβλητή (**?o**) αν δεν αναζητούμε κάποιο συγκεκριμένο αντικείμενο. Αυτό επιτρέπει ευέλικτη σύνθεση ερωτημάτων για γενική ανάλυση δεδομένων.



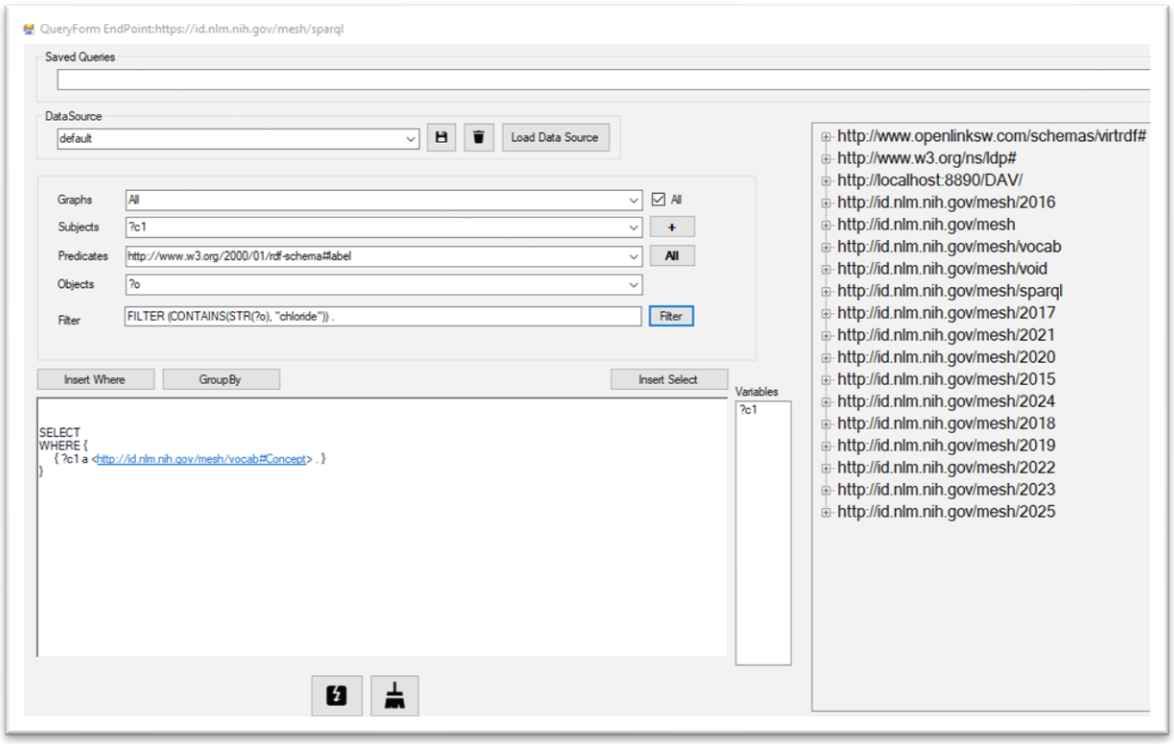

Εικόνα 25. Ορισμός μεταβλητής

7. **Φιλτράρισμα**:

- Πατήστε το κουμπί **Filter** και επιλέξτε φιλτράρισμα με κείμενο.
- Επιλέξτε το **Contains** ως συνθήκη και εισαγάγετε τη λέξη "chloride".

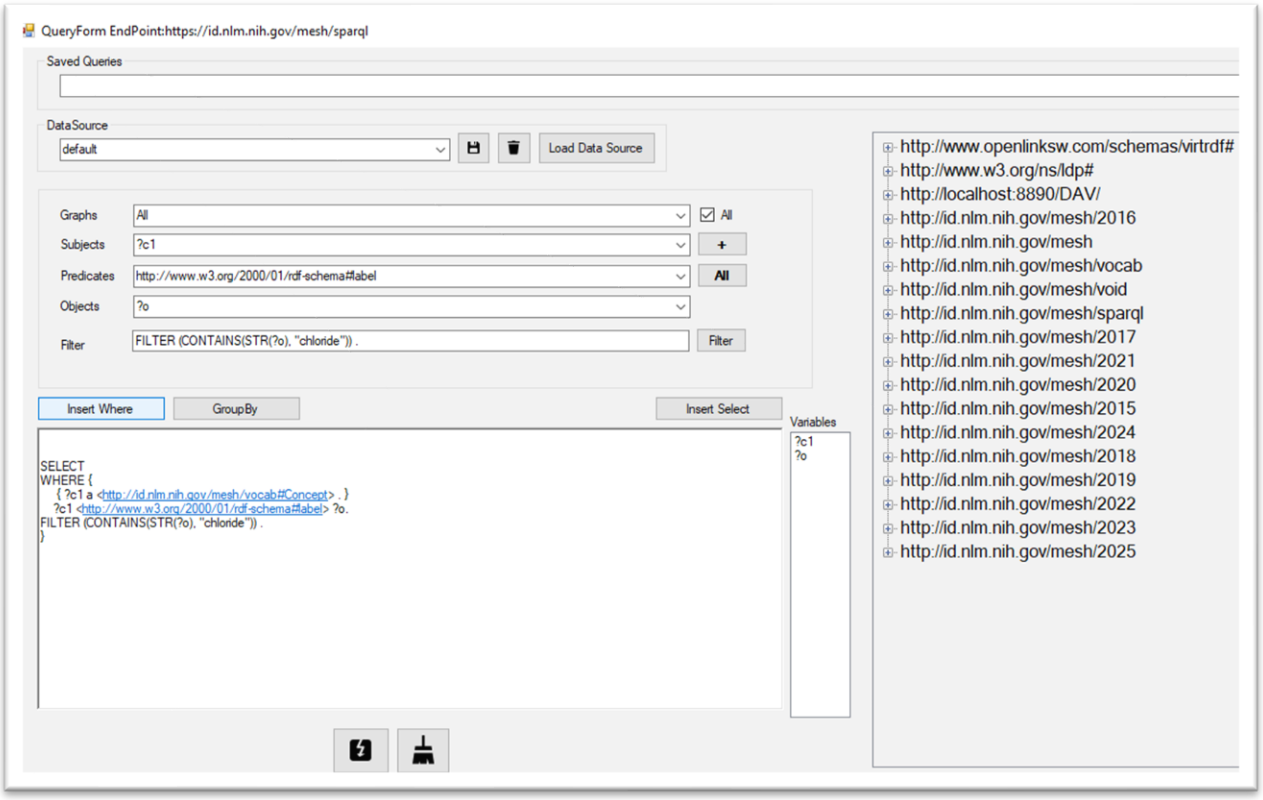



Εικόνα 26. Φόρμα Φιλτραρίσματος



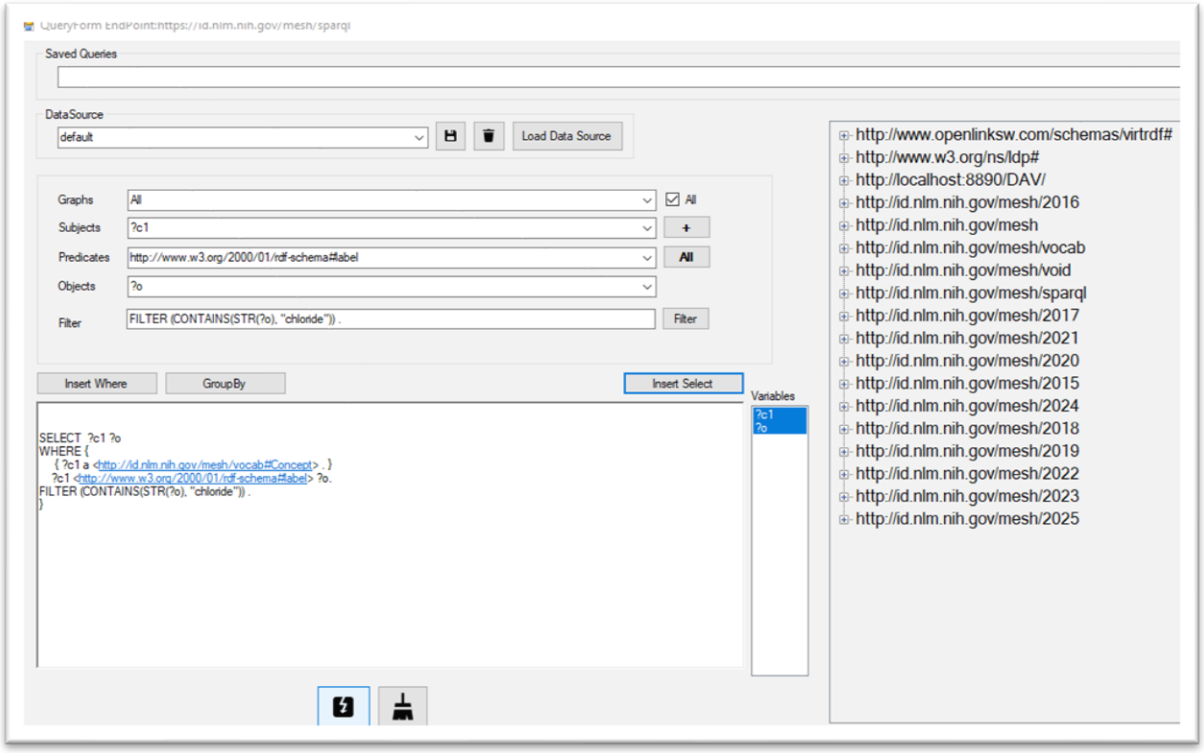

Εικόνα 27. Κώδικας φιλτραρίσματος

8. **Αποθήκευση Statement**: Πατήστε **Save** και **Insert Where** για να προσθέσετε το statement στο ερώτημά σας.




Εικόνα 28. Εισαγωγή κώδικα φιλτραρίσματος στο Ερώτημα

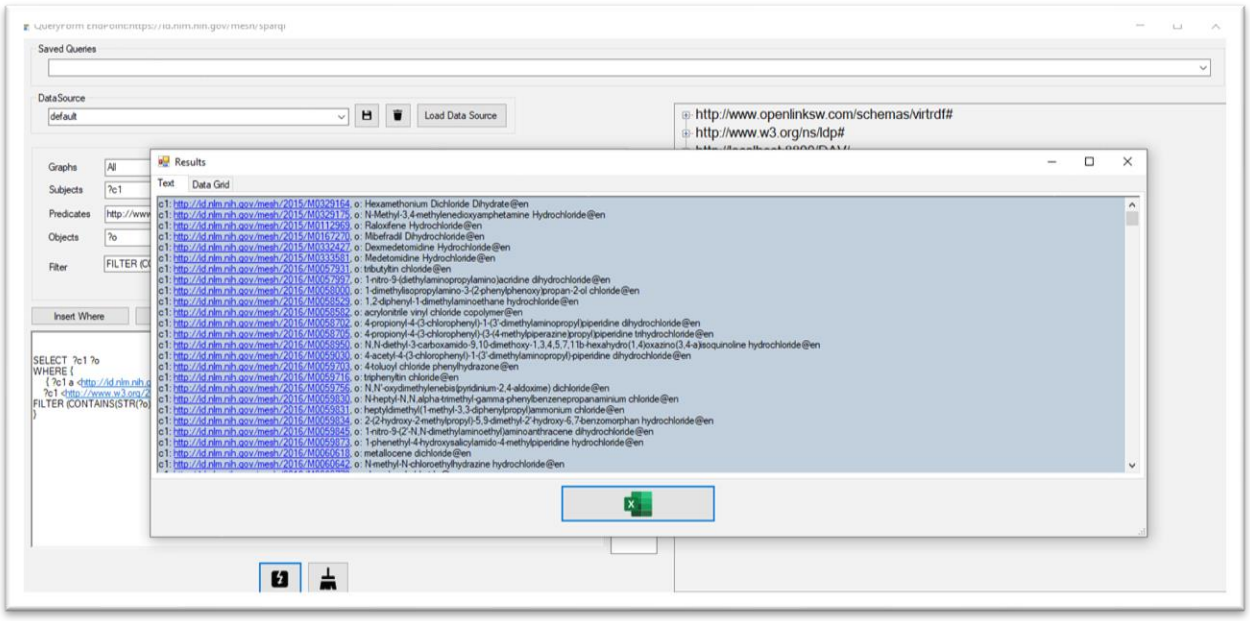
9. **Επιλογή Μεταβλητών**: Επιλέξτε ποιες μεταβλητές θέλετε να εμφανιστούν στο **SELECT**.




Εικόνα 29. Επιλογή μεταβλητών

10. **Εκτέλεση Ερωτήματος**: Πατήστε **Execute** για να εκτελέσετε το ερώτημα.

11. **Εμφάνιση Αποτελεσμάτων**: Τα αποτελέσματα εμφανίζονται στη φόρμα **Results**.

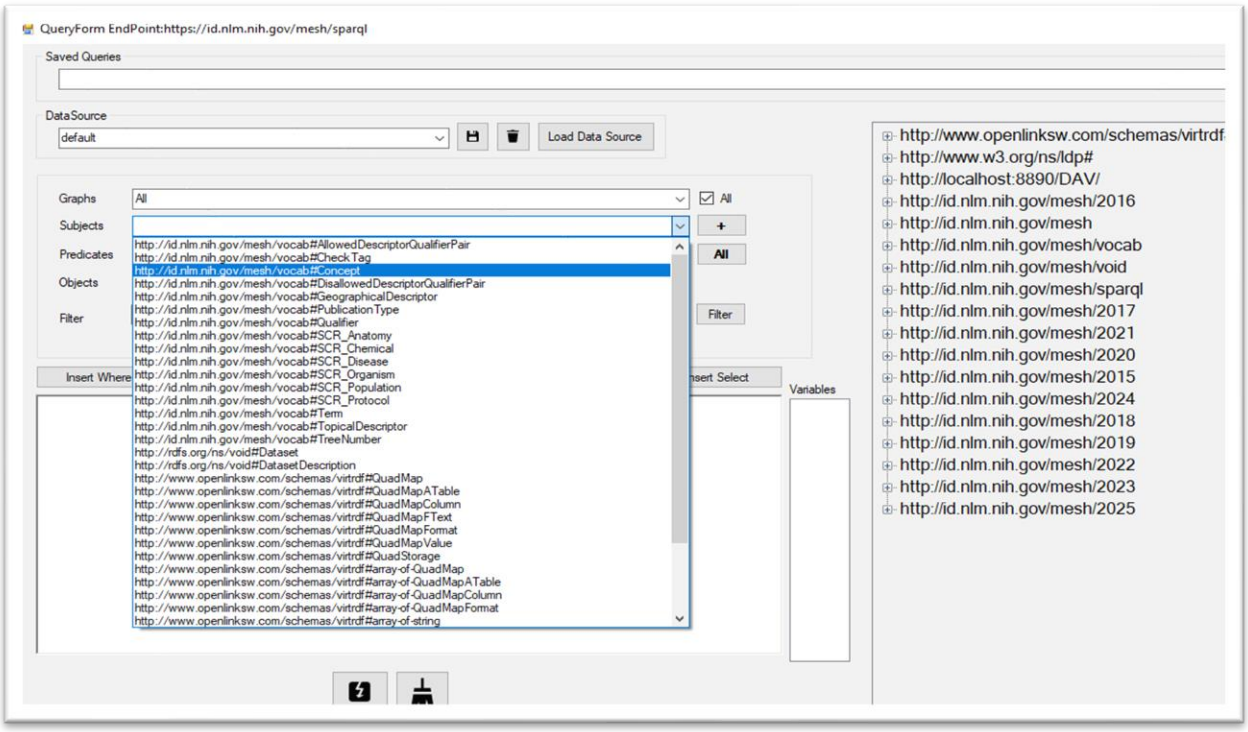



Εικόνα 30. Αποτελέσματα

## Χρήση Group By

Για να βρούμε συσχετίσεις ανάμεσα σε Concepts:

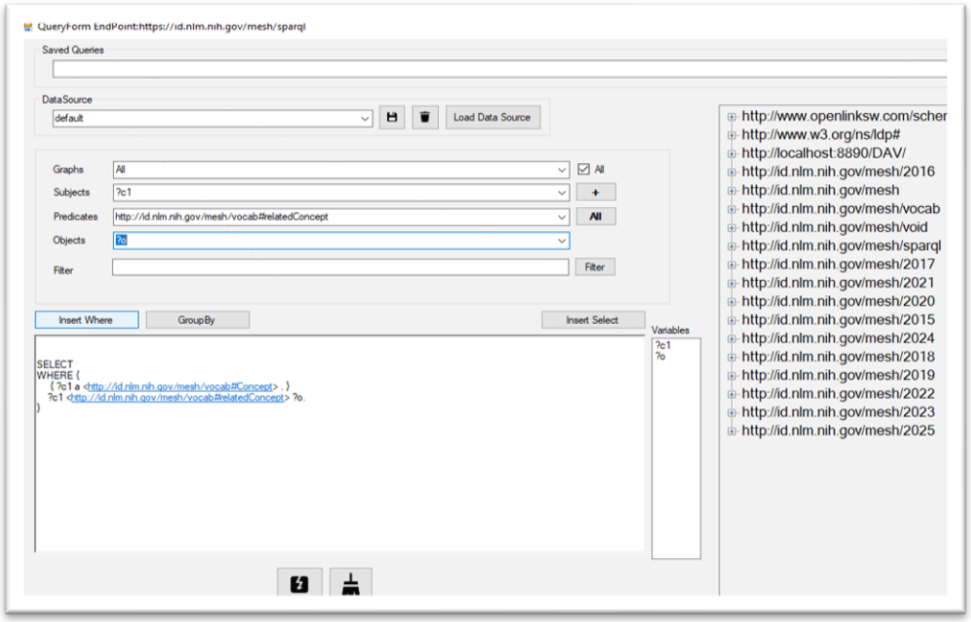
**Επιλογή Subject**: Από τη λίστα των Subjects, επιλέξτε το **Concept** που σας ενδιαφέρει.




Εικόνα 31. Επιλογή Subject

**Προσθήκη Statement**: Πατήστε το κουμπί "+" και εισαγάγετε τα ακόλουθα:

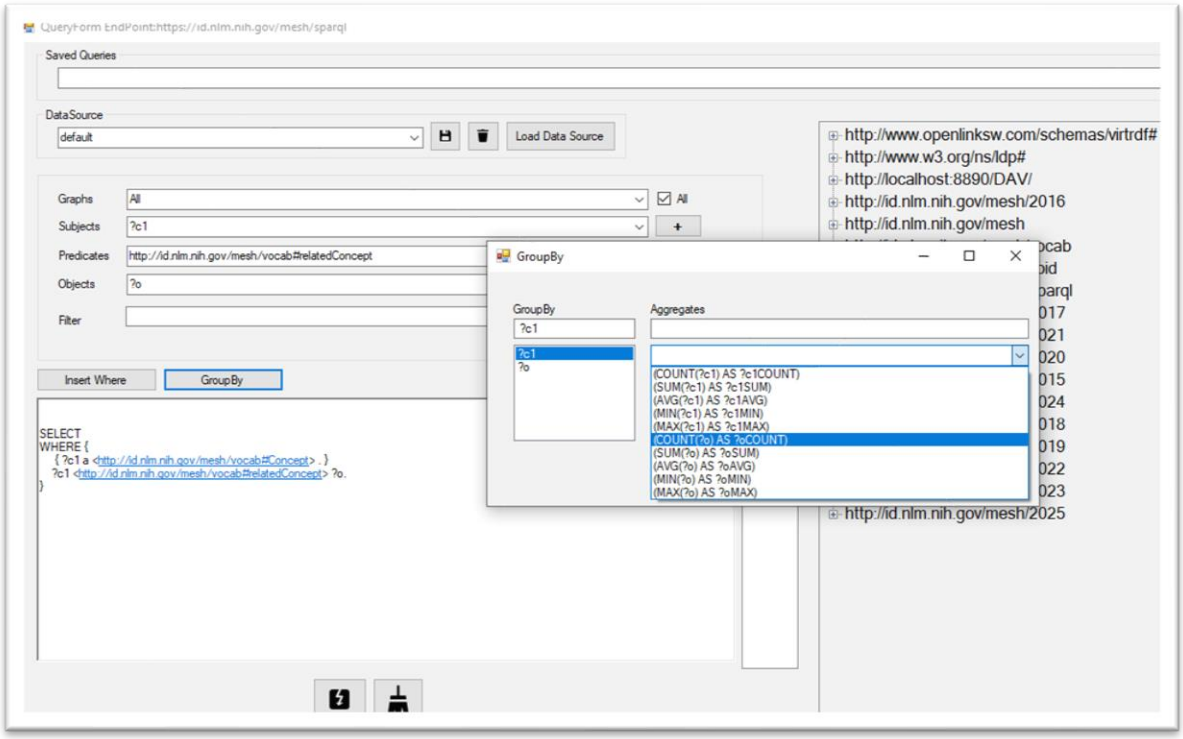
- **Subject**: Το Concept που επιλέξατε.
- **Predicate**: Επιλέξτε τη συσχέτιση που θέλετε (π.χ., " relatedConcept ").
- **Object**: Τοποθετήστε μία μεταβλητή (π.χ., `?o`).




Εικόνα 32. Προσθήκη Statement

**Προσθήκη Group By**:

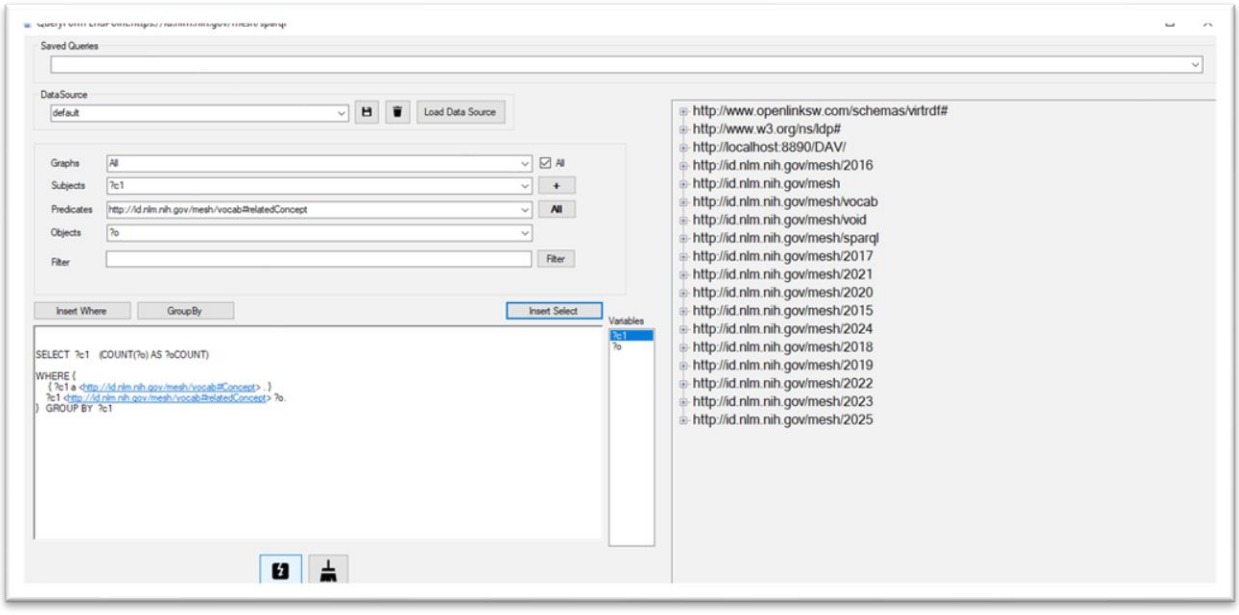
- Πατήστε **Group By** και επιλέξτε τη μεταβλητή `?c1`.




Εικόνα 33.Φόρμα Group By

**Επιλογή Aggregate Function**:

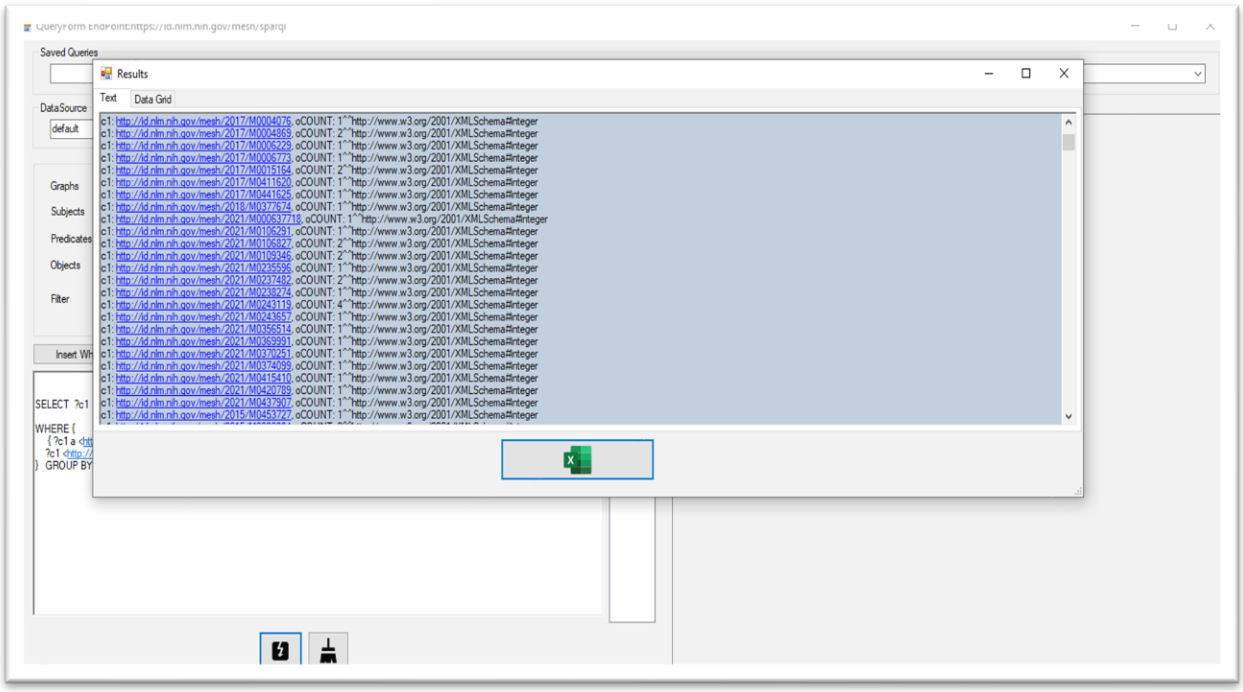
- Εάν χρειάζεται, εφαρμόστε μία συναρτησιακή λειτουργία (π.χ., COUNT) στη μεταβλητή `?c1`.




Εικόνα 34. Τελικός κώδικας ερωτήματος

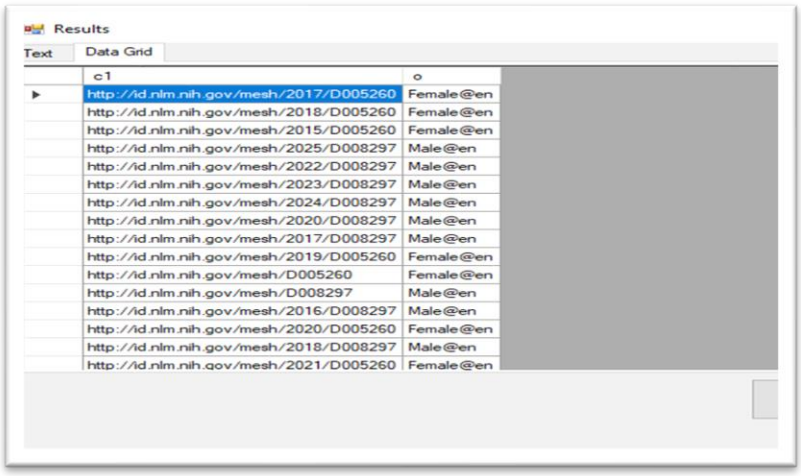
**Εκτέλεση Ερωτήματος**: Πατήστε **Execute**.

**Ανάλυση Αποτελεσμάτων**: Τα αποτελέσματα εμφανίζονται ομαδοποιημένα βάσει της μεταβλητής που επιλέξατε.



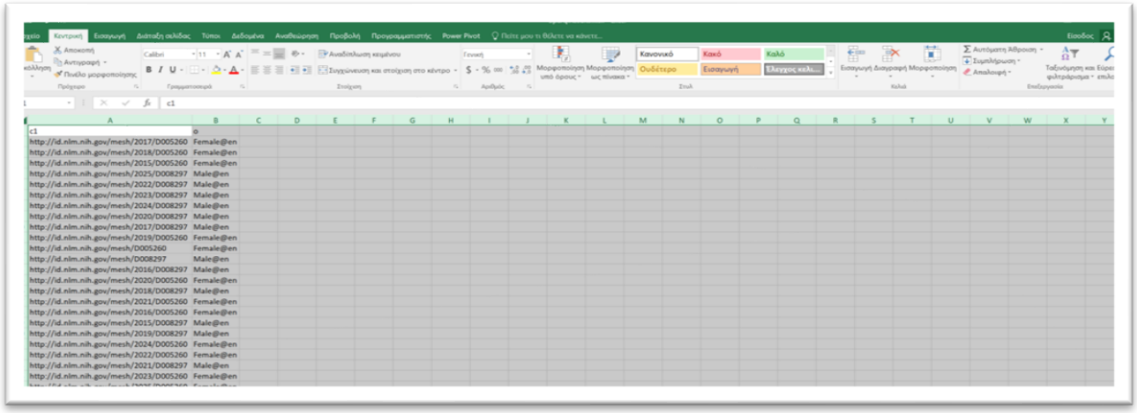

Εικόνα 35. Αποτελέσματα

Στο δεύτερο tab μπορούμε να τα δούμε σε μορφή data grid.




Εικόνα 36. Αποτελέσματα σε Data Grid

Πατώντας το πλήκτρο excel μπορούμε να τα εξάγουμε σε αρχείο excel.

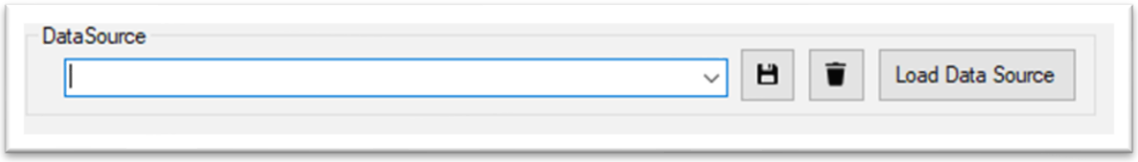


Εικόνα 37. Αποτελέσματα σε Excel

# Κεφάλαιο 6: Τεκμηρίωση Κώδικα

Η εφαρμογή έχει δημιουργηθεί σε γλώσσα προγραμματισμού **VB** με χρήση του **.Net Framework**. Επιτρέπει την εύκολη δημιουργία και εκτέλεση ερωτημάτων **SPARQL**, αξιοποιώντας ένα φιλικό προς το χρήστη γραφικό περιβάλλον (GUI). Οι χρήστες μπορούν να επιλέξουν δεδομένα από πηγές, να ορίσουν **graphs**, **subjects**, **predicates** και **objects**, και να συνθέσουν ένα πλήρες ερώτημα **SPARQL** για εκτέλεση. Το σύστημα βασίζεται στη δυναμική φόρτωση δεδομένων και την ευέλικτη διαχείριση πηγών και γράφων, παρέχοντας τη δυνατότητα διαμόρφωσης πολύπλοκων ερωτημάτων.

## 6.1 Φόρτωση Πηγών Δεδομένων




Εικόνα 38. Λίστα Data Source

Κατά τη φόρτωση της φόρμας, το σύστημα ανακτά και εμφανίζει τις διαθέσιμες πηγές δεδομένων στο πεδίο επιλογής (ComboBox) `cmbDataSources`. Οι πηγές αυτές ανακτώνται από αρχείο **JSON** μέσω της μεθόδου `JsonSettingsManager.LoadComboBoxFromList`. Αυτό επιτρέπει τη δυναμική φόρτωση των διαθέσιμων πηγών.

```
Private Sub QueryForm_Load(sender As Object, e As EventArgs) Handles Me.Load

    Me.Text = Me.Text + " EndPoint:" + endPointUrl

    JsonSettingsManager.LoadComboBoxFromList(cmbDataSources, "DataSources")

End Sub
```

Η μέθοδος φορτώνει τις διαθέσιμες πηγές από το αρχείο ρυθμίσεων **JSON** και ενημερώνει τον τίτλο της φόρμας με το URL του **SPARQL End point** που έχουμε επιλέξει αρχικά.

### 6.1.1 Ανάκτηση Γράφων

Εικόνα 39. Λίστα Γράφων

Η μέθοδος `btnLoadDataSource_Click` εκτελεί ένα ερώτημα **SPARQL** για την ανάκτηση των διαθέσιμων γράφων από την επιλεγμένη πηγή δεδομένων. Αν η πηγή είναι διαφορετική από την 'default', χρησιμοποιείται η εντολή `SERVICE` ώστε να υποστηρίζονται **federated queries**.

```
Dim query As String

If cmbDataSources.Text <> "default" Then
```

```
    query = "SELECT DISTINCT ?graph
            WHERE SERVICE <" + cmbDataSources.Text + "> { {
                GRAPH ?graph { ?s ?p ?o }
            }}"
Else
    query = "SELECT DISTINCT ?graph
            WHERE {
                GRAPH ?graph { ?s ?p ?o }
            }"
End If
```

Η μέθοδος εκτελεί το παραπάνω ερώτημα και αποθηκεύει τα αποτελέσματα στη λίστα `cmbGraphs`, επιτρέποντας στον χρήστη να επιλέξει έναν γράφο.

### 6.1.2 Ανάκτηση Subject

Εικόνα 40. Λίστα Subject

Μετά την επιλογή γραφήματος, το σύστημα ανακτά τα διαθέσιμα **subjects** που περιέχονται στο συγκεκριμένο γράφο. Αυτά τα subjects αντιστοιχούν σε δηλώσεις `rdf:type`.

```
Dim query As String = "SELECT DISTINCT ?type
                      WHERE {
                          GRAPH <" + selectedGraph + "> {
                              ?subject a ?type .
                          }
                      }
                      ORDER BY ?type"
```

Η μέθοδος δημιουργεί και εκτελεί το παραπάνω ερώτημα, προσθέτοντας τα μοναδικά subjects στο `cmbSubjects`.

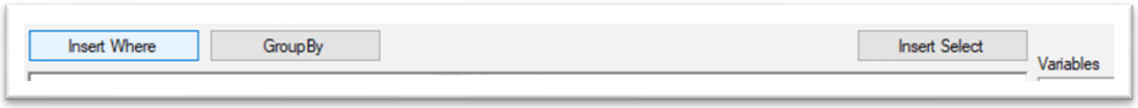
### 6.1.3 Ανάκτηση Ιδιοτήτων (Predicates)

Εικόνα 41. Λίστα Predicates

Η μέθοδος `cmbSubjects_SelectedIndexChanged` ανακτά τις διαθέσιμες ιδιότητες (**predicates**) για το επιλεγμένο subject. Αυτές οι ιδιότητες αποτελούν τα αντικείμενα των σχέσεων που σχετίζονται με το subject.

```
Dim query As String = "SELECT DISTINCT ?predicate

                      WHERE {

                          GRAPH <" + cmbGraphs.Text + "> {

                              ?s a <" + selectedSubject + "> .

                              ?s ?predicate ?o .

                          }

                      }

                      LIMIT 100"
```

## 6.2 Δημιουργία Ερωτημάτων SPARQL




Εικόνα 42. Πλήκτρο Insert Where

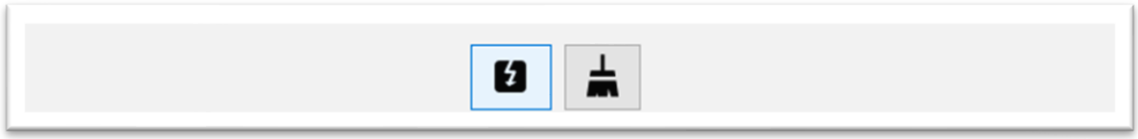
Η μέθοδος `btnInsertWhere_Click` επιτρέπει στους χρήστες να δημιουργήσουν προτάσεις **SPARQL** χρησιμοποιώντας τις επιλογές τους για subject, predicate και object. Αυτές οι προτάσεις συνδυάζονται στο τελικό ερώτημα.

```
Dim statement As String

statement = formatString(cmbSubjects.Text) + " " + formatString(cmbPredicates.Text) + " " + formatString(cmbObjects.Text) + "."

statements.Add(statement)



statementsString = ""

For Each s In statements
```

```
    statementsString = statementsString + s + vbCrLf

Next


rtxtQuery.Text = prefixSelect + aggregateSelect + fromWhere + statementsString
+ closeQuery + groupByString
```

Το σύστημα συνδυάζει τα μέρη του ερωτήματος (**SELECT**, **WHERE**) με τις προτάσεις που δημιουργούνται από τον χρήστη.

### 6.2.1 Εκτέλεση Ερωτήματος



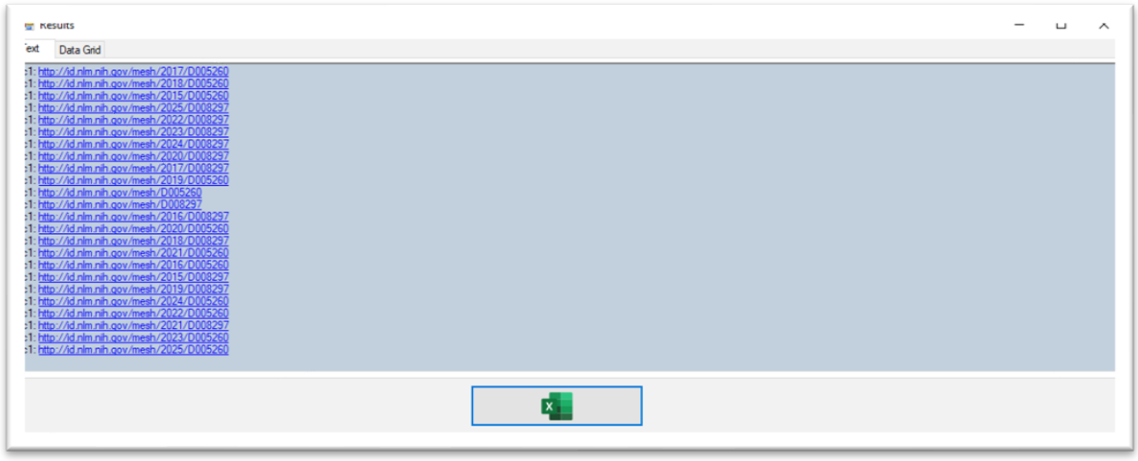
Εικόνα 43. Πλήκτρο εκτέλεσης ερωτήματος

Η μέθοδος `btnTestQuery_Click` εκτελεί το ερώτημα **SPARQL** που έχει δημιουργήσει ο χρήστης και εμφανίζει τα αποτελέσματα σε νέο παράθυρο διαλόγου.

```
Dim f As New Results

f.resultList = RunSparqlQuery(rtxtQuery.Text)

If f.resultList.Count = 0 Then MsgBox("No results found")

f.ShowDialog()
```

Αν δεν υπάρχουν αποτελέσματα, εμφανίζεται μήνυμα προς τον χρήστη. Αν υπάρχουν, τα αποτελέσματα εμφανίζονται σε νέο παράθυρο.

### 6.2.2 Φόρμα Results




Εικόνα 44. Φόρμα Αποτελεσμάτων

Η φόρμα **Results** έχει σχεδιαστεί για να εμφανίζει τα αποτελέσματα ερωτημάτων **SPARQL** με φιλικό προς τον χρήστη τρόπο και να παρέχει στον χρήστη τρεις διαφορετικές επιλογές για να δει και να διαχειριστεί αυτά τα αποτελέσματα:

1. Προβολή των αποτελεσμάτων ως κείμενο σε **RichTextBox**.
2. Προβολή των αποτελεσμάτων σε μορφή πίνακα μέσω **DataGridView**.
3. Εξαγωγή των αποτελεσμάτων σε αρχείο **Excel** και άνοιγμα του αρχείου για περαιτέρω ανάλυση.

**Δομή της Φόρμας**

Η φόρμα περιλαμβάνει τα εξής βασικά στοιχεία:

- **RichTextBox (rtxtQueryResults)**: Προβάλλει τα αποτελέσματα του ερωτήματος ως απλό κείμενο, ιδανικό για χρήστες που προτιμούν την προβολή δεδομένων σε ακατέργαστη μορφή.
- **DataGridView (DataGridView1)**: Εμφανίζει τα αποτελέσματα σε μορφή πίνακα, ιδανικό για δεδομένα με πολλές στήλες και γραμμές.
- **Κουμπί εξαγωγής Excel (Button1)**: Επιτρέπει στον χρήστη να εξάγει τα αποτελέσματα σε αρχείο Excel και να το ανοίξει.

**Ανάλυση Κώδικα**

*1. Αρχικοποίηση της Φόρμας (Results_Load)*

Η μέθοδος `Results_Load` καλείται κατά τη φόρτωση της φόρμας και εκτελεί τα εξής βήματα:

1. Μετατρέπει τη λίστα αποτελεσμάτων (`resultList`) σε μορφή κειμένου και την εμφανίζει στο **RichTextBox** (`rtxtQueryResults`).
2. Μετατρέπει τη λίστα αποτελεσμάτων σε **DataTable** μέσω της μεθόδου `ConvertToDataTable` και τη συνδέει με το **DataGridView** (`DataGridView1`).

3. Ρυθμίζει το **DataGridView** ώστε οι στήλες και οι γραμμές να προσαρμόζονται αυτόματα στο περιεχόμενο.
4. Δημιουργεί ένα αρχείο **Excel** (`SparqlResults.xlsx`) με τα αποτελέσματα, καλώντας τη μέθοδο `CreateExcelFromSparqlResults`.

```
Private Sub Results_Load(sender As Object, e As EventArgs) Handles Me.Load

    rtxtQueryResults.Text = ConvertListToString(resultList)

    DataGridView1.DataSource = ConvertToDataTable(resultList)

    With DataGridView1

        .AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells

        .AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells

        .ColumnHeadersHeightSizeMode                                    =
DataGridViewColumnHeadersHeightSizeMode.AutoSize

    End With

    filePath = "SparqlResults.xlsx"

    CreateExcelFromSparqlResults(resultList, filePath)

End Sub
```

### Μετατροπή Λίστας σε DataTable

Η μέθοδος **`ConvertToDataTable`** μετατρέπει τη λίστα αποτελεσμάτων (`resultList`) σε **DataTable**, επιτρέποντας την εμφάνιση των δεδομένων στο **DataGridView**.

### Βήματα:

1. Δημιουργεί ένα νέο αντικείμενο **DataTable.**
2. Προσθέτει τις στήλες στο **DataTable** βάσει των κλειδιών του πρώτου **dictionary** στη λίστα.
3. Για κάθε **dictionary** στη λίστα, δημιουργεί μια νέα γραμμή (**DataRow**) και προσθέτει τα δεδομένα στο **DataTable**.

```
Public Function ConvertToDataTable(resultsList As List(Of Dictionary(Of String,
String))) As DataTable

    Dim dataTable As New DataTable()

    If resultsList.Count = 0 Then

        Return dataTable

    End If

    For Each column As String In resultsList(0).Keys
```

```
            dataTable.Columns.Add(column)
        Next
        For Each row As Dictionary(Of String, String) In resultsList
            Dim dataRow As DataRow = dataTable.NewRow()
            For Each column As String In row.Keys
                dataRow(column) = row(column)
            Next
            dataTable.Rows.Add(dataRow)
        Next
        Return dataTable
End Function
```

Αυτή η μέθοδος επιτρέπει τη δυναμική διαχείριση των δεδομένων, προσαρμόζοντας τις στήλες και τις γραμμές του πίνακα στις ανάγκες του χρήστη.

### 6.2.3 Άνοιγμα Αρχείου Excel

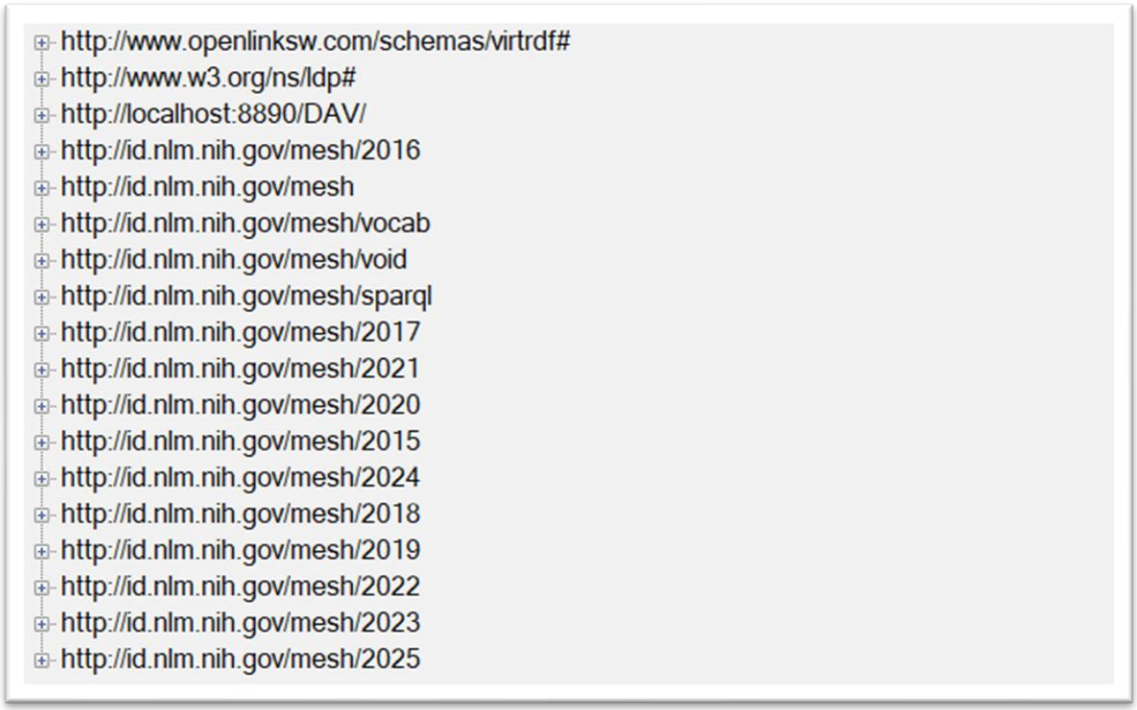
Εικόνα 45. Εξαγωγή σε Excel

Η μέθοδος **`Button1_Click`** καλείται όταν ο χρήστης πατήσει το κουμπί εξαγωγής Excel. Χρησιμοποιεί τη μέθοδο **`OpenExcelFile`** για να ανοίξει το αρχείο Excel που δημιουργήθηκε.

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
    OpenExcelFile(filePath)
End Sub
```

Αυτή η μέθοδος επιτρέπει στον χρήστη να ανοίξει και να επεξεργαστεί τα αποτελέσματα στο Excel με ένα μόνο κλικ.

## 6.3 Δυναμικό TreeView




Εικόνα 46. Tree View

Η παρούσα υλοποίηση του **TreeView** έχει σχεδιαστεί για την προβολή δεδομένων **RDF** από ένα **SPARQL end point**, χρησιμοποιώντας μια ιεραρχική δομή. Το **TreeView** αναπτύσσεται δυναμικά ανάλογα με το επίπεδο δεδομένων που επεκτείνεται κάθε φορά. Στόχος είναι να παρουσιάζεται σταδιακά η πληροφορία σε τέσσερα επίπεδα: **Graphs**, **Types**, **Predicates** και **Objects**.

Η λειτουργία του **TreeView** ακολουθεί μια προσέγγιση δυναμικής φόρτωσης, ώστε να διατηρείται η αποδοτικότητα και η ευελιξία. Συγκεκριμένα:

1. **Αρχικοποίηση του TreeView:**
   - Κατά την εκκίνηση, το **TreeView** γεμίζει με τα διαθέσιμα **Graphs** από το **SPARQL end point**.
   - Κάθε κόμβος **Graph** περιέχει έναν placeholder κόμβο (dummy), ο οποίος υποστηρίζει τη δυναμική φόρτωση δεδομένων όταν επεκτείνεται.
2. **Διαχείριση Επέκτασης Κόμβων:**
   - Όταν ο χρήστης πατάει το σύμβολο + για να επεκτείνει έναν κόμβο, ελέγχεται το επίπεδό του μέσω της ιδιότητας Tag.
   - Ανάλογα με το επίπεδο, εκτελείται το αντίστοιχο **SPARQL query** για να φορτωθούν τα δεδομένα και να προστεθούν νέοι κόμβοι στο **TreeView**.
3. **Ιεραρχική Δομή:** Το **TreeView** οργανώνεται σε τέσσερα επίπεδα:
   - **Graph:** Το βασικό επίπεδο που αντιστοιχεί σε διαφορετικά γραφήματα.
   - **Type:** Οι τύποι των subjects που ανήκουν σε κάθε γράφημα.
   - **Predicate:** Οι ιδιότητες που συνδέουν τα subjects με τα objects.
   - **Object:** Τα αντικείμενα που συνδέονται με ένα συγκεκριμένο predicate.
4. **Αποφυγή Επαναφόρτωσης:**
   - Αν ένας κόμβος έχει ήδη φορτώσει τους υποκόμβους του, δεν γίνεται νέα κλήση προς το **SPARQL end point** για αποφυγή επαναφόρτωσης δεδομένων.

Η υλοποίηση χωρίζεται σε δύο κύρια μέρη:

### *1. Αρχικοποίηση του TreeView*

Κατά την αρχικοποίηση, εκτελείται ένα **SPARQL query** για την ανάκτηση των διαθέσιμων **graphs** από το **SPARQL end point**. Κάθε **Graph** που επιστρέφεται δημιουργεί έναν κόμβο στο **TreeView** με την ιδιότητα `Tag` ορισμένη ως "Graph". Επίσης, προστίθεται ένας **dummy** υποκόμβος για να υποστηρίξει τη δυναμική φόρτωση δεδομένων.

```
Public Sub FillTreeViewFromGraphs(treeView As TreeView, cmbGraphs As ComboBox)

    ' Καθαρισμός του TreeView πριν την εισαγωγή νέων κόμβων

    treeView.Nodes.Clear()



    ' Προσθήκη ριζικών κόμβων από το cmbGraphs

    For Each item As String In cmbGraphs.Items

        Dim rootNode As TreeNode = treeView.Nodes.Add(item)



        ' Ορισμός του Tag ως "Graph" για τον κόμβο επιπέδου Graph

        rootNode.Tag = "Graph"



        ' Προσθήκη dummy υποκόμβου για να εμφανιστεί το "+"

        rootNode.Nodes.Add("dummy")

    Next

End Sub
```

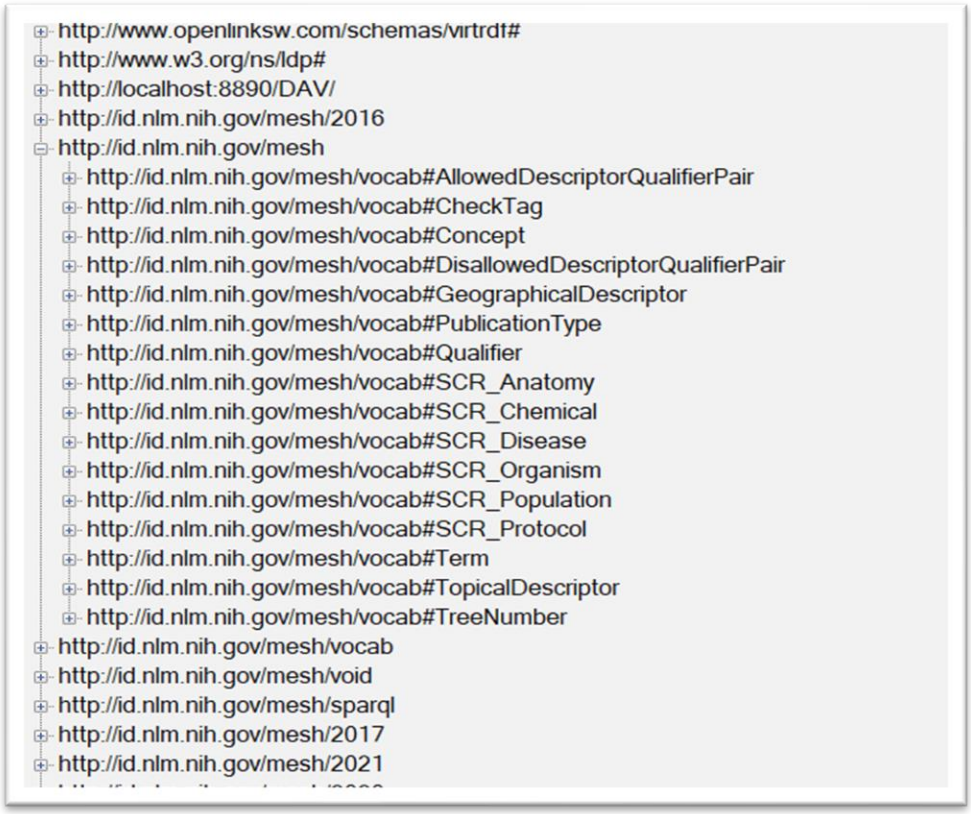
### *2. Διαχείριση του `BeforeExpand`*

Το event **`BeforeExpand`** ενεργοποιείται όταν ο χρήστης επεκτείνει έναν κόμβο. Η μέθοδος που καλείται ελέγχει το επίπεδο του κόμβου μέσω του `Tag` και εκτελεί το κατάλληλο **SPARQL query** για να φορτώσει τους υποκόμβους. Ο **dummy** κόμβος διαγράφεται και αντικαθίσταται από τους πραγματικούς κόμβους.

```
Private Sub treeView_BeforeExpand(sender As Object, e As
TreeViewCancelEventArgs) Handles treeView.BeforeExpand

    Dim selectedNode As TreeNode = e.Node



    ' Κλήση της μεθόδου για φόρτωση υποκόμβων από το module
```

```
    Tree.LoadChildrenForNode(selectedNode)

End Sub
```

Αυτή η υλοποίηση διασφαλίζει τη δυναμική φόρτωση των δεδομένων στο **TreeView**, επιτρέποντας την αποτελεσματική παρουσίαση των **RDF δεδομένων** με ιεραρχική δομή και αποφεύγοντας την υπερφόρτωση των δεδομένων.




Εικόνα 47. Επέκταση Γράφου

Όταν επεκτείνεται ένας κόμβος **Graph**, εκτελείται ένα **SPARQL query** για να ανακτηθούν οι διαθέσιμοι **τύποι (types)** για τα subjects του συγκεκριμένου γραφήματος. Οι τύποι που επιστρέφονται προστίθενται ως υποκόμβοι με `Tag` που ορίζεται ως **'Type'**.

```
Case "Graph"

    If node.Nodes.Count = 1 AndAlso node.Nodes(0).Text = "dummy" Then

        node.Nodes.Clear()



        Dim graph As String = node.Text

        Dim query As String =

            $"SELECT DISTINCT ?type

            WHERE {{

              GRAPH <{graph}> {{

                ?subject a ?type .

              }}

            }}
```

```
            ORDER BY ?type"


        Dim results As List(Of Dictionary(Of String, String)) =
RunSparqlQuery(query)


        For Each row As Dictionary(Of String, String) In results

            If row.ContainsKey("type") Then

                Dim type As String = row("type")

                Dim typeNode As TreeNode = node.Nodes.Add(type)

                typeNode.Tag = "Type" ' Επίπεδο: Type

                typeNode.Nodes.Add("dummy") ' Dummy κόμβος για επόμενο επίπεδο

            End If

        Next

    End If
```

*Λειτουργία:*

- Όταν ο χρήστης επεκτείνει τον κόμβο **Graph**, το query ανακτά όλους τους τύπους που σχετίζονται με τα subjects του γραφήματος.
- Οι τύποι προστίθενται ως κόμβοι με **Tag** = "Type".
- Κάθε τύπος περιέχει έναν **dummy** κόμβο για το επόμενο επίπεδο (Predicate).

Όταν επεκτείνεται ένας κόμβος **Predicate**, εκτελείται ένα **SPARQL query** για να ανακτηθούν τα **objects** που συνδέονται με αυτό το predicate. Τα **objects** προστίθενται ως υποκόμβοι με `Tag` που ορίζεται ως '**Object**'. Δεδομένου ότι αυτό είναι το τελευταίο επίπεδο, δεν προστίθεται νέος **dummy** κόμβος.

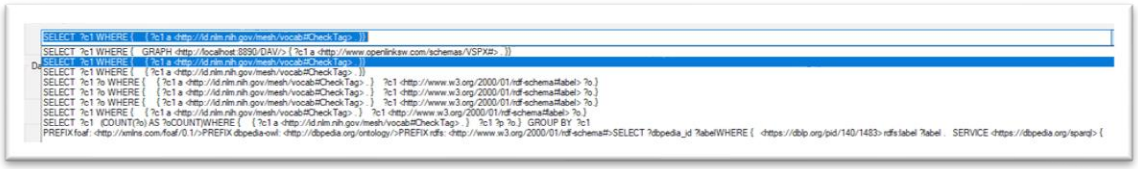
```
Case "Predicate"

    If node.Nodes.Count = 1 AndAlso node.Nodes(0).Text = "dummy" Then

        node.Nodes.Clear()


        Dim graph As String = node.Parent.Parent.Text

        Dim type As String = node.Parent.Text

        Dim predicate As String = node.Text

        Dim query As String =

            $"SELECT DISTINCT ?object
```

```
            WHERE {{
              GRAPH <{graph}> {{
                ?subject a <{type}> ;
                <{predicate}> ?object .
              }}
            }}
            ORDER BY ?object"

        Dim results As List(Of Dictionary(Of String, String)) = RunSparqlQuery(query)

        For Each row As Dictionary(Of String, String) In results
            If row.ContainsKey("object") Then
                Dim [object] As String = row("object")
                Dim objectNode As TreeNode = node.Nodes.Add([object])
                objectNode.Tag = "Object" ' Επίπεδο: Object
            End If
        Next
    End If
```

*Λειτουργία:*

- Όταν ο χρήστης επεκτείνει τον κόμβο **Predicate**, το query ανακτά τα **objects** που σχετίζονται με το συγκεκριμένο **predicate**.
- Τα **objects** προστίθενται ως κόμβοι με **Tag** = "Object".
- Δεν προστίθεται **dummy** κόμβος καθώς δεν υπάρχει επόμενο επίπεδο.

## 6.4 Module Query Manager



Εικόνα 48. Αποθηκευμένα Εωτήματα

Το **Module Query Manager** παρέχει λειτουργικότητα για τη διαχείριση ερωτημάτων **SPARQL** σε ένα αρχείο κειμένου. Στόχος του είναι να επιτρέπει την αποθήκευση νέων ερωτημάτων, καθώς και την ανάκτηση αποθηκευμένων ερωτημάτων από το αρχείο. Τα ερωτήματα αποθηκεύονται με τη χρήση ενός ειδικού χαρακτήρα διαχωρισμού, προκειμένου να διασφαλιστεί η σωστή αποθήκευση και ανάκτηση τους.

### Περιγραφή Λειτουργιών

*1. Αρχείο Αποθήκευσης*

Το module χρησιμοποιεί μία σταθερή τιμή (filePath) για το μονοπάτι του αρχείου που θα αποθηκεύονται τα ερωτήματα. Το μονοπάτι αυτό μπορεί να προσαρμοστεί ανάλογα με τις ανάγκες. Αν το αρχείο δεν υπάρχει κατά την έναρξη της εφαρμογής, δημιουργείται αυτόματα.

```
Private ReadOnly filePath As String = "queries.txt"
```

*2. Χρήση Ειδικού Χαρακτήρα Διαχωρισμού*

Για τον διαχωρισμό των ερωτημάτων μέσα στο αρχείο, χρησιμοποιείται ο ειδικός χαρακτήρας '###'. Αυτός ο χαρακτήρας εξασφαλίζει ότι τα ερωτήματα αποθηκεύονται με σαφήνεια και ανακτώνται σωστά.

```
Private ReadOnly delimiter As String = "###"
```

*3. Μέθοδος EnsureFileExists*

Η μέθοδος αυτή διασφαλίζει ότι το αρχείο κειμένου υπάρχει πριν εκτελεστεί οποιαδήποτε λειτουργία. Αν το αρχείο δεν υπάρχει, δημιουργείται ένα νέο κενό αρχείο.

```
Private Sub EnsureFileExists()

    If Not File.Exists(filePath) Then

        File.Create(filePath).Dispose()
```

```
    End If

End Sub
```

*4. Μέθοδος SaveQuery*

Η μέθοδος αποθηκεύει ένα νέο ερώτημα στο αρχείο, εφόσον το ερώτημα δεν υπάρχει ήδη στη λίστα των αποθηκευμένων ερωτημάτων. Χρησιμοποιεί τη μέθοδο LoadQueries για να ελέγξει αν το ερώτημα υπάρχει ήδη στο αρχείο. Εάν το ερώτημα υπάρχει, η μέθοδος εξέρχεται χωρίς να αποθηκεύσει τίποτα.

```
Public Sub SaveQuery(query As String)

    EnsureFileExists()

    Dim existingQueries As List(Of String) = LoadQueries()

    If existingQueries.Contains(query) Then

        Exit Sub

    End If

    Using writer As StreamWriter = New StreamWriter(filePath, append:=True)

        writer.WriteLine(query & delimiter)

    End Using

End Sub
```

*5. Μέθοδος LoadQueries*

Η μέθοδος αυτή διαβάζει το περιεχόμενο του αρχείου και διαχωρίζει τα ερωτήματα με τη βοήθεια του ειδικού χαρακτήρα διαχωρισμού ('###'). Επιστρέφει μία λίστα με τα αποθηκευμένα ερωτήματα.

```
Public Function LoadQueries() As List(Of String)

    EnsureFileExists()

    Dim queries As New List(Of String)

    Using reader As StreamReader = New StreamReader(filePath)

        Dim content As String = reader.ReadToEnd()

        Dim lines As String() = content.Split(New String() {delimiter},
StringSplitOptions.RemoveEmptyEntries)

        queries.AddRange(lines)
```

```
    End Using

    Return queries

End Function
```

*6. Αποθήκευση Κατά την Εκτέλεση*

Η αποθήκευση ενός ερωτήματος πραγματοποιείται κάθε φορά πριν την εκτέλεση του ερωτήματος (εφόσον δεν υπάρχει ήδη αποθηκευμένο). Ακολουθεί ένα παράδειγμα χρήσης του module για την αποθήκευση και ανάκτηση ερωτημάτων:

```
' Παράδειγμα χρήσης του module

Public Class QueryHandler

    Private Sub btnTestQuery_Click(sender As Object, e As EventArgs) Handles
btnTestQuery.Click

        If selectString = "" Then

            MsgBox("Δεν έχετε επιλέξει μεταβλητές στο Select")

            Exit Sub

        End If



        Dim f As New Results

        f.resultList = RunSparqlQuery(rtxtQuery.Text)



        ' Αποθήκευση του ερωτήματος στο αρχείο

        QueryManager.SaveQuery(rtxtQuery.Text)



        If f.resultList.Count = 0 Then

            MsgBox("No results found")

        End If

        f.ShowDialog()

    End Sub



    Private Sub LoadSavedQueries()
```

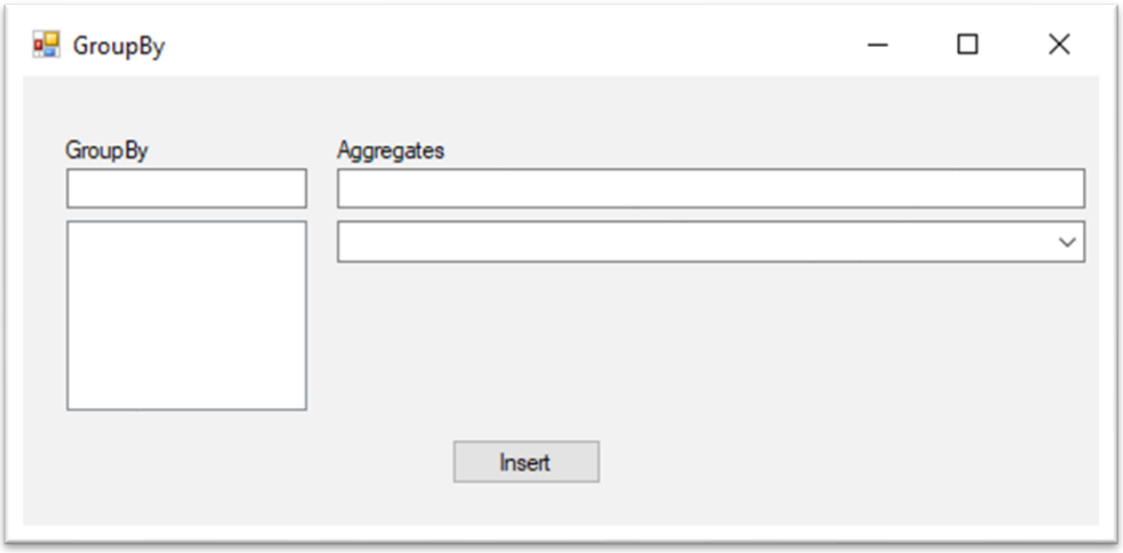
```
        ' Φόρτωση των αποθηκευμένων ερωτημάτων σε λίστα

        Dim savedQueries As List(Of String) = QueryManager.LoadQueries()



        ' Ενημέρωση του control (π.χ., ListBox)

        lstQueries.Items.Clear()

        lstQueries.Items.AddRange(savedQueries.ToArray())

    End Sub

End Class
```

**Σημαντικά Σημεία:**

- **Αρχείο Ερωτημάτων:** Τα ερωτήματα αποθηκεύονται στο αρχείο queries.txt, το οποίο μπορεί να προσαρμοστεί.
- **Διαχωρισμός Ερωτημάτων:** Τα ερωτήματα διαχωρίζονται με τον χαρακτήρα '###'.
- **Αναγνώριση και Αποθήκευση:** Το σύστημα ελέγχει αν το ερώτημα έχει αποθηκευτεί ήδη πριν το αποθηκεύσει ξανά.
- **Ανάκτηση Ερωτημάτων:** Η μέθοδος LoadQueries επιστρέφει μία λίστα από τα αποθηκευμένα ερωτήματα, η οποία μπορεί να χρησιμοποιηθεί για την εμφάνιση των ερωτημάτων στην εφαρμογή.

Αυτή η προσέγγιση εξασφαλίζει ότι τα ερωτήματα διατηρούνται οργανωμένα και ανακτώνται εύκολα όταν χρειαστεί, ενώ επιτρέπει την αποθήκευση τους με ασφαλή και αποτελεσματικό τρόπο.

## 6.5 Κώδικας GroupBy




Εικόνα 49. Φόρμα Group By

Η τεκμηρίωση του κώδικα για την κλάση **Group By** περιγράφει τη διαδικασία διαχείρισης του τμήματος **GROUP BY** και των συναρτήσεων συγκέντρωσης (aggregate functions) σε ερωτήματα SPARQL. Ο κώδικας επιτρέπει στον χρήστη να επιλέξει μεταβλητές, να προσθέσει συναρτήσεις συγκέντρωσης (όπως COUNT, SUM, AVG, κ.λπ.), και να ενσωματώσει αυτές τις πληροφορίες σε ένα ερώτημα SPARQL.

1. Διαχείριση διπλού κλικ στη λίστα μεταβλητών (lstGroupBy)

```
Private Sub lstGroupBy_DoubleClick(sender As Object, e As EventArgs) Handles
lstGroupBy.DoubleClick

    txtGroupBy.Text = txtGroupBy.Text + " " + lstGroupBy.SelectedItem

End Sub
```

Ο χρήστης προσθέτει μεταβλητές από τη λίστα lstGroupBy στο πεδίο κειμένου txtGroupBy με διπλό κλικ. Αυτό επιτρέπει την επιλογή μεταβλητών που θα συμπεριληφθούν στη δήλωση **GROUP BY**.

2. Φόρτωση των συναρτήσεων συγκέντρωσης στο ComboBox (GroupBy_Load)

```
Private Sub GroupBy_Load(sender As Object, e As EventArgs) Handles Me.Load

    cmbAggregate.Items.Clear()

    Dim aggregateFunctions As String() = {"COUNT", "SUM", "AVG", "MIN", "MAX"}

    For Each variableName As String In lstGroupBy.Items

        For Each func As String In aggregateFunctions
```

```
            Dim aggregateOption As String = $"({func}({variableName}) AS
{variableName}{func})"

            cmbAggregate.Items.Add(aggregateOption)

        Next

    Next

End Sub
```

Κατά την φόρτωση της φόρμας, οι συναρτήσεις συγκέντρωσης (COUNT, SUM, AVG, MIN, MAX) φορτώνονται στο ComboBox cmbAggregate. Για κάθε μεταβλητή της λίστας lstGroupBy, δημιουργούνται επιλογές συνάρτησης με τη μορφή `(SUM(?var) AS ?varSUM)`.

3. Αποθήκευση της δήλωσης GROUP BY (btnSave_Click)

```
Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles
btnSave.Click

    groupByString = "  GROUP BY " + txtGroupBy.Text

    aggregateSelect = txtAggregates.Text + vbCrLf

    newQueryForm.rtxtQuery.Text = prefixSelect + aggregateSelect + fromWhere +
statementsString + closeQuery + groupByString

    Me.Hide()

End Sub
```

Η μέθοδος αυτή αποθηκεύει την τελική δήλωση **GROUP BY** και τις συναρτήσεις συγκέντρωσης στο ερώτημα SPARQL. Ενημερώνει το πεδίο rtxtQuery στη φόρμα newQueryForm με το πλήρες ερώτημα.

4. Ενημέρωση συναρτήσεων συγκέντρωσης (cmbAggregate_SelectedValueChanged)

```
Private Sub cmbAggregate_SelectedValueChanged(sender As Object, e As EventArgs)
Handles cmbAggregate.SelectedValueChanged

    txtAggregates.Text = txtAggregates.Text + " " + cmbAggregate.Text

End Sub
```

Όταν ο χρήστης επιλέγει μια συνάρτηση από το `cmbAggregate`, αυτή προστίθεται στο πεδίο `txtAggregates`, το οποίο αποθηκεύει όλες τις συναρτήσεις συγκέντρωσης που θα ενσωματωθούν στο ερώτημα **SELECT** του SPARQL.

## 5. Άνοιγμα φόρμας GroupBy και μεταφορά μεταβλητών (btnGroupBy_Click)

```
Private Sub btnGroupBy_Click(sender As Object, e As EventArgs) Handles btnGroupBy.Click

    Dim newGroupBy As New GroupBy

    For Each item As Object In lstSelectList.Items

        newGroupBy.lstGroupBy.Items.Add(item)

    Next

    newGroupBy.Show()

End Sub
```

Η μέθοδος αυτή δημιουργεί μια νέα φόρμα `GroupBy`, όπου οι μεταβλητές από την υπάρχουσα λίστα `lstSelectList` μεταφέρονται στη λίστα `lstGroupBy` της νέας φόρμας για να χρησιμοποιηθούν στο GROUP BY.

**Γενική Λογική της Κλάσης**

1. **Δημιουργία και Επεξεργασία Δεδομένων για GROUP BY:** Ο χρήστης επιλέγει μεταβλητές από μια λίστα και τις προσθέτει στη δήλωση GROUP BY.
2. **Εφαρμογή Συναρτήσεων Συγκέντρωσης:** Παρέχεται η δυνατότητα εφαρμογής συναρτήσεων συγκέντρωσης (COUNT, SUM, κ.λπ.) στις μεταβλητές, με έτοιμες επιλογές μέσω του ComboBox.
3. **Ενημέρωση του Ερωτήματος SPARQL:** Το τελικό ερώτημα SPARQL ενημερώνεται δυναμικά με βάση τις επιλογές του χρήστη.

Αυτή η κλάση παρέχει στον χρήστη μια ευέλικτη και δυναμική μέθοδο για τη δημιουργία και επεξεργασία SPARQL ερωτημάτων με χρήση της δήλωσης **GROUP BY** και συναρτήσεων συγκέντρωσης.

## 6.6 Διαχείριση Ρυθμίσεων

Εικόνα 50. Μορφή αρχείου ρυθμίσεων

Το `JsonSettingsManager` Module διαχειρίζεται τις ρυθμίσεις του χρήστη μέσω ενός αρχείου JSON και παρέχει εύκολες λειτουργίες για την αποθήκευση, ανάγνωση και επεξεργασία των

δεδομένων αυτών. Εδώ είναι η τεκμηρίωση για τις λειτουργίες και τη δομή του αρχείου ρυθμίσεων:

**Δημιουργία και Έλεγχος Υπάρχοντος Αρχείου Ρυθμίσεων:** EnsureSettingsFileExists

Η μέθοδος `EnsureSettingsFileExists` διασφαλίζει ότι το αρχείο `settings.json` υπάρχει στον φάκελο της εφαρμογής. Αν το αρχείο δεν υπάρχει, δημιουργείται με προκαθορισμένες ρυθμίσεις (όπως για παράδειγμα η λίστα με τα SPARQL end points).

```
Public Sub EnsureSettingsFileExists()

    If Not File.Exists(settingsFilePath) Then

        ' Δημιουργεί το αρχείο με προεπιλεγμένες ρυθμίσεις

        Dim defaultSettings As New Dictionary(Of String, Object) From {

            {"EndPoints", New List(Of String) From
{"https://sparql.dblp.org/sparql", "https://dbpedia.org/sparql"}}

        }

        SaveSettings(defaultSettings)

    End If

End Sub
```

**Λειτουργία:**

- Ελέγχει αν το αρχείο settings.json υπάρχει.
- Αν δεν υπάρχει, δημιουργεί το αρχείο με τις προκαθορισμένες ρυθμίσεις για τους SPARQL end points.

**Ανάγνωση Αρχείου Ρυθμίσεων:** LoadSettings

Η μέθοδος `LoadSettings` διαβάζει το περιεχόμενο του αρχείου JSON και το μετατρέπει σε ένα αντικείμενο τύπου `Dictionary(Of String, Object)`.

```
Public Function LoadSettings() As Dictionary(Of String, Object)

    EnsureSettingsFileExists()

    Try

        Dim json As String = File.ReadAllText(settingsFilePath)

        Dim settings As Dictionary(Of String, Object) =
JsonConvert.DeserializeObject(Of Dictionary(Of String, Object))(json)

        Return settings

    Catch ex As Exception
```

```
        MessageBox.Show("Σφάλμα κατά την φόρτωση των ρυθμίσεων: " & ex.Message,
"Σφάλμα", MessageBoxButtons.OK, MessageBoxIcon.Error)

        Return New Dictionary(Of String, Object)

    End Try

End Function
```

**Λειτουργία:**

- Εξασφαλίζει ότι το αρχείο υπάρχει μέσω της EnsureSettingsFileExists.
- Διαβάζει το αρχείο JSON και το μετατρέπει σε αντικείμενο τύπου Dictionary.
- Επιστρέφει το αντικείμενο Dictionary με τα δεδομένα.

**Αποθήκευση Ρυθμίσεων:** SaveSettings

Η μέθοδος `SaveSettings` μετατρέπει το αντικείμενο `Dictionary` σε μορφή JSON και το γράφει στο αρχείο `settings.json`.

```
Public Sub SaveSettings(settings As Dictionary(Of String, Object))

    Try

        Dim json As String = JsonConvert.SerializeObject(settings,
Formatting.Indented)

        File.WriteAllText(settingsFilePath, json)

    Catch ex As Exception

        MessageBox.Show("Σφάλμα κατά την αποθήκευση των ρυθμίσεων: " &
ex.Message, "Σφάλμα", MessageBoxButtons.OK, MessageBoxIcon.Error)

    End Try

End Sub
```

**Λειτουργία:**

- Μετατρέπει το αντικείμενο `Dictionary` σε μορφή JSON.
- Αποθηκεύει το JSON στο αρχείο `settings.json`.

**Αλληλεπίδραση με ComboBox:** LoadComboBoxFromList

Η μέθοδος `LoadComboBoxFromList` φορτώνει τιμές από μια λίστα που αποθηκεύεται στο αρχείο ρυθμίσεων και τις προσθέτει στο ComboBox.

```
Public Sub LoadComboBoxFromList(cmb As ComboBox, key As String)

    Dim settings As Dictionary(Of String, Object) = LoadSettings()



    If settings.ContainsKey(key) Then
```

```
        Dim list As List(Of String) = JsonConvert.DeserializeObject(Of List(Of
String))(settings(key).ToString())

        cmb.Items.Clear()

        cmb.Items.AddRange(list.ToArray())

    End If

End Sub
```

**Λειτουργία:**

- Διαβάζει τη λίστα από το αρχείο ρυθμίσεων για το δεδομένο key.
- Προσθέτει τις τιμές από τη λίστα στο ComboBox.

**Δομή του Αρχείου Ρυθμίσεων**

Το αρχείο `settings.json` χρησιμοποιεί τη μορφή κλειδιών και τιμών (key-value) ή λιστών. Ένα παράδειγμα της δομής του αρχείου είναι:

```
{

  "EndPoints": [

    "https://sparql.dblp.org/sparql",

    "https://dbpedia.org/sparql"

  ]

}
```

Αυτό το παράδειγμα περιέχει το κλειδί `EndPoints`, το οποίο είναι μια λίστα από URLs που αναφέρονται σε SPARQL end points. Το αρχείο μπορεί να περιέχει περισσότερα κλειδιά, όπως `"FavoriteQueries"`, για άλλες ρυθμίσεις ή δεδομένα χρήστη.

Το `JsonSettingsManager` παρέχει εύκολη διαχείριση των ρυθμίσεων χρήστη μέσω ενός αρχείου JSON, επιτρέποντας την αποθήκευση, φόρτωση, και επεξεργασία των δεδομένων. Οι λειτουργίες του είναι χρήσιμες για την αλληλεπίδραση με τα στοιχεία της εφαρμογής, όπως τα ComboBox, και προσφέρουν ευελιξία στη διαχείριση των ρυθμίσεων της εφαρμογής.

# Κεφάλαιο 7: Συμπεράσματα και Μελλοντικές Επεκτάσεις

Η παρούσα διπλωματική εργασία ασχολείται με την ανάπτυξη μιας εφαρμογής ανοικτού κώδικα σε περιβάλλον .NET, η οποία επιτρέπει την οπτικοποίηση και τη διαχείριση SPARQL ερωτημάτων με εύχρηστο τρόπο. Η εφαρμογή αυτή αποσκοπεί στην απλοποίηση της διαδικασίας δημιουργίας και εκτέλεσης SPARQL ερωτημάτων, επιτρέποντας ακόμα και σε χρήστες χωρίς προηγούμενη εμπειρία στη γλώσσα να την χρησιμοποιούν αποτελεσματικά. Ο σκοπός της εργασίας ήταν να γεφυρώσει το χάσμα μεταξύ της τεχνικής φύσης της SPARQL και της ανάγκης για εύχρηστα και κατανοητά εργαλεία, προσφέροντας μια φιλική πλατφόρμα χρήσης.

**Κύριες Δυνατότητες της Εφαρμογής:**

- **Διαχείριση End points**: Η εφαρμογή επιτρέπει στους χρήστες να επιλέγουν και να διαχειρίζονται SPARQL end points, με τις προσωπικές τους ρυθμίσεις να αποθηκεύονται για μελλοντική χρήση.
- **Γραφήματα RDF**: Παρουσίαση των δεδομένων σε δενδρική μορφή, βοηθώντας τον χρήστη να κατανοήσει καλύτερα τη δομή των RDF δεδομένων.
- **Δημιουργία SPARQL Ερωτημάτων**: Μέσω ενός φιλικού προς τον χρήστη περιβάλλοντος, οι χρήστες μπορούν να επιλέξουν predicates, subjects και objects για τη σύνθεση SPARQL ερωτημάτων.
- **Αποθήκευση και Επαναχρησιμοποίηση Ερωτημάτων**: Η δυνατότητα αποθήκευσης των δημιουργημένων ερωτημάτων διευκολύνει την επαναχρησιμοποίηση τους σε μελλοντικές αναλύσεις.
- **Εξαγωγή Δεδομένων**: Οι χρήστες μπορούν να εξάγουν τα αποτελέσματα των ερωτημάτων τους σε διάφορες μορφές, όπως κείμενο, DataGrid ή αρχεία Excel.
- **Συγκεντρωτικές Συναρτήσεις**: Η εφαρμογή υποστηρίζει τη χρήση συγκεντρωτικών συναρτήσεων, παρέχοντας μεγαλύτερη ευχέρεια στην ανάλυση των δεδομένων.

Η συνεισφορά της εργασίας εντοπίζεται σε πολλαπλά επίπεδα. Αρχικά, η εφαρμογή καθιστά την διαδικασία δημιουργίας και διαχείρισης SPARQL ερωτημάτων πιο προσιτή και αποτελεσματική για χρήστες όλων των επιπέδων. Παράλληλα, η εφαρμογή αυτή ενισχύει τη χρηστικότητα των SPARQL end points και προσφέρει μια πλατφόρμα για τη βελτίωση της αποδοτικότητας στη διαχείριση των RDF δεδομένων.

**Μελλοντικές Επεκτάσεις**

Παρόλο που η εφαρμογή προσφέρει ένα ισχυρό και ευέλικτο σύνολο λειτουργιών, υπάρχουν αρκετές περιοχές για περαιτέρω βελτίωση και εξέλιξη:

1. **Ενσωμάτωση Προηγμένων Γραφικών Αναπαραστάσεων**: Η προσθήκη διαδραστικών γραφημάτων και της αναπαράστασης σχέσεων μεταξύ των κόμβων μέσω διαδραστικών χάρτη μπορεί να διευκολύνει σημαντικά την κατανόηση των δεδομένων RDF, κάνοντάς τα πιο προσιτά.
2. **Υποστήριξη Περισσότερων Εξαγωγών Δεδομένων**: Εκτός από την υποστήριξη εξαγωγής σε μορφή Excel, θα μπορούσε να προστεθεί υποστήριξη για άλλες μορφές δεδομένων, όπως JSON, XML και RDF. Αυτή η δυνατότητα θα προσφέρει μεγαλύτερη ευχέρεια στους χρήστες που επιθυμούν να δουλέψουν με διαφορετικά formats.
3. **Προσαρμογή Διεπαφής για Κινητές Συσκευές**: Η ανάπτυξη μιας έκδοσης της εφαρμογής για φορητές συσκευές ή η μετάβαση σε μια web-based εφαρμογή θα επεκτείνει

τη χρηστικότητα του εργαλείου, προσφέροντας στους χρήστες την δυνατότητα να το χρησιμοποιούν σε οποιοδήποτε περιβάλλον.

4. **Υλοποίηση Αυτόματης Πρόβλεψης Ερωτημάτων**: Η χρήση τεχνητής νοημοσύνης για την πρόβλεψη πιθανών ερωτημάτων βάσει των δεδομένων που παρέχονται ή των προτιμήσεων του χρήστη θα μπορούσε να προσφέρει μια εξαιρετικά βελτιωμένη εμπειρία χρήστη. Αυτό θα επιτρέψει στους χρήστες να προχωρούν πιο γρήγορα στη σύνθεση των ερωτημάτων τους, ακόμα και αν δεν είναι έμπειροι με τη γλώσσα SPARQL.

Συνοψίζοντας, η εργασία αυτή αποτελεί ένα σημαντικό βήμα για τη διευκόλυνση της χρήσης της SPARQL και την καλύτερη αξιοποίηση των RDF δεδομένων. Η ανάπτυξη αυτής της εφαρμογής προσφέρει έναν πολύτιμο μηχανισμό για τη βελτίωση της αποδοτικότητας στη διαχείριση δεδομένων και ανοίγει νέους δρόμους για τη χρήση αυτών των εργαλείων σε επιστημονικά, επιχειρηματικά και εκπαιδευτικά περιβάλλοντα.

Επιπλέον, το γεγονός ότι η εφαρμογή είναι **λογισμικό ανοικτού κώδικα** (https://github.com/bekdim072/SparqlQueryBuilder.git) επιτρέπει σε προγραμματιστές και ερευνητές να συμβάλλουν στη συνεχή ανάπτυξή της, βελτιώνοντας τις δυνατότητές της και επεκτείνοντάς τη για νέες εφαρμογές. Αυτό το έργο μπορεί να αποτελέσει τη βάση για περαιτέρω εξελίξεις στον τομέα της διαχείρισης δεδομένων RDF και των εργαλείων που διευκολύνουν τη χρήση της SPARQL, επεκτείνοντας τις δυνατότητες και τις εφαρμογές σε ένα ευρύ φάσμα τομέων.

# Βιβλιογραφία


[1] B. DuCharme, *Learning SPARQL: A practical guide for querying and updating RDF data*. O'Reilly Media, Inc., 2009.

[2] P. Bouros, S. Athanassopoulos, and N. Mamoulis, "SPARQL Query Optimization on Top of DHTs," in *Proc. 9th Int. Symp. Web and Wireless Geographical Information Systems (W2GIS 2009)*, Springer, 2009, pp. 123-134.

[3] D. Zhang and L. Gruenwald, "SPARQL Query Parallel Processing: A Survey," in *Proc. Int. Conf. Database and Expert Systems Applications (DEXA 2011)*, Springer, 2011, pp. 197-211.

[4] G. Antoniou and F. van Harmelen, *A Semantic Web Primer*. MIT Press, 2004.

[5] W3C, *SPARQL 1.1 Query Language*, World Wide Web Consortium, 2013. [Online]. Available: https://www.w3.org/TR/sparql11-query/.

[6] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data – The Story So Far," *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 3, 2009.

[7] S. Harris and A. Seaborne, *SPARQL 1.1 Querying with SPARQL*. O'Reilly Media, 2013.

[8] R. Cyganiak, D. Wood, and M. Lanthier, *RDF 1.1 Concepts and Abstract Syntax*, W3C, 2014.

[9] J. Pérez, M. Arenas, and C. Gutierrez, "Semantics and Complexity of SPARQL," in *Proc. ISWC*, 2006.

[10] O. Hartig and S. Auer, *SPARQL for the Web of Data*, Springer, 2011.

[11] J. Broekstra, A. Kampman, and F. van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema," *Int. J. Digital Libraries*, 2002.

[12] D. Booth and J. Diamond, *Building the Web of Data*, O'Reilly Media, 2015.

[13] E. Tanter and F. Wimm, *Semantic Web Technologies*, Wiley, 2012.

[14] M. Rys, "SPARQL Querying and SPARQL Performance," Springer, 2011.

[15] W3C, *SPARQL Protocol and RDF Query Language (SPARQL)*, World Wide Web Consortium, 2025. [Online]. Available: https://www.w3.org/TR/sparql11-query/.

[16] DBpedia, 2025. [Online]. Available: https://www.dbpedia.org/.

[17] Wikidata, 2025. [Online]. Available: https://www.wikidata.org/.