ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι.

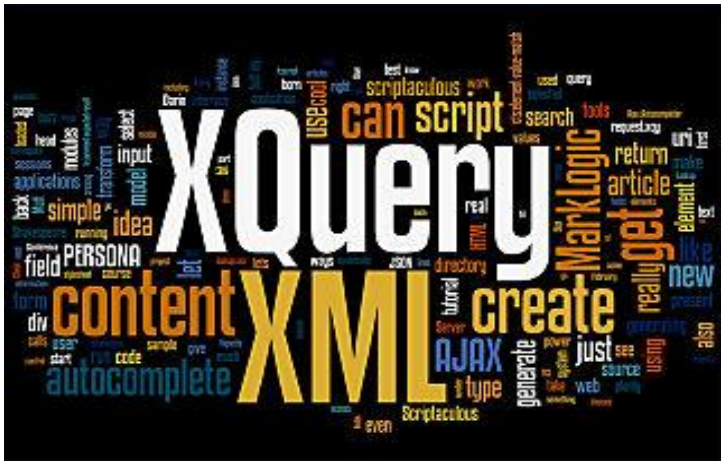
ΘΕΣΣΑΛΟΝΙΚΗΣ ΣΧΟΛΗ

ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

Υλοποίηση εργαλείου γραφικής απεικόνισης της XQuery



Των φοιτητών

Πλιάκα Αχιλλέα
Τσέκου Κων/νου

Αρ. Μητρώου: 2077 - 2042

Επιβλέπων καθηγητής

Ευκλείδης Κεραμόπουλος

Θεσσαλονίκη
2010

Πρόλογος

Όπως όλοι γνωρίζουμε η πτυχιακή είναι ένα από τα σημαντικότερα έργα στην πορεία κάθε ανθρώπου σαν φοιτητής. Από τα πρώτα εξάμηνα δείξαμε ιδιαίτερο ενδιαφέρον για τις βάσεις δεδομένων και γενικά για τον προγραμματισμό πάνω σε αυτές. Όποιο μάθημα υπήρχε στην σχολή μας που είχε σαν θέμα τις βάσεις δεδομένων επιλέξαμε να το παρακολουθήσουμε. Σαν αποτέλεσμα και η πτυχιακή μας δεν θα μπορούσε να είχε θέμα που να μην αναφέρεται σε βάσεις δεδομένων.

Σημαντικός παράγοντας στην επιλογή της πτυχιακής μας ήταν ο εισηγητής καθηγητής μας κ. Ευκλείδης Κεραμόπουλος τον οποίο θέλουμε να ευχαριστήσουμε για την αμέριστη συμπαράστασή του τονίζοντας ότι όποτε χρειαζόμασταν βοήθεια ήταν πάντα δίπλα μας.

Περίληψη

Σκοπός της πτυχιακής μας είναι η παρουσίαση της γλώσσας XQuery αναφέροντας ιστορικά στοιχεία για το πότε και από ποιους δημιουργήθηκε και για ποιούς λόγους. Επίσης γίνεται αναφορά στις δυνατότητες της XQuery τους λόγους που μπορεί να χρησιμοποιηθεί και συγκρίνεται με την γλώσσα SQL. Στη συνέχεια αναφέρουμε τα δομικά στοιχεία της καθώς και τον τρόπο σύνταξής της. Για να γίνει πιο κατανοητή η γλώσσα XQuery δημιουργήσαμε metaphors τα οποία έχουν σαν θέμα την οδική κυκλοφορία. Σε επίπεδο προγραμματισμού δημιουργήσαμε εφαρμογή με την οποία ακόμα και ένας χρήστης ο οποίο δεν έχει και τις πιο σύνθετες γνώσεις να μπορεί να συντάξει τα δικά του ερωτήματα, να τα τρέξει στο περιβάλλον της DB2 και να αποθηκεύσει τα αποτελέσματα του ερωτήματός του σε ένα xml αρχείο. Η εφαρμογή μας έχει δημιουργηθεί στο πρόγραμμα NetBeans.

Preview

The objective of our final year project is the presentation of XQuery language, referring to historical facts about when and by who has been created and for what reason. In addition, there is a reference to the capabilities of XQuery and the reasons that can be used. A comparison between XQuery and SQL took place, mentioning the advantages and the disadvantages. Furthermore, we introduce the structure of XQuery and it's syntax. We created metaphors which their theme is from road circulation. Considering that it isn’t easy for a naive user to use XQuery, we created a program that gives the capability in a user to define graphically his query and execute it in the platform of IBM DB2, saving the results in a xml file. Finally our application has been created in NetBeans program.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή στο θέμα της πτυχιακής μας

Σε αυτό το κεφάλαιο θα κάνουμε μια μικρή εισαγωγή στα βασικά στοιχεία της πτυχιακής μας αναφέροντας το θέμα της και τον κύριο στόχο της καθώς και το πώς υλοποιήθηκε.

1.1 Στόχοι της πτυχιακής μας

Κεντρικό θέμα της πτυχιακής μας είναι η γραφική αναπαράσταση της γλώσσα XQuery. Χρησιμοποιώντας metaphors τα οποία είναι εμπνευσμένα από την οδική κυκλοφορία αναπαριστούμε τα βασικά στοιχεία της XQuery. Τα metaphors βοηθούν στον χρήστη να καταλάβει την λειτουργία που επιτελεί κάθε στοιχείο της XQuery μελετώντας την αντίστοιχη λειτουργία που επιτελεί το metaphor στην οδική κυκλοφορία.
Ακόμα εκτός από την γραφική αναπαράσταση της XQuery αναφέρουμε όλα τα απαραίτητα στοιχεία της XQuery όπως οι λόγοι της δημιουργίας της, ιστορικά στοιχεία, τρόποι σύνταξης και γενικά ό,τι απαιτείται για να μπορέσει να γίνει κατανοητή από έναν χρήστη.
Επίσης για να μπορέσουμε να κάνουμε πιο εύκολη την κατανόηση της XQuery δημιουργήσαμε εφαρμογή η οποία επιτρέπει σε έναν απλό χρήστη να μπορέσει να συντάξει τα δικά του ερωτήματα, να εμφανίσει τα αποτελέσματα σε αρχείο xml και να μπορέσει αυτά τα αποτελέσματα να τα επεξεργαστεί.

1.2 Υλοποίηση της εφαρμογής

Για να μπορέσουμε να δημιουργήσουμε την εφαρμογή μας, εργαστήκαμε στο πρόγραμμα Netbeans. Επιλέξαμε αυτό το πρόγραμμα γιατί δίνει την δυνατότητα στον χρήστη να γράψει όσο τον δυνατόν λιγότερο κώδικα παρέχοντας έτοιμα κομμάτια μπλοκ κώδικα. Έτσι σε αντίθεση με άλλα προγράμματα ο χρήστης δεν χρειάζεται να μπει στην διαδικασία να γράψει κώδικα για το κάθε στοιχείο που υπάρχει στην εφαρμογή του, αλλά παρέχονται παλέτες στις οποίες υπάρχουν λειτουργικά στοιχεία όπως κουμπιά, εργαλειοθήκες κ.α. που μόλις ο χρήστης επιλέξει κάποια από αυτά δημιουργείται αμέσως το αντίστοιχο μπλοκ κώδικα στο script της εφαρμογής γλιτώνοντας τον χρήστη από την συγγραφή τεράστιων κομματιών κώδικα.
Η εφαρμογή μας επικοινωνεί με την DB2 στην οποία αποστέλλονται τα ερωτήματα που συντάσσει ο χρήστης, η DB2 μετά με την σειρά της αποστέλλει τα αποτελέσματα του ερωτήματος στην εφαρμογή μας και αυτή τα αποθηκεύει σε ένα αρχείο xml το οποίο ανοίγει και εμφανίζεται στον χρήστη αυτόματα.
Το αρχείο xml το οποίο περιέχει τα αποτελέσματα του ερωτήματος ανοίγει με το πρόγραμμα Altova Xml Spy που μας δίνει την δυνατότητα να τα επεξεργαστούμε παρέχοντας μια σειρά από λειτουργίες.
Στα επόμενα κεφάλαια θα ακολουθήσει αναλυτική παρουσίαση του κάθε προγράμματος καθώς και των λειτουργιών του.

Κεφάλαιο 2

Εισαγωγή στην XQuery

Στόχος αυτού του κεφαλαίου είναι να δώσει στον αναγνώστη τον λόγο ύπαρξης καθώς και τις δυνατότητες της XQuery. Επίσης παρέχει μια πρώτη ματιά στα χαρακτηριστικά της XQuery τα οποία όμως καλύπτονται ενδελεχώς στα επόμενα κεφάλαια. Τέλος αναφέρονται στις πιο συνηθισμένες εκφράσεις, στις οποίες θα αναφερθούμε λεπτομερώς αργότερα.

2.1 Τι είναι η XQuery;

Τα τελευταία χρόνια η χρήση της XML έχει εξαπλωθεί. Τεράστιο πλήθος πληροφοριών αποθηκεύονται στην XML, τόσο σε βάσεις δεδομένων της καθώς και σε έγγραφα σε ένα σύστημα αρχείων. Αυτό περιλαμβάνει υψηλά δομημένα δεδομένα, όπως είναι διαγράμματα πωλήσεων, ημι-δομημένα δεδομένα όπως είναι κατάλογοι προϊόντων καθώς και συσχετιζόμενα αδόμητα δεδομένα όπως για παράδειγμα τα γράμματα και τα βιβλία. Όλο και περισσότερες πληροφορίες μεταβιβάζονται ανάμεσα σε συστήματα ως προσωρινά XML έγγραφα.

Όλα αυτά τα δεδομένα χρησιμοποιούνται για ποικίλους λόγους. Σε κάθε μία από τις χρήσεις, ενδιαφερόμαστε σε διαφορετικούς παράγοντες των δεδομένων και είναι απαραίτητο να σχηματοποιηθούν και να τροποποιηθούν κατάλληλα ώστε να καλύπτουν τις δικές μας ανάγκες.

Η XQuery είναι μία γλώσσα η οποία θέτει ερωτήματα και είναι σχεδιασμένη από το W3C ώστε να ικανοποιεί αυτές τις απαιτήσεις. Επιτρέπει στον χρήστη να επιλέξει τα XML δεδομένα τα οποία τον ενδιαφέρουν και επίσης του δίνει την δυνατότητα να τα αναδιοργανώσει και να τα μετατρέψει πάντα σύμφωνα με τις απαιτήσεις του.

Το 1999 ιδρύθηκε η ομάδα εργασίας της XQuery. Αυτήν η ομάδα εργασίας εργάστηκε πάνω στην ανάπτυξη και την αναβάθμιση της XQuery.
Τον Ιούνιο του 2001, δημιουργήθηκε η XQuery 1.0 WD (Working Data) και αποτελεί την πρώτη έκδοση της XQuery που παρουσιάσθηκε από την W3C.
Τον Ιανουάριο του 2007, η έκδοση 1.0 της XQuery γίνεται πραγματικότητα από την W3C, αποτελεί την πιο πρόσφατη και παράλληλα τη πιο διαδεδομένη έκδοση της XQuery η οποία υποστηρίζεται από διαφορετικά API όπως το Saxon και DataDirect XQuery.

2.2 Δυνατότητες της XQuery

Η XQuery έχει ένα μεγάλο πλήθος από χαρακτηριστικά τα οποία επιτρέπουν να γίνουν διαφόρων ειδών ενέργειες πάνω σε XML δεδομένα και έγγραφα περιλαμβανομένων των:

- Επιλογή πληροφοριών βασισμένων σε συγκεκριμένα κριτήρια
- Απόρριψη μη επιθυμητών πληροφοριών
- Έρευνα για πληροφορίες μέσα σε ένα έγγραφο ή κατάλογο εγγράφων
- Ένωση δεδομένων από πολλαπλά έγγραφα ή συλλογή εγγράφων
- Ταξινόμηση, κατηγοριοποίηση και συνάθροιση δεδομένων
- Μετατροπή και αλλαγή της δομής των XML δεδομένων σύμφωνα με κάποιο άλλο XML λεξιλόγιο ή δομή
- Υλοποίηση αριθμητικών υπολογισμών σε νούμερα και ημερομηνίες
- Επιδέξιος χειρισμός χαρακτήρων για την αναδιάρθρωση ενός κειμένου

Όπως μπορούμε να παρατηρήσουμε, η XQuery μπορεί να χρησιμοποιηθεί όχι μόνο για να εξάγει τμήματα των XML εγγράφων, αλλά και για να χειριστεί και να μετασχηματίσει τα αποτελέσματα. Μία δυνατότητα την οποία δεν παρέχει η XQuery 1.0 είναι το update, το οποίο είναι πολύ χρήσιμο και κυρίως στην περίπτωση της αποθήκευσης XML δεδομένων στις βάσεις δεδομένων. Γίνονται προσπάθειες αυτή η δυνατότητα να ενταχθεί σε μεταγενέστερες εκδόσεις της XQuery.

2.3 Χρήση της XQuery

Υπάρχουν πολλοί λόγοι για τους οποίους χρησιμοποιείται η XML. Μερικά παραδείγματα συνηθισμένων τρόπων χρήσεως της XQuery είναι:

- Εξαγωγή πληροφοριών από μια συσχετιζόμενη βάση δεδομένων για να χρησιμοποιηθούν σε μια υπηρεσία του διαδικτύου
- Παραγωγή αναφορών σχετικά με δεδομένα τα οποία είναι αποθηκευμένα σε βάση δεδομένων για παρουσίαση στο διαδίκτυο ως XHTML
- Έρευνα εγγράφων σε τοπικές XML βάσεις δεδομένων και παρουσίαση των αποτελεσμάτων
- Εξαγωγή δεδομένων από βάσεις δεδομένων ή πακέτα λογισμικών και μετατροπή τους σε αιτήσεις ολοκλήρωσης
- Συνδυασμός περιεχομένων από παραδοσιακές μη-XML πηγές σε εφαρμογές διαχείρισης και διανομής
- Επί τούτου θέτει ερωτήματα σε μοναδικά XML έγγραφα με σκοπό την έρευνα και την δοκιμή

2.4 Ο Σχεδιασμός της XQuery

Η ομάδα εργασίας της XML Query του World Wide Web Consortium (W3C), άρχισε να δουλεύει πάνω στην XQuery το 1999. Χρησιμοποιήθηκε στην αρχή ως μια XML query γλώσσα επηρεασμένη από δυο προηγούμενες γλώσσες: την XQL και την XML-QL.

Η ομάδα εργασίας ανέλαβε να σχεδιάσει μια γλώσσα η οποία:

- Θα ήταν χρήσιμη τόσο για δομημένα όσο και για ημιδομημένα έγγραφα
- Θα αποτελούσε ένα ανεξάρτητο πρωτόκολλο επιτρέποντας ένα ερώτημα να αξιολογηθεί σε οποιοδήποτε σύστημα με αναμενόμενα αποτελέσματα
- Θα αποτελούσε μια επεξηγηματική γλώσσα παρά μια διαδικαστική
- Να είναι αυστηρά τυποποιημένη, επιτρέποντας τα ερωτήματα να μεταγλωττιστούν ώστε να αναγνωριστούν πιθανά λάθη και να βελτιστοποιηθεί η αξιολόγηση τους
- Να επιτρέπει τις αναζητήσεις σε συλλογές εγγράφων
- Να χρησιμοποιεί και να μοιράζει όσο το δυνατόν περισσότερα με κατάλληλες W3C εισηγήσεις όπως XML 1.0, Namespaces, XML Schema και XPath.

Οι XQuery εισηγήσεις περικλείουν 11 ξεχωριστά έγγραφα και πάνω από 1000 εκτυπωμένες σελίδες. Σχεδιάστηκαν ώστε να χρησιμοποιηθούν από τους δημιουργούς του XQuery software και ποικίλουν σε αναγνωσιμότητα και ευκολία προσέγγισης.

2.5 Το Περιβάλλον της XQuery

Η XQuery είναι βασισμένη πάνω σε άλλες τεχνολογίες κυρίως σε XPath, XSLT, SQL και XML Schema. Το κεφάλαιο αυτό επεξηγεί πως η XQuery συμπίπτει με αυτές τις τεχνολογίες.

2.6 XQuery και XPath

Η XPath ξεκίνησε ως μια γλώσσα για τη συλλογή στοιχείων και χαρακτηριστικών από ένα XML έγγραφο καθώς διείσδυε στην ιεραρχία του και φιλτράροντας ανεπιθύμητα περιεχόμενα. Η XPath 1.0 είναι μια σαφώς απλή αλλά χρήσιμη εισήγηση, η οποία καθορίζει τα path expressions, και πρόκειται για ένα περιορισμένο σύνολο λειτουργιών. Η XPath 2.0 έγινε κάτι περισσότερο από αυτό, περικλείοντας μια ευρεία ποικιλία από εκφράσεις και λειτουργίες, όχι μόνο path expressions.

Η XQuery 1.0 και η XPath 2.0 συμπίπτουν σε μεγάλο βαθμό. Έχουν τα ίδια μοντέλα δεδομένων και το ίδιο σύνολο built-in λειτουργιών και τελεστών. Η XPath 2.0 αποτελεί στην ουσία ένα υποσύνολο της XQuery 1.0. Η XQuery περιέχει ορισμένα γνωρίσματα που δεν υπάρχουν στην XPath, όπως FLWORs

και XML δομητές. Αυτό συμβαίνει επειδή τα γνωρίσματα αυτά δεν είναι σχετικά με το selecting, αντιθέτως έχουν να κάνουν με τη δόμηση ή τα sorting query αποτελέσματα.

Η XPath 2.0 δημιουργήθηκε με την πρόθεση να είναι όσο το δυνατόν συμβατή με την XPath 1.0. Σχεδόν όλες οι εκφράσεις της XPath 1.0 ισχύουν στην XPath 2.0, με κάποιες αμελητέες διαφορές στον τρόπο που γίνονται οι εκτιμήσεις.

2.7 XQuery σε σχέση XSLT

Η XSLT είναι μια W3C γλώσσα για τη μετατροπή XML εγγράφων σε άλλα XML έγγραφα ή ακόμα και οποιουδήποτε είδους εγγράφων. Υπάρχουν πολλά κοινά χαρακτηριστικά στις δυνατότητες της XQuery και της XSLT. Στην πραγματικότητα, τα πρότυπα της XSLT 2.0 είναι βασισμένα πάνω στην XPath 2.0, οπότε έχει το ίδιο μοντέλο δεδομένων και υποστηρίζει τις ίδιες built-in λειτουργίες και τους ίδιους τελεστές όπως η XQuery καθώς και πολλές ίδιες εκφράσεις.

Διαφορές XQuery και XSLT:

- Οι XSLT εφαρμογές δημιουργήθηκαν για να μετατρέπουν ολόκληρα έγγραφα· φορτώνουν ολόκληρα έγγραφα στη μνήμη. Αντιθέτως της XQuery δημιουργήθηκαν για την επιλογή δεδομένων από μια βάση. Η XQuery σχεδιάστηκε ώστε να είναι κλιμακωτή και να μπορεί να εκμεταλλευτεί τα χαρακτηριστικά των βάσεων δεδομένων.
- Η XQuery έχει μια πιο σύνθετη non-XML σύνταξη η οποία σε πολλές περιπτώσεις είναι πιο εύκολη ώστε να γράψεις και να διαβάσεις (και να ενσωματωθεί σε κώδικα) από ότι η XML σύνταξη της XSLT.
- Η XQuery σχεδιάστηκε για να επιλέγει από μια συλλογή εγγράφων. Οι FLOWER εκφράσεις καθιστούν εύκολη τη σύνδεση πληροφοριών μεταξύ εγγράφων. Η XSLT 2.0 μπορεί να χειριστεί πολλαπλά έγγραφα αλλά οι επεξεργαστές της δεν βελτιστοποιήθηκαν γι’ αυτήν την περίπτωση χρήσης.

Γενικά, όταν μετατρέπουμε ένα ολόκληρο XML έγγραφο από ένα XML λεξιλόγιο (τρόπος σύνταξης) σε ένα άλλο, είναι καλύτερα να χρησιμοποιήσουμε την XSLT. Όταν όμως αποσκοπούμε στην επιλογή υποσυνόλου πληροφοριών από ένα XML έγγραφο ή μια βάση δεδομένων, τότε πρέπει να χρησιμοποιήσουμε την XQuery.

2.8 XQuery σε σύγκριση με την SQL

Η XQuery δανείστηκε ιδέες από την SQL, ακόμα και κάποιοι από τους σχεδιαστές της XQuery ήταν σχεδιαστές της SQL. Η διαφορά μεταξύ XQuery και SQL είναι απλή, η XQuery είναι για xml ενώ η SQL για αλληλοσχετιζόμενα δεδομένα. Ωστόσο, όλο και περισσότερο η διαφορά αυτήν άρχισε να γίνεται αδιευκρίνιστη καθώς στα αλληλοσχετιζόμενα δεδομένα χρησιμοποιούνται xml frontends (το οποίο σημαίνει να μετατρέπουμε το περιεχόμενο σε xml και μετά να συνθέτουμε, χρησιμοποιώντας την xml σαν βάση δεδομένων κατά την

διάρκεια της σύνθεσης) επιτρέποντας την XML να αποθηκευτεί σε αλληλοσχετιζόμενη βάση δεδομένων.

Η XQuery είναι απίθανο να αντικαταστήσει την SQL για τα αυστηρά δομημένα δεδομένα που είναι αποθηκευμένα σε αλληλοσχετιζόμενες βάσεις δεδομένων. Το πιο πιθανό όμως είναι αυτές οι δυο να συνυπάρχουν, με την XQuery να χρησιμοποιείται σε λιγότερο δομημένα δεδομένα ή σε δεδομένα που προορίζονται για XML εφαρμογές και την SQL να εξακολουθεί να χρησιμοποιείται για αυστηρά δομημένα δεδομένα.

2.9 XQuery και XML Schema

Η XML Schema είναι ένα W3C πρότυπο για προσδιοριστικές παραστάσεις, που χρησιμοποιείται για την επιβεβαίωση XML εγγράφων και να προσδιορίζει τύπους σε XML attributes. Η XQuery χρησιμοποιεί το type system της XML Schema, το οποίο περιέχει built-in types που αναπαριστούν κοινά datatypes όπως string, δεκαδικούς και ημερομηνίες. Η XML Schema προσδιορίζει μια γλώσσα για τον ορισμό δικών μας types βασισμένα σε built-in types.

Εάν ένα έγγραφο εισόδου σε ένα ερώτημα περιέχει μια παράσταση, τα types μπορούν να χρησιμοποιηθούν όταν αξιολογούνται οι εκφράσεις του εγγράφου. Αυτό έχει ως πλεονέκτημα να επιτρέπει τους επεξεργαστές να βελτιώνουν το ερώτημα και να εντοπίζουν τα λάθη νωρίτερα.

Οι χρήστες της XQuery δεν είναι υποχρεωμένοι να χρησιμοποιούν παραστάσεις. Είναι επιτρεπτό να γράψουν ένα ολόκληρο ερώτημα χωρίς καμιά αναφορά σε παραστάσεις. Ωστόσο, παρέχεται ένα πλούσιο σύνολο λειτουργιών και τελεστών, οπότε είναι χρήσιμο να κατανοήσουμε το type system και τη χρήση των built-in types ακόμα και αν δεν υπάρχουν παραστάσεις.

2.10 Το Μοντέλο Δεδομένων της XQuery

Η XQuery έχει ένα μοντέλο δεδομένων που χρησιμοποιείται για να ορίσει όλες τις τιμές μέσα στα ερωτήματα, συμπεριλαμβάνοντας και τις τιμές μέσα στα έγγραφα εισόδου, στα αποτελέσματα και οποιεσδήποτε άλλες ενδιάμεσες τιμές. Το μοντέλο δεδομένων της XQuery είναι γνωστό ως XQuery 1.0 και XPath 2.0 Data Model ή XDM. Δεν είναι ακριβώς ίδιο με το Infoset (το W3C μοντέλο για XML έγγραφα) γιατί υποστηρίζει τιμές που δεν είναι ολοκληρωμένα XML έγγραφα, όπως ακολουθίες στοιχείων και ατομικά στοιχεία.

Η κατανόηση του μοντέλου δεδομένων της XQuery είναι ανάλογη με την κατανόηση πινάκων, στηλών και γραμμών όταν μαθαίνουμε SQL. Περιγράφει τη δομή και των εισόδων και των εξόδων ενός ερωτήματος. Δεν χρειάζεται να είσαι ειδικός στην πλοκή του μοντέλου δεδομένων για να γράψεις XML ερωτήματα αλλά είναι απαραίτητο να κατανοήσεις τα βασικά συστατικά:

- *Node:* Μια XML δομή όπως ένα στοιχείο ή χαρακτηριστικό

- *Atomic Value:* Μια απλή τιμή δεδομένου χωρίς να συνοδεύεται από κάποιο markup
- *Item:* Ένας γενικός όρος που αναφέρεται είτε στο node είτε στην atomic value
- *Sequence:* Μια λίστα από κανένα, ένα ή και περισσότερα items

2.11 Κόμβοι

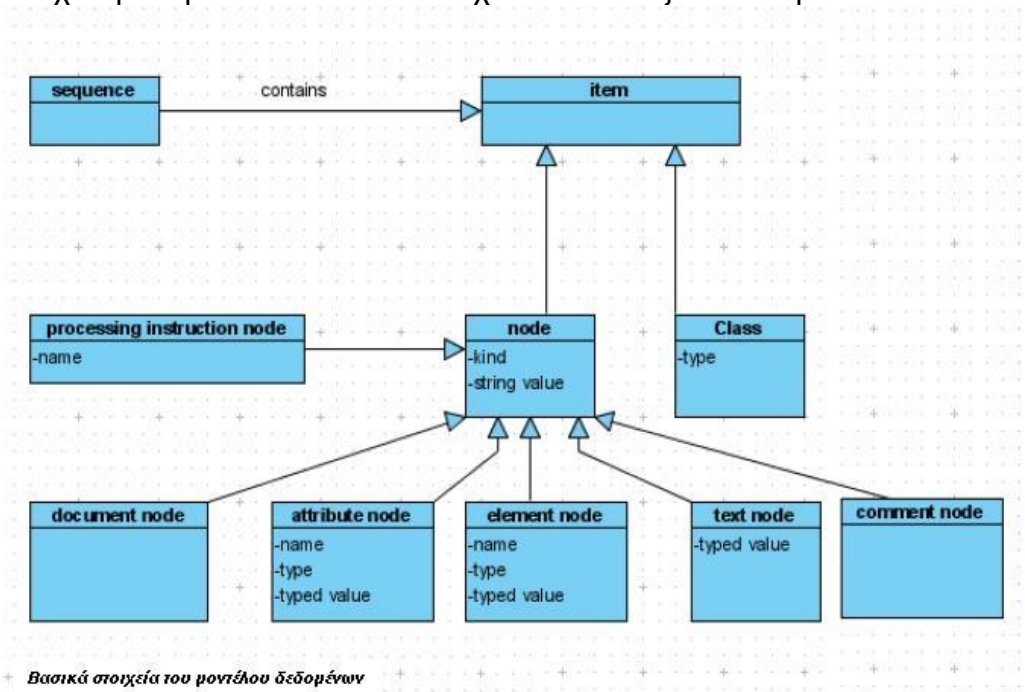
Οι κόμβοι χρησιμοποιούνται για να αναπαραστήσουν τη δομή της XML ως στοιχεία και χαρακτηριστικά. Αναφέρονται με πολλές εκφράσεις, περικλείοντας δομητές και path expressions.

❖ ***Κόμβοι kinds***

Η XQuery χρησιμοποιεί 6 ειδών κόμβους:

- Element nodes
  Απεικονίζει ένα στοιχείο XML
- Attribute nodes
  Απεικονίζει ένα XML χαρακτηριστικό
- Document nodes
  Απεικονίζει ένα XML έγγραφο
- Text nodes
  Απεικονίζει χαρακτήρες δεδομένων
- Processing instruction nodes
  Απεικονίζει XML οδηγίες επεξεργασίας
- Comment nodes
  Απεικονίζει ένα XML σχόλιο

Η σχέση ανάμεσα σ' αυτά τα στοιχεία απεικονίζονται στην εικόνα 2.1



Βασικά στοιχεία του μοντέλου δεδομένων

Εικόνα 2.1

2.12 Η Δομή της XQuery

Path Expressions

Οι path expressions χρησιμοποιούνται για να εξερευνήσουμε έγγραφα εισόδου ώστε να επιλέγουν στοιχεία και χαρακτηριστικά. Το κεφάλαιο αυτό εξηγεί πως χρησιμοποιούνται οι path expressions για να επιλέξουν στοιχεία και χαρακτηριστικά από ένα έγγραφο εισόδου και να ορίσει κατηγορήματα ώστε να φιλτράρει αυτά τα αποτελέσματα. Καλύπτει επίσης τις διαφορετικές μεθόδους εκχώρησης των εγγράφων.

Μια path expression αποτελείται από ένα ή περισσότερα βήματα που χωρίζονται με slash (/) ή double slash (//). Για παράδειγμα, το path:

Doc(“catalog.xml”)/catalog/product

επιλέγει όλα τα προϊόντα του στοιχείου catalog από το έγγραφο catalog.xml.

Ο πίνακας 2.1 δείχνει κάποια άλλα απλά path expressions

Παραδείγματα Εκφράσεων

Παράδειγμα	*Επεξήγηση*
doc("catalog.xml")/catalog	Το στοιχείο catalog που είναι το απομακρυσμένο στοιχείο του εγγράφου
doc("catalog.xml")//product	Όλα τα στοιχεία product μέσα στο έγγραφο
doc("catalog.xml")//product/@dept	Όλα τα χαρακτηριστικά dept του στοιχείου product στο έγγραφο
doc("catalog.xml")/catalog/*	Όλα τα παραγόμενα στοιχεία του στοιχείου catalog
doc("catalog.xml")/catalog/*/number	Όλα τα στοιχεία number που παράγονται απο το στοιχείο catalog

Πίνακας 2.1 Παραδείγματα Εκφράσεων

Οι path expressions επιστρέφουν κόμβους σε τάξη εγγράφου. Αυτό σημαίνει ότι τα παραδείγματα του πίνακα επιστρέφουν τα στοιχεία product με την ίδια σειρά που αυτά εμφανίζονται στο έγγραφο catalog.xml.

2.12.1Path Expressions και Πλαίσια Κειμένου

Μια path expression πάντα υπολογίζεται αναφορικά σε ένα ειδικό αντικείμενο πλαισίου, το οποίο ωφελεί ως σημείο εκκίνησης για το μονοπάτι. Κάποια path expressions ξεκινάνε με ένα βήμα το οποίο θέτει το αντικείμενο πλαισίου ως:

doc(“catalog.xml”)/catalog/product/number

Η λειτουργία κλήσης doc(“catalog.xml”) επιστρέφει το κόμβο εγγράφου του catalog.xml το οποίο γίνεται το αντικείμενο πλαισίου. Όταν το αντικείμενο αυτό είναι κόμβος λέγεται *context node.* Το υπόλοιπο μονοπάτι υπολογίζεται αναλογικά σε αυτό. Άλλο ένα παράδειγμα:

$catalog/product/number

όπου η τιμή της μεταβλητής $catalog θέτει το περιεχόμενο. Η μεταβλητή πρέπει να επιλέξει μηδέν, ένα ή περισσότερους κόμβους οι οποίοι θα γίνουν οι κόμβοι πλαισίου για την υπόλοιπη έκφραση.

Μια path expression μπορεί επίσης να είναι συγγενική. Για παράδειγμα, μπορεί επίσης να ξεκινάει με ένα όνομα, όπως:

product/number

Αυτό σημαίνει ότι η έκφραση μονοπατιού θα υπολογιστεί αναλογικά με το τρέχων κόμβο πλαισίου, ο οποίος θα πρέπει να έχει ορισθεί έξω από την έκφραση. Μπορεί να έχει ορισθεί από τον επεξεργαστή εκτός του πεδίου δράσης του ερωτήματος ή σε μια εξωτερική έκφραση.

2.12.2 Βήματα και Αλλαγή Πλαισίου Κειμένου

Το αντικείμενο πλαισίου αλλάζει με κάθε βήμα. Ένα βήμα επιστρέφει μια ακολουθία από μηδέν, ένα ή περισσότερους κόμβους που χρησιμεύουν ως αντικείμενο περιβάλλοντος για την αξιολόγηση του επόμενου βήματος. Για παράδειγμα:

doc(“catalog.xml”)/catalog/product/number

το βήμα doc(“catalog.xml”) επιστρέφει ένα κόμβο εγγράφου ο οποίος μας συνδέει με το xml έγγραφο που θέλουμε να χρησιμοποιήσουμε και λειτουργεί ως αντικείμενο πλαισίου όταν αξιολογείται το βήμα catalog. Το βήμα catalog αξιολογείται χρησιμοποιώντας τον προηγούμενο κόμβο εγγράφου ως τρέχων κόμβο περιβάλλοντος, επιστρέφοντας μια ακολουθία ενός στοιχείου catalog του κόμβου εγγράφου δηλαδή στο παράδειγμά μας του “catalog.xml”. Αυτό το στοιχείο catalog εξυπηρετεί μετά σαν κόμβος περιβάλλοντος για την αξιολόγηση του βήματος product, το οποίο επιστρέφει την ακολουθία product.

Το τελικό βήμα, number, αξιολογείται διαδοχικά για κάθε product στην ακολουθία. Κατά τη διάρκεια αυτής της διαδικασίας, ο επεξεργαστής κρατάει στοιχεία από 3 πράγματα:

- Τον ίδιο τον κόμβο περιβάλλοντος – για παράδειγμα, το στοιχείο product που είναι στην τρέχουσα επεξεργασία
- Την ακολουθία περιβάλλοντος, η οποία είναι η ακολουθία των αντικειμένων που επεξεργάζονται εκείνη τη στιγμή – για παράδειγμα όλα τα στοιχεία product
- Τη θέση του κόμβου μέσα στην ακολουθία, η οποία χρησιμοποιείται για να ξαναβρίσκει τους κόμβους βασιζόμενος στη θέση τους

2.12.3 Είδη Βημάτων

Όπως είδαμε στα προηγούμενα παραδείγματα, τα βήματα σε ένα μονοπάτι μπορούν να είναι απλά εκφράσεις όπως κλήσεις λειτουργιών (doc(“catalog.xml”)) ή αναφορά σε μεταβλητές ($catalog). Οποιαδήποτε έκφραση που επιστρέφει κόμβους μπορεί να είναι στην αριστερή πλευρά του τελεστή slash.

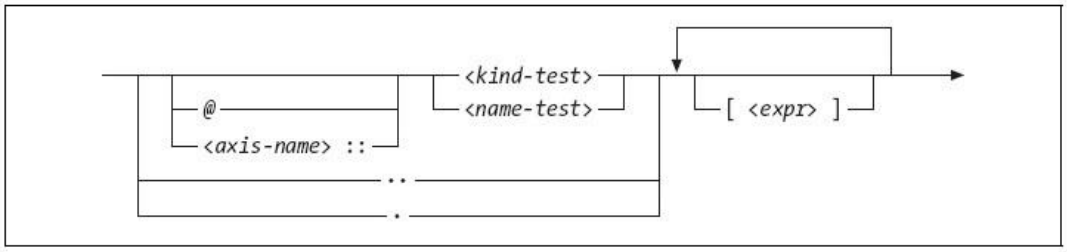
Άλλο ένα είδος βήματος είναι το *axis step* που μας επιτρέπει να μεταφερόμαστε ανάμεσα στην XML ιεραρχία κόμβων. Υπάρχουν 2 είδη:

- *Forward step*

  Το βήμα αυτό επιλέγει απόγονους ή κόμβους που εμφανίζονται μετά τον κόμβο περιβάλλοντος
- *Reverse step*

  Το βήμα αυτό επιλέγει προγόνους ή κόμβους που εμφανίζονται πριν τον κόμβο περιβάλλοντος

Στα μέχρι τώρα παραδείγματα, τα catalog, product και @dept είναι όλα axis step. Η σύνταξη ενός axis step φαίνεται στην εικόνα 2.2



Εικόνα 2.2 Σύνταξη ενός βήματος σε ένα path expression

2.12.4 Άξονες

Κάθε εμπρόσθιο ή αντίστροφο βήμα έχει έναν άξονα, ο οποίος καθορίζει τη κατεύθυνση και τη σχέση των επιλεγμένων κόμβων. Για παράδειγμα, το *child :: axis* (εμπρόσθιος άξονας) μπορεί να χρησιμοποιηθεί για να υποδείξει ότι μόνο ένας κόμβος παιδί πρέπει να επιλεχθεί, ενώ το *parent :: axis* (αντίστροφος άξονας) χρησιμοποιείται για να υποδείξει ότι μόνο ένας κόμβος γονέας πρέπει να επιλεχθεί.

Στον πίνακα 2.2 φαίνεται η λίστα με τους 12 άξονες

Axis	Έννοια
self::	Ο ίδιος ο κόμβος του πλαισίου κειμένου.
child::	Ο απόγονος του κόμβου του πλαισίου κειμένου. Τα χαρακτηριστικά δεν θεωρούνται απόγονοι ενός στοιχείου. Αυτός είναι και ο default άξονας εάν δεν έχει οριστεί κανένας.
descendant::	Όλοι οι απόγονοι του κόμβου του πλαισίου κειμένου (απόγονος, απόγονος του απογόνου, κ.τ.λ.). Τα χαρακτηριστικά δεν θεωρούνται απόγονοι.
descendant-or-self::	Ο ίδιος ο κόμβος του πλαισίου κειμένου και οι απόγονοί του.

attribute::	Τα χαρακτηριστικά του κόμβου του πλαισίου κειμένου (εάν υπάρχουν).
following::	Όλοι οι κόμβοι που ακολουθούν τον κόμβου του πλαισίου κειμένου στο έγγραφο, εκτός από τους απογόνους του πλαισίου κειμένου.
following-sibling::	Όλοι οι “αδερφικοί” κόμβοι που ακολουθούν τον κόμβο του πλαισίου κειμένου. Τα χαρακτηριστικά των ίδιων στοιχείων δεν λαμβάνονται υπ’ όψιν σαν “αδερφικά”.
parent::	Ο γονέας του κόμβου του πλαισίου κειμένου (εάν υπάρχουν). Αυτός είναι είτε το στοιχείο είτε ο κόμβος του εγγράφου ο οποίος τον περιέχει. Ο πρόγονος ενός χαρακτηριστικού είναι το ίδιο το στοιχείο του, παρόλο που δεν θεωρείται απόγονος αυτού του στοιχείου.
ancestor::	Όλοι οι πρόγονοι του κόμβου του πλαισίου κειμένου (γονέας, γονέας του γονέα, κ.τ.λ.).
ancestor-or-self::	Ο κόμβος του πλαισίου κειμένου και όλοι οι πρόγονοί του
preceding::	Όλοι οι κόμβοι που προηγούνται του κόμβου του πλαισίου κειμένου του εγγράφου, εκτός από τους απογόνους του κόμβου του πλαισίου κειμένου.
preceding-sibling::	Όλοι οι “αδερφικοί” κόμβοι του κόμβου του πλαισίου κειμένου που προηγούνται. Τα χαρακτηριστικά των ίδιων στοιχείων δεν θεωρούνται “αδερφικοί” κόμβοι.

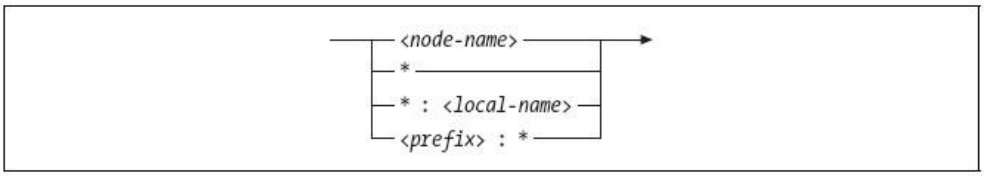
Πίνακας 2.2 *Οι 12 άξονες*

2.12.5 Κόμβοι Tests

Κάθε βήμα άξονα έχει ένα *node test*. Το τεστ κόμβου υποδεικνύει ποιος κόμβος (από το όνομα ή το είδος) θα επιλεχθεί από τους καθορισμένους άξονες. Για παράδειγμα, *child:product* επιλέγει μόνο στοιχεία product απόγονοι του κόμβου περιβάλλοντος. Δεν επιλέγει άλλου είδους απόγονους ή άλλα στοιχεία product που δεν είναι απόγονοι του κόμβου.

- **Node Name Tests**

Σε προηγούμενα παραδείγματα, τα περισσότερα από τα τεστ κόμβων βασίζονταν σε ονόματα, όπως product και dept. Αυτά είναι γνωστά ως name tests. Η σύνταξη ενός name test κόμβου φαίνεται στην εικόνα 2.3



Εικόνα 2.3 Σύνταξη ενός κόμβου name test

- **Node Name Tests and Namespace**

Τα ονόματα που χρησιμοποιούνται σε τεστ κόμβων ονομάζονται *qualified names*, που σημαίνει ότι έχουν επηρεαστεί από δηλώσεις namespace. Μια δήλωση namespace είναι σε περιθώριο αν εμφανίζεται σε εξωτερικό στοιχείο ή στον πρόλογο ενός ερωτήματος. Τα ονόματα μπορούν είτε να έχουν πρόθεμα είτε όχι. Αν ένα όνομα έχει πρόθεμα, το πρόθεμα πρέπει να χαρτογραφείται σε μια namespace δήλωση.

Αν ένα στοιχείο ονόματος δεν έχει πρόθεμα, και υπάρχει ένα εξ ορισμού δηλωμένο namespace, θεωρείται πως θα βρίσκεται σ' αυτό το namespace, αλλιώς δε βρίσκεται σε κανένα namespace. Τα ονόματα χαρακτηριστικών από την άλλη, δεν επηρεάζονται από εξ ορισμού δηλώσεις namespace.

Η χρήση namespace προθεμάτων σε ένα path expression φαίνεται στο επόμενο παράδειγμα, όπου το πρόθεμα prod πρώτα χαρτογραφείται στο namespace και μετά χρησιμοποιείται στα βήματα prod: product και prod: number. Το πρόθεμα χρησιμεύει ως proxy για το όνομα του namespace. Δεν είναι απαραίτητο τα προθέματα στο path expression να ταιριάζουν με τα προθέματα στο έγγραφο εισόδου, το μόνο που είναι απαραίτητο είναι τα προθέματα να χαρτογραφούνται στο ίδιο namespace. Στο παράδειγμα που ακολουθεί μπορούμε να χρησιμοποιήσουμε το πρόθεμα pr αντί για prod στο ερώτημα, αρκεί να χρησιμοποιούμε το ίδιο σε όλο το ερώτημα.

Example. Prefixed name tests

Input document (prod_ns.xml)

```
<prod:product xmlns:prod="http://datypic.com/prod">
<prod:number>563</prod:number>
<prod:name language="en">Floppy Sun Hat</prod:name>
</prod:product>
```

Query

```
declare namespace prod = "http://datypic.com/prod";
<prod:prodList>{
doc("prod_ns.xml")/prod:product/prod:number
}</prod:prodList>
```

Results

```
<prod:prodList xmlns:prod="http://datypic.com/prod">
<prod:number>563</prod:number>
</prod:prodList>
```

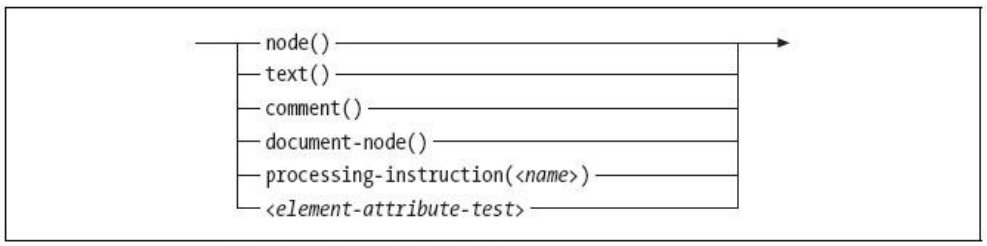
❖ **Node Name Tests και Wildcards**

Μπορούμε να χρησιμοποιήσουμε wildcards για να ταιριάξουμε ονόματα. Το βήμα *child::** (εν συντομία μόνο *) μπορεί να χρησιμοποιηθεί για να επιλέξουμε όλους τους απόγονους στοιχείων, ασχέτως του ονόματος. Παρομοίως το @* (ή attribute::*) μπορεί να χρησιμοποιηθεί για να επιλέξουμε όλα τα χαρακτηριστικά, ασχέτως του ονόματος.

Επί προσθέτως, τα wildcards μπορούν να χρησιμοποιηθούν μόνο για το namespace and/ ή για τοπικό μέρος του ονόματος. Το βήμα prod:* επιλέγει όλα τα απόγονα στοιχεία στο namespace που χαρτογραφείται στο πρόθεμα prod και το βήμα *: product επιλέγει όλα τα απόγονα στοιχεία product που βρίσκονται σε οποιοδήποτε ή σε κανένα namespace.

❖ **Node Kind Tests**

Εκτός από τα τεστ βασισμένα στο όνομα κόμβου, υπάρχουν και τα τεστ βασισμένα στο είδος του κόμβου. Η σύνταξη ενός τέτοιου τεστ φαίνεται στην εικόνα 2.4



Εικόνα 2.4 Σύνταξη ενός kind test κόμβου

Το τεστ κόμβου ( ) θα ανακτήσει όλα τα διαφορετικά είδη κόμβων. Μπορούμε να ορίσουμε τον κόμβο ( ) ως ολόκληρο βήμα και θα είναι εξ ορισμού *child::axis*. Σ' αυτή την περίπτωση, θα επαναφέρει όλους τους απόγονους κόμβους στοιχείων, κειμένων, σχολίων, και οδηγίες επεξεργασίας (αλλά όχι χαρακτηριστικών καθώς δε θεωρούνται απόγονοι). Αυτό είναι σε αντίθεση με το *, το οποίο επιλέγει κόμβους απόγονων στοιχείων μόνο.

Μπορούμε επίσης να χρησιμοποιήσουμε το κόμβο ( ) σε συνδυασμό με τους άξονες. Για παράδειγμα, *ancestor::node()* θα επιστρέψει όλους τους κόμβους πρόγονων στοιχείων και τον κόμβο εγγράφου (εάν υπάρχει). Είναι διαφορετικό από το *ancestor::** το οποίο επιστρέφει μόνο κόμβους πρόγονων στοιχείων. Μπορούμε να χρησιμοποιήσουμε ακόμα και *attribute::node()* που επιστρέφει κόμβους χαρακτηριστικών, αλλά δε χρησιμοποιείται συχνά επειδή σημαίνει το ίδιο με το @*.

Εάν χρησιμοποιούμε διαγραμματικές παραστάσεις, μπορούμε επίσης να ελέγξουμε στοιχεία και χαρακτηριστικά βασιζόμενοι στον τύπο τους χρησιμοποιώντας είδη τεστ κόμβων. Για παράδειγμα, μπορούμε να ορίσουμε στοιχείο (*, ProductType) για να επιστρέψει όλα τα στοιχεία με τύπο

ProductType, ή στοιχείο (product, ProductType) που θα επιστρέψει όλα τα στοιχεία με όνομα product τύπου ProductType.

2.12.6 Predicates

Τα κατηγορήματα χρησιμοποιούνται σε μια path expression για να φιλτράρουν τα αποτελέσματα μόνο κόμβους που συναντούν συγκεκριμένα κριτήρια. Χρησιμοποιώντας ένα κατηγόρημα, μπορούμε για παράδειγμα να επιλέξουμε μόνο τα στοιχεία που έχουν μια συγκεκριμένη τιμή για ένα χαρακτηριστικό ή απόγονο στοιχείο, χρησιμοποιώντας ένα κατηγόρημα σαν *[@dept = "ACC"]*. Μπορούμε επίσης να επιλέξουμε στοιχεία που έχουν ειδικά στοιχεία χαρακτηριστικών, χρησιμοποιώντας ένα κατηγόρημα όπως *[color]* ή στοιχεία που συμβαίνουν σε συγκεκριμένες θέσεις μέσα στους γονείς του, χρησιμοποιώντας κατηγόρημα όπως *[3]*.

Η σύνταξη ενός κατηγορήματος είναι απλά μια έκφραση μέσα σε τετραγωνικές αγκύλες (**[]**). Ο πίνακας 2.3 που ακολουθεί δείχνει κάποια παραδείγματα κατηγορημάτων.

Παράδειγμα	**Έννοια**
product[name = "Floppy Sun Hat"]	Όλα τα προϊόντα που έχουν την τιμή του “name” ίσο με “Floppy Sun Hat”
product[number < 500]	Όλα τα προϊόντα που έχουν το “number” μικρότερο από 500
product[@dept = "ACC"]	Όλα τα προϊόντα που έχουν το χαρακτηριστικό “dept” ίσο με “ACC”
product[desc]	Όλα τα προϊόντα που έχουν τουλάχιστον ένα “desc” απόγονο
product[@dept	Όλα τα προϊόντα που έχουν ένα “desc” χαρακτηριστικό
product[@dept]/number	Όλα τα “number” των απογόνων των προϊόντων που έχουν ένα “dept”χαρακτηριστικό

Πίνακας 2.3 *Παραδείγματα κατηγορημάτων*

Αν μια έκφραση αποτιμάται σε οτιδήποτε άλλο εκτός αριθμού, ορίζεται η λειτουργική του Boolean τιμή. Αυτό σημαίνει πως εάν αποτιμάται στο *xs:boolean* η τιμή *false*, ο αριθμός 0 ή NaN, ένα string μηδενικού μήκους, ή η κενή ακολουθία θεωρείται ψευδής. Στις περισσότερες άλλες περιπτώσεις, θεωρείται αληθής. Εάν τη λειτουργική Boolean τιμή είναι αληθής για έναν κόμβο, ο κόμβος επιστρέφεται. Εάν είναι ψευδής, δεν επιστρέφεται.

Αν μια έκφραση αποτιμάται σε αριθμό, προσδιορίζεται η αριθμητική θέση του αντικειμένου μέσα στην ακολουθία που μόλις επεξεργάστηκε. Αυτά κάποιες φορές ονομάζονται *positional predicates.* Οποιαδήποτε έκφραση κατηγορήματος που αποτιμάται σε έναν ακέραιο μπορεί να θεωρηθεί κατηγόρημα θέσεως. Αν ορίσουμε έναν αριθμό μεγαλύτερο από τον αριθμό των

αντικειμένων σε μια ακολουθία πλαισίου, δε θα εμφανίσει σφάλμα, απλά δε θα επιστρέψει κανένα κόμβο.

Όπως βλέπουμε στο τελευταίο παράδειγμα, το κατηγόρημα δεν απαιτείται να εμφανίζεται στο τέλος του path expression, μπορεί να εμφανιστεί στο τέλος οποιουδήποτε βήματος.

Να σημειώσουμε ότι το *product[number]* είναι διαφορετικό του *product/number*. Ενώ και οι 2 εκφράσεις φιλτράρουν προϊόντα που δεν έχουν απόγονο number, στην προηγούμενη έκφραση, το στοιχείο product επιστρέφεται. Στην επόμενη περίπτωση, το στοιχείο number επιστρέφεται.

2.12.7 FLWOR

Η πιο συνηθισμένη δομή στην XQuery είναι η δομή FLWOR (διαβάζεται flower). Η δομή αυτήν είναι βασισμένη πάνω στη δομή της SQL, χρησιμοποιείται όμως για πιο πολύπλοκα ερωτήματα. Αντίθετα με την SQL, η FLWOR μας επιτρέπει να διαχειριστούμε, να μετασχηματίσουμε και να ταξινομήσουμε τα αποτελέσματα. Για να επιτρέψει πιο ευανάγνωστες και δομημένες επιλογές, η FLWOR επιτρέπει λειτουργικότητες όπως ένωση δεδομένων από πολλαπλές πηγές, δόμηση νέων στοιχείων και χαρακτηριστικών, εκτίμηση λειτουργιών σε ενδιάμεσες τιμές και ταξινόμηση αποτελεσμάτων.

Το όνομα FLWOR συμβολίζει τις λέξεις κλειδιά *for, let, where, order by* και *return* που χρησιμοποιούνται στην έκφραση όπως φαίνεται στο παράδειγμα

Παράδειγμα FLWOR

```
for $prod in doc("catalog.xml")//product
let $prodDept := $prod/@dept
where $prodDept = "ACC" or $prodDept = "WMN"
return $prod/name
```

- *for*

Ο όρος αυτός θέτει μια επανάληψη στα στοιχεία product που επιστρέφονται από την έκφραση. Η μεταβλητή $prod καθορίζεται για κάθε product διαδοχικά στην ακολουθία. Το υπόλοιπο της FLWOR αποτιμάται μια φορά για κάθε προϊόν, στη συγκεκριμένη περίπτωση 4 φορές.

- *let*

Ο όρος αυτός δεσμεύει τη μεταβλητή $prodDept στην τιμή του χαρακτηριστικού dept.

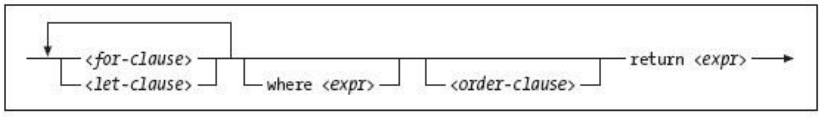
- *where*

Ο όρος αυτός επιλέγει στοιχεία των οποίων το χαρακτηριστικό dept είναι ισοδύναμο με ACC ή WMN.

- *return*

Ο όρος αυτός επιστρέφει το όνομα καθενός από τα 3 στοιχεία που περνούν τον όρο where.

Η γενική σύνταξη της FLWOR φαίνεται στην εικόνα 2.5



Εικόνα 2.5 Σύνταξη ενός FLOWR

Μπορούν να υπάρχουν πολλαπλοί for και let όροι, σε οποιαδήποτε σειρά, ακολουθούμενοι από έναν προαιρετικό όρο where, ακολουθούμενος από έναν προαιρετικό όρο order by, ακολουθούμενος από έναν υποχρεωτικό όρο return. Μια FLWOR πρέπει να έχει τουλάχιστον έναν όρο for ή let.

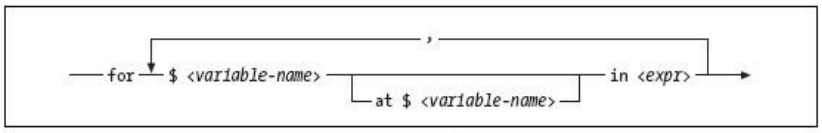
Οι FLWOR μπορούν να αποτελούν όλο το ερώτημα ή να εμφανίζονται απλά σε άλλες εκφράσεις όπως στον όρο return ενός άλλου FLWOR ή ακόμα και στη κλήση μιας λειτουργίας όπως:

```
max(for $prod in doc("catalog.xml")//product
        return xs:integer($prod/number))
```

Οι λέξεις κλειδιά for και return ευθυγραμμίζονται κάθετα για να κάνουν τη δομή πιο εμφανής, που παρόλο είναι μια καλή πρακτική δεν είναι πάντοτε εφικτή.

❖ Ο Όρος *For*

Ο όρος for, του οποίου η σύνταξη φαίνεται στην εικόνα που ακολουθεί, θέτει μια επανάληψη που επιτρέπει στο υπόλοιπο FLWOR να εκτιμηθεί πολλές φορές, μια φορά για κάθε αντικείμενο στην ακολουθία που επιστρέφεται από την έκφραση μετά τη δεσμευμένη λέξη *in*. Αυτήν η ακολουθία, γνωστή ως *binding sequence*, μπορεί να υπολογίσει οποιαδήποτε ακολουθία από μηδέν, ένα ή περισσότερα αντικείμενα. Στο προηγούμενο παράδειγμα, ήταν μια ακολουθία από στοιχεία product, μπορούσε όμως να είναι ατομικές τιμές, κόμβοι οποιουδήποτε είδους ή ένας συνδυασμός αντικειμένων. Αν η binding sequence είναι μια κενή ακολουθία, το υπόλοιπο FLWOR απλά δεν υπολογίζεται (επαναλαμβάνεται 0 φορές).



Εικόνα 2.6 Σύνταξη της πρότασης for

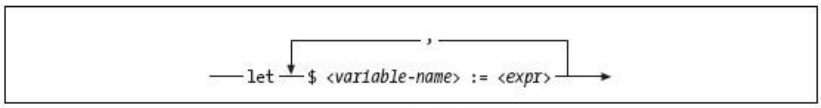
Η FLWOR έκφραση με τους for όρους της είναι παρόμοια με τους βρόχους των διαδικαστικών γλωσσών όπως η C. Ωστόσο, μια σημαντική διαφορά είναι πως στην XQuery, επειδή είναι μια συναρτησιακή γλώσσα, οι επαναλήψεις μπορούν να είναι σε οποιαδήποτε σειρά. Δε χρειάζεται να είναι ακολουθιακά, η μια μετά

την άλλη. Ως επακόλουθο αυτού, δε χρειάζεται να κρατάμε μετρητές μεταβλητών που αυξάνουν σε κάθε επανάληψη.

❖ Ο Όρος *Let*

Ο όρος let είναι ένας εύκολος τρόπος να δεσμεύουμε μια μεταβλητή σε μια τιμή. Αντίθετα με τον όρο for, ο όρος let δεν καταλήγει σε επανάληψη, δεσμεύει ολόκληρη την ακολουθία στη μεταβλητή παρά κάθε αντικείμενο διαδοχικά. Ο όρος αυτός εξυπηρετεί ως προγραμματική ευκολία που αποφεύγει να επαναλαμβάνει την ίδια έκφραση πολλές φορές. Χρησιμοποιώντας κάποιες εφαρμογές, μπορεί να βελτιώσει την απόδοση του καθώς η έκφραση υπολογίζεται μόνο μια φορά αντί κάθε φορά που χρειάζεται. Άλλη μια πρακτική χρήση του όρου let είναι ότι μπορεί να εκτελέσει πολλαπλές λειτουργίες ή εφαρμογές σε σειρά.

Η σύνταξη του όρου let φαίνεται στην εικόνα 2.7



Εικόνα 2.7 Σύνταξη της πρότασης let

Ένας ή περισσότεροι όροι let μπορούν να ανακατευτούν με έναν ή περισσότερους όρους for. Κάθε ένας από τους όρους let και for μπορούν να αναφέρονται σε μια μεταβλητή δεσμευμένη σε οποιοδήποτε από τους προηγούμενους όρους. Η μόνη προϋπόθεση είναι ότι πρέπει να εμφανίζονται πριν από οποιονδήποτε where, order by, ή return όρο.

❖ Ο Όρος *Where*

Ο όρος where χρησιμοποιείται για να καθορίσει κριτήρια με τα οποία φιλτράρει τα αποτελέσματα του FLWOR. Ο όρος αυτός μπορεί να αναφέρεται σε μεταβλητές που έχουν δεσμευτεί από τους όρους for ή let. Εκτός του να διερμηνεύει πολύπλοκα φίλτρα, ο όρος where είναι επίσης πολύ χρήσιμος και για ενώσεις.

Μόνο ένας όρος where μπορεί να περιέχεται σε κάθε FLWOR, αλλά μπορεί να συνταχθεί με πολλές εκφράσεις ενωμένες με τις λέξεις κλειδιά *and* και *or* όπως φαίνεται στο παράδειγμα

Παράδειγμα μιας πρότασης where με πολλαπλές εκφράσεις

```
for $prod in doc("catalog.xml")//product
let $prodDept := $prod/@dept
where $prod/number > 100
        and starts-with($prod/name, "F")
        and exists($prod/colorChoices)
```

```
        and ($prodDept = "ACC" or $prodDept = "WMN")
return $prod
```

Όταν χρησιμοποιούμε μονοπάτια μέσα σε έναν όρο where, πρέπει να προσέξουμε να ξεκινούν με μια έκφραση η οποία θέτει το πλαίσιο αλλιώς ο επεξεργαστής δε ξέρει που να ψάξει για να βρει το ζητούμενο.

Η Boolean τιμή της έκφρασης where υπολογίζεται. Αυτό σημαίνει πως αν η έκφραση where αποτιμηθεί σε μια Boolean τιμή false, το μηδενικού μεγέθους string, ο αριθμός 0 ή NaN, ή η κενή ακολουθία θεωρούνται false και η επιστρεφόμενη έκφραση δεν υπολογίζεται. Αντίθετα αν η Boolean τιμή είναι true, τότε η υπολογίζεται η επιστρεφόμενη έκφραση.

❖ Ο Όρος *Return*

Ο όρος return αποτελείται από τη δεσμευμένη λέξη return ακολουθούμενη από την έκφραση που πρέπει να επιστραφεί. Υπολογίζεται μια φορά για κάθε επανάληψη, υποθέτοντας ότι η έκφραση where είναι αληθής. Η τιμή που επιστρέφεται από ολόκληρο το FLWOR είναι μια ακολουθία αντικειμένων που επιστρέφονται από κάθε υπολογισμό του όρου return.

Εάν περισσότερες από μία εκφράσεις πρέπει να περιληφθούν στον όρο return, μπορούν να συνδυαστούν σε μια ακολουθία.

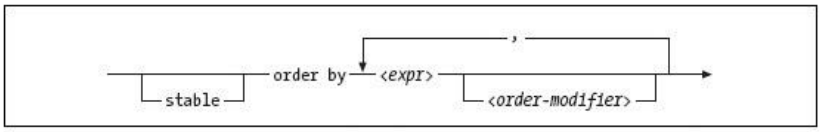
```
for $i in (1 to 3)
return (<one>{$i}</one>, <two>{$i}</two>)
```

Παρενθέσεις και κόμμα χρησιμοποιούνται για να υποδείξουν ότι πρέπει να επιστραφεί μια ακολουθία από 2 στοιχεία. Αν δε χρησιμοποιούνται παρενθέσεις ή κόμμα, οι δομητές 2 στοιχείων δε θεωρούνται μέρος του FLWOR.

❖ Ο Όρος *Order By*

Ο όρος order by χρησιμοποιείται για την ταξινόμηση των αποτελεσμάτων. Αποτελείται από έναν ή περισσότερους διαρρυθμιστικούς προσδιορισμούς, διαχωρισμένοι με κόμμα, οπού ο κάθε ένας αποτελείται από μια έκφραση και έναν προαιρετικό τροποποιητή. Η έκφραση μπορεί να επιστρέψει μόνο μια τιμή για κάθε αντικείμενο που ταξινομείται.

Η σύνταξη του όρου order by φαίνεται στην εικόνα 2.8



Εικόνα 2.8 Σύνταξη της πρότασης order by

2.12.8 Joins

Ένωση είναι ο σύνδεσμος μεταξύ δύο εγγράφων που έχουν ένα κοινό στοιχείο. Αυτή η εφαρμογή είναι πολύ συνηθισμένη σε βάσεις δεδομένων. Η SQL διαθέτει διαφορετικά είδη ενώσεων: inner join, left outer join, right outer join και self join. Παρόμοια και η XQuery μπορεί να ενώσει διαφορετικά έγγραφα με τη μόνη διαφορά ότι στην XQuery δεν έχουμε τις δεσμευμένες λέξεις inner join ή outer join.

- **Inner Joins**

Οι ενώσεις μπορούν να επεκταθούν ώστε να επιτρέψουν περισσότερες από δύο πηγές να ενωθούν. Για παράδειγμα, υποθέτουμε ότι μαζί με τα έγγραφα catalog.xml και order.xml θέλουμε να ενώσουμε και το έγγραφο prices.xml, που περιέχει πληροφορίες για τις τρέχουσες τιμές των προϊόντων.

Το επόμενο παράδειγμα δείχνει την ένωση του εγγράφου prices.xml με τα άλλα έγγραφα ώστε να παρέχει τις πληροφορίες για τις τιμές. Χρησιμοποιεί δύο εκφράσεις στον όρο where για να εφαρμόσει τις δύο ενώσεις.

Παραδείγματα Inner Join

Query

```
for $item in doc("order.xml")//item,
    $product in doc("catalog.xml")//product,
    $price in doc("prices.xml")//prices/priceList/prod
where $item/@num = $product/number and $product/number = $price/@num
return <item num="{$item/@num}"
                name="{$product/name}"
                 price="{$price/price}"/>
```

Results

```
<item num="557" name="Fleece Pullover" price="29.99"/>
<item num="563" name="Floppy Sun Hat" price="69.99"/>
<item num="443" name="Deluxe Travel Bag" price="39.99"/>
<item num="557" name="Fleece Pullover" price="29.99"/>
```

- **Outer Join**

Στο προηγούμενο παράδειγμα, τα αποτελέσματα δεν περικλείουν αντικείμενα στα οποία δεν αντιστοιχούν προϊόντα ή προϊόντα στα οποία δεν αντιστοιχούν αντικείμενα. Υποθέτουμε ότι θέλουμε να κάνουμε μια λίστα με προϊόντα και να την ενώσουμε με το έγγραφο των τιμών. Ακόμα και αν δεν υπάρχει καμία τιμή, μπορούμε να συμπεριλάβουμε το προϊόν στη λίστα. Στις βάσεις δεδομένων αυτήν η ένωση είναι γνωστή ως outer join.

Το παράδειγμα που ακολουθεί δείχνει αυτήν την ένωση. Χρησιμοποιεί δύο FLWOR, ένα ενσωματωμένο στον όρο return του άλλου. Το εξωτερικό FLWOR επιστρέφει τη λίστα των προϊόντων, ανεξάρτητα από τη διαθεσιμότητα των τιμών. Το εσωτερικό FLWOR επιλέγει την τιμή, εάν είναι διαθέσιμη.

Παραδείγματα Outer join

Query

```
for $product in doc("catalog.xml")//product
return <product number="{$product/number}">{
                    attribute price
                          {for $price in doc("prices.xml")//prices/priceList/prod
                           where $product/number = $price/@num
                           return $price/price}
}</product>
```

Results

```
<product number="557" price="29.99"/>
<product number="563" price="69.99"/>
<product number="443" price="39.99"/>
<product number="784" price=""/>
```

2.12.9 Ταξινόμηση

Τα path expressions, τα οποία χρησιμοποιούνται κυρίως για να επιλέξουν στοιχεία και χαρακτηριστικά από έγγραφα εισόδου, πάντα επιστρέφουν αντικείμενα σε μορφή εγγράφου. Τα FLWOR, εξ ορισμού, επιστρέφουν αποτελέσματα βασισμένα στην επιθυμητή ακολουθία που κατηγοριοποιήθηκε στον όρο for, η οποία είναι συχνά σε μορφή εγγράφου αν χρησιμοποιήθηκε path expression. Ο μόνος τρόπος για να ταξινομήσουμε και να παρουσιάσουμε τα δεδομένα σε διαφορετική μορφή από αυτή ενός εγγράφου είναι χρησιμοποιώντας τον όρο order by του FLWOR.

2.12.10 Κατηγοριοποίηση

Τα ερωτήματα συχνά γράφονται για να συνοψίσουν ή να οργανώσουν πληροφορίες σε κατηγορίες. Για να ομαδοποιήσουμε τις πληροφορίες και να κάνουμε υπολογισμούς, πρέπει να χρησιμοποιήσουμε τις λειτουργίες άθροισης

- *Count*

Η λειτουργία count χρησιμοποιείται για να καθορίσει τον αριθμό των αντικειμένων σε μια ακολουθία.

- *Sum*

Η λειτουργία sum χρησιμοποιείται για να καθορίσει την τελική τιμή των αντικειμένων σε μια ακολουθία.

- *Min and Max*

Οι λειτουργίες min και max χρησιμοποιούνται για να καθορίσουν την ελάχιστη και την μέγιστη τιμή αντίστοιχα των αντικειμένων σε μια ακολουθία.

- *Avg*

Η λειτουργία avg χρησιμοποιείται για να καθορίσει τον μέσο όρο των τιμών των αντικειμένων σε μια ακολουθία.

Οι λειτουργίες sum και avg δέχονται αριθμητικές τιμές ενώ οι λειτουργίες min και max δέχονται τιμές οποιουδήποτε τύπου (strings, dates). Όλες οι λειτουργίες χειρίζονται τα δεδομένα χωρίς τύπο ως αριθμητικά.

2.12.11 Συναρτήσεις

Οι λειτουργίες είναι ένα σημαντικό χαρακτηριστικό της XQuery που επιτρέπει μια ευρεία παράταξη από built-in λειτουργικότητα, καθώς επίσης και την ικανότητα να αναπαράγει και να επαναχρησιμοποιεί κομμάτια των ερωτημάτων. Υπάρχουν δύο είδη λειτουργιών: οι built-in λειτουργίες και οι user-defined λειτουργίες.

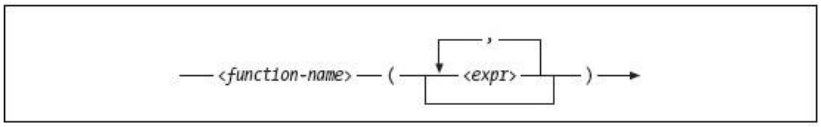
Built-in σε σύγκριση με τις User-Defined Functions

Οι built-in λειτουργίες είναι ένα καθιερωμένο σύνολο, καθορισμένο από την εισήγηση Λειτουργιών και Τελεστών και υποστηρίζεται από όλες τις εφαρμογές της XQuery. Η εισήγηση Λειτουργιών και Τελεστών ορίζει built-in τελεστές εκτός των λειτουργιών. Αυτοί οι τελεστές είναι σαν τις λειτουργίες, αλλά δεν μπορούν να κληθούν απευθείας από ένα ερώτημα· δημιουργούν όμως αντίγραφο ενός συμβόλου ή μιας δεσμευμένης λέξης στην XQuery.

Η user-defined λειτουργία είναι αυτήν που καθορίζεται από τον συντάκτη του ερωτήματος, είτε μέσα στο ίδιο το ερώτημα είτε σε κάποια εξωτερική βιβλιοθήκη.

Calling Functions

Η σύνταξη της κλήσης λειτουργιών είναι ίδια είτε πρόκειται για built-in λειτουργία είτε για user-defined λειτουργία. Αποτελείται από το αρμόδιο όνομα της λειτουργίας, ακολουθούμενο από μια λίστα ορισμάτων μέσα σε παρενθέσεις, διαχωρισμένα με κόμμα. Οι κλήσεις λειτουργιών μπορούν να περιληφθούν οπουδήποτε μια έκφραση το επιτρέπει.



Εικόνα 2.9 Σύνταξη της κλήσης μιας συνάρτησης

2.12.12 Εκφρασεις

Μια έκφραση στην XQuery είναι η βασική ενότητα. Μια έκφραση μπορεί να είναι πολύ απλή αλλά μπορεί να είναι και πολύ σύνθετη. Κάθε έκφραση υπολογίζει ακολουθίες, οι οποίες μπορεί να είναι μια μονή ατομική τιμή, ένας κόμβος, η κενή ακολουθία ή πολλαπλές ατομικές τιμές. Οι εκφράσεις αυτές μπορούν να χωριστούν σε 11 κατηγορίες:

- Primary: literals, κλήσεις συναρτήσεων και εκφράσεις που εσωκλείονται σε παρενθέσεις
- Comparison: σύγκριση βασισμένη σε μια τιμή
- Conditional: if-then-else έκφραση
- Logical: Boolean and/or τελεστής
- Path expressions
- Constructor: πρόσθεση XML στα αποτελέσματα
- FLWOR
- Quantified: καθορισμός αν ακολουθίες εκπληρώνουν τις συνθήκες
- Sequence-related: δημιουργία και συνδυασμός ακολουθιών
- Type-related: έλεγχος τιμών βασισμένες σε τύπο
- Arithmetic: πρόσθεση, αφαίρεση, πολλαπλασιασμός, διαίρεση

❖ **Primary Expressions**

- **Literals**

Ένα literal μπορεί να είναι αριθμητική τιμή ή string. Αυτές οι τιμές αναπαρίστανται απευθείας στα ερωτήματα. Η αναφορά οντοτήτων αντικαθιστά χαρακτήρες που έχουν συγκεκριμένη σημασία στην XQuery. Η αναφορά οντοτήτων ξεκινά με '**&**' ακολουθούμενη από διάφορους χαρακτήρες και τερματίζεται με ' **;** '. Παράδειγμα:

- < : <
- > : >
- & : &

Ένα string μπορεί επίσης να περιέχει μια αναφορά χαρακτήρων. Η αναφορά χαρακτήρων είναι ένα ύφος αναφοράς από XML για ένα Unicode χαρακτήρα. Η δομή της είναι παρόμοια με τη δομή της αναφοράς οντοτήτων, με τη διαφορά ότι στη μέση είναι ο Unicode αριθμός του χαρακτήρα. Για παράδειγμα μπορούμε να αντικαταστήσουμε το χαρακτήρα € με την Unicode αναπαράστασή του: €

- **Variables**

Στην XQuery, μια μεταβλητή αναπαρίσταται με το σύμβολο **$** ακολουθούμενο από ένα QName. Αυτό το όνομα πρέπει να υποστηρίζεται από το XML-qualified. Γι' αυτό το λόγο, θα πρέπει ένα πρόθεμα να προηγείται του ονόματος. Αν ένα πρόθεμα προηγείται του ονόματος, τότε το όνομα συνοδεύεται από ένα namespace. Η μεταβλητή, κατά τη διάρκεια της εκτέλεσης, αναπαριστά μια ειδική ακολουθία. Αντίθετα με τις διαδικαστικές γλώσσες, μια μεταβλητή αναπαριστά μόνο μια τιμή. Δεν μπορούμε να προσδιορίσουμε νέα τιμή σε μια μεταβλητή, οπότε πρέπει να δημιουργήσουμε μια νέα. Με την κλήση της λειτουργίας "sql:variable()" μπορούμε να χρησιμοποιήσουμε μια μεταβλητή από την SQL. Μια μεταβλητή μπορούμε να την ορίσουμε σε οποιοδήποτε σημείο όπως μέσα στους όρους for και let.

❖ Comparison Expressions

Στην XQuery υπάρχουν 4 είδη σύγκρισης: οι τελεστές γενικής σύγκρισης, οι τελεστές σύγκρισης τιμών, οι τελεστές σύγκρισης κόμβων και οι τελεστές σύγκρισης node order. Και τα 4 είδη χρησιμοποιούνται για να συγκρίνουν διαφορετικές τιμές. Για κάθε τελεστή σύγκρισης, αν ένας από τους 2 τελεστέους είναι η κενή ακολουθία, το αποτέλεσμα είναι κενή ακολουθία.

- **General Comparison Operators**

Οι τελεστές γενικής σύγκρισης χρησιμοποιούνται για να συγκρίνουν ατομικές τιμές, ακολουθίες ή συνδυασμό και των δύο. Οι κύριοι τελεστές είναι:

- = : ισοδύναμο
- != : όχι ισοδύναμο
- > : μεγαλύτερο
- < : μικρότερο
- <= : μεγαλύτερο ή ίσο
- <= : μικρότερο ή ίσο

Όταν συγκρίνουμε 2 ακολουθίες, το αποτέλεσμα είναι true αν το σύστημα βρει στη $2^{η}$ ακολουθία μια τιμή που επιστρέφει true όταν συγκριθεί με μια τιμή της $1^{ης}$ ακολουθίας. Μια γενική σύγκριση περιμένει στοιχεία που έχουν ίδιο τύπο σε κάθε πλευρά του τελεστή. Αν μια από τις δύο τιμές είναι untype, δηλώνεται στον τύπο της $2^{ης}$ τιμής. Αν και οι δύο τιμές είναι untype, τότε δηλώνονται στο "xs:string".

- **Value Comparison Operators**

Οι τελεστές σύγκρισης τιμών χρησιμοποιούνται για τη σύγκριση ατομικών τιμών. Ωστόσο, μπορούμε να χρησιμοποιήσουμε και τους τελεστές γενικής σύγκρισης για τη σύγκριση αυτών των τιμών. Οι τελεστές τιμών μπορούν να συγκρίνουν μόνο μονές ατομικές τιμές. Υπάρχουν 6 τελεστές σύγκρισης τιμών:

- eq : ίσο
- ne : όχι ίσο
- gt : μεγαλύτερο
- lt : μικρότερο
- ge : μεγαλύτερο ή ίσο
- le : μικρότερο ή ίσο

Αν θέλουμε να συγκρίνουμε μια τιμή με διαφορετικό τύπο string ή untype τιμή, πρέπει να δηλώσουμε με σαφή τρόπο την untype τιμή. Αυτό συμβαίνει επειδή με αυτόν τον τύπο τελεστών, το σύστημα αυτόματα δηλώνει τις untype τιμές σε τιμές string.

- **Node Comparison Operators**

Ο μοναδικός τελεστής σύγκρισης κόμβων είναι ο '**is**'. Αυτός ο τελεστής είναι μόνο διαθέσιμος για ατομικές τιμές τύπου node. Κάθε τιμή πρέπει να είναι μονό node. Το αποτέλεσμα μιας σύγκρισης κόμβων είναι true μόνο αν και οι δύο τελεστέοι αναπαριστούν τον ίδιο κόμβο.

- **Node Order Comparison Operators**

Η σύγκριση node order συγκρίνει δύο διαφορετικούς κόμβους εάν ο ένας είναι πριν ή μετά από τον άλλον στο xml δέντρο. Υπάρχουν 2 τελεστές:

- <<

Η δομή της σύγκρισης με αυτόν τον τελεστή είναι τελεστέος1 << τελεστέος2. Στόχος αυτής της σύγκρισης είναι να ξέρουμε αν ο τελεστέος1 είναι πριν τον τελεστέο2 στο xml δέντρο.

- >>

Η δομή της σύγκρισης με αυτόν τον τελεστή είναι τελεστέος1 >> τελεστέος2. Στόχος αυτής της σύγκρισης είναι να ξέρουμε αν ο τελεστέος1 είναι μετά τον τελεστέο2 στο xml δέντρο.

❖ Conditional Expressions

Η XQuery διαθέτει τον βρόγχο if-then-else. Το "**if**" παίρνει μια έκφραση υπόθεσης. Αν η υπόθεση είναι αληθής, το σύστημα εκτελεί τις εντολές μέσα στο "**then**". Αν όχι, τότε εκτελεί τις εντολές μέσα στον όρο "**else**". Η έκφραση υπόθεσης πρέπει να είναι μέσα σε παρενθέσεις. Για περισσότερες από μια υποθέσεις μπορούμε να χρησιμοποιήσουμε και τον όρο "**else if**".

❖ Logical Expressions

Οι λογικές εκφράσεις χρησιμοποιούνται κυρίως στον βρόγχο if, στον όρο where ή σε path expression. Οι εκφράσεις αυτές κατασκευάστηκαν με κάποιες

Boolean τιμές και τους τελεστές “**and**” και “**or**”. Μια έκφραση υπόθεσης που κατασκευάστηκε με τον τελεστή “and” όπως “boolean1 and boolean2” επιστρέφει true αν και οι 2 τιμές είναι αληθείς. Μια έκφραση υπόθεσης που κατασκευάστηκε με τον τελεστή “or” επιστρέφει true αν μια από τις 2 ή και οι 2 τιμές είναι αληθείς. Οι εκφράσεις θεωρούνται false όταν υπάρχει μια ψευδής τιμή Boolean, ο αριθμός 0 ή NaN, string μηδενικού μήκους ή κενή ακολουθία.

❖ Constructor Expressions

Υπάρχουν 2 είδη έκφρασης δομητών: οι άμεσες και οι υπολογισμένες εκφράσεις. Αυτές οι εκφράσεις συντάσσουν τα στοιχεία XML.

• Direct Constructors

Στην XQuery η θέληση να συντάξουμε ένα XML έγγραφο είναι πολύ συνηθισμένη. Αυτό μπορούμε να το κάνουμε με έναν άμεσο δομητή στοιχείων. Προσδιορίζει ένα XML στοιχείο χρησιμοποιώντας τη σύνταξη της XML. Μπορούμε να συντάσσουμε ένα στοιχείο με κάποιες σταθερές αλλά η μεγαλύτερη δυνατότητα που μας παρέχει είναι η δυναμική σύνταξη ενός XML εγγράφου. Μπορούμε επίσης να συντάξουμε κάποια χαρακτηριστικά για τα XML αποτελέσματα.

• Computed Constructors

Συνήθως, όταν ξέρουμε το όνομα του χαρακτηριστικού ή του στοιχείου χρησιμοποιούμε τους άμεσους δομητές. Ωστόσο, κάποιες φορές πρέπει να υπολογίσουμε δυναμικά το όνομα του χαρακτηριστικού ή του στοιχείου. Η λύση της XQuery γι’ αυτό είναι οι compute constructors. Είναι πολύ χρήσιμοι όταν:

- Θέλουμε να αντιγράψουμε ένα στοιχείο αλλά με μερικές τροποποιήσεις.
- Θέλουμε να κάνουμε ένα στοιχείο με ένα χαρακτηριστικό του εγγράφου εισόδου.
- Θέλουμε να αναζητήσουμε το όνομα των στοιχείων σε διαφορετικά αρχεία.

Για να συντάξουμε ένα υπολογισμένο στοιχείο, χρησιμοποιούμε τη λέξη κλειδί “element”. Αυτή η λέξη κλειδί ακολουθείται από το όνομα του στοιχείου και κάποια χαρακτηριστικά μέσα σε αγκύλες: “element name {content}”. Μπορούμε επίσης να υπολογίσουμε και το όνομα του στοιχείου αντικαθιστώντας το όνομα με την έκφραση υπολογισμού: “element {expressionToComputeName} {content}.

❖ Quantified Expressions

Υπάρχουν 2 τελεστές για να χτίσουμε μια quantified έκφραση: “**some**” και “**every**”. Και οι 2 χρησιμοποιούν τη δομή “some/every variable in PathExpression satisfies (TestExpression)”. Ο τελεστής some χρησιμοποιείται για να γνωρίζουμε αν κάποιο στοιχείο στο αποτέλεσμα πληρεί τις προϋποθέσεις τις υπόθεσης ενώ ο τελεστής every για να γνωρίζουμε αν όλα τα στοιχεία πληρούν τις προϋποθέσεις. Οι τελεστές αυτοί είναι γνωστοί και ως *quantifier*. Σε μια quantified έκφραση δε μπορούμε να χρησιμοποιήσουμε τη λειτουργία “not”. Μπορούμε όμως να χρησιμοποιήσουμε πολλαπλές μεταβλητές.

❖ Sequence-related Expressions

Αυτές οι εκφράσεις μας επιτρέπουν να συνδυάζουμε αποτελέσματα. Ο τρόπος για τον συνδυασμό διαφορετικών αποτελεσμάτων είναι η δημιουργία ή ο συνδυασμός διαφορετικών ακολουθιών. Υπάρχουν 4 δομητές για τον συνδυασμό ακολουθιών:

- **Sequence Constructor**

Ο πρώτος και πιο εύκολος τρόπος για να συνδυάσουμε 2 ή περισσότερες ακολουθίες είναι η δημιουργία μιας ακολουθίας με τις άλλες πρώτες. Για να γίνει αυτό, χρησιμοποιούμε το δομητή ακολουθίας. Η δομή αυτού του δομητή είναι πολύ απλή, βάζουμε τις ακολουθίες μέσα σε παρενθέσεις διαχωρισμένες με κόμμα: (sequence1, sequence2, sequence3, …). Το αποτέλεσμα διατηρεί την ίδια σειρά όπως στον τελεστή ακολουθίας. Ο δομητής ακολουθίας μπορεί να συνδέσει αλυσιδωτά ατομικές τιμές και είναι ο μόνος που μπορεί να το κάνει αυτό.

- **Union Expression**

Για να συνδέσουμε δύο ακολουθίες μπορούμε να χρησιμοποιήσουμε επίσης και μια έκφραση ένωσης. Αυτήν η έκφραση συντάσσεται με τον τελεστή “**|**” ή με τη λέξη κλειδί “**union**”. Η ένωση διαγράφει τις διπλότυπες τιμές από το αποτέλεσμα, αντίθετα με την απλή σύνδεση που δεν τις διαγράφει.

- **Intersect Expression**

Η έκφραση τομής συνδυάζει όλες τις τιμές που έχουν όλες οι ακολουθίες αυτής της έκφρασης. Χρησιμοποιεί τη λέξη κλειδί “**intersect**”. Όπως και στην ένωση, η τομή διαγράφει όλες τις διπλότυπες τιμές.

- **Except Expression**

Η έκφραση εξαίρεσης επιλέγει όλα τα δεδομένα που υπάρχουν στη πρώτη ακολουθία και δεν υπάρχουν στη δεύτερη. Για τη σύνταξη αυτής της έκφρασης χρησιμοποιείται η λέξη κλειδί "**except**". Όπως και στις δυο προηγούμενες εκφράσεις, έτσι και η έκφραση εξαίρεσης διαγράφει τις διπλότυπες εγγραφές.

❖ **Type-related Expressions**

Οι type-related εκφράσεις είναι όλες οι εκφράσεις που τροποποιούν, ελέγχουν και επιβεβαιώνουν έναν τύπο στην XQuery.

- **The "instance of" Expression**

Στην XQuery, κάθε τιμή είναι μια ακολουθία και σε κάθε ακολουθία αντιστοιχεί ένας ακολουθιακός τύπος. Ο τελεστής instance δε δηλώνει μια ακολουθία αλλά ελέγχει αν στην ακολουθία αντιστοιχεί ένας ειδικός ακολουθιακός τύπος. Αν αντιστοιχεί αυτός ο τύπος, ο τελεστής επιστρέφει true, αλλιώς επιστρέφει false. Η αντιστοίχηση ακολουθιακού τύπου δεν περιλαμβάνει αριθμητικούς τύπους.

- **Casting**

Η έκφραση δήλωσης κάνει την ίδια δουλειά με το δομητή αλλά η σύνταξή της είναι διαφορετική. Άλλη μια διαφορά είναι ότι μπορεί να χρησιμοποιηθεί με τύπους ονομάτων που δεν είναι σε namespace. Η έκφραση χρησιμοποιεί τον τελεστή "**cast as**" για τη δήλωση μιας τιμής και η δομή της είναι: "value cast as AtomicType". Μια δήλωση δε μπορεί να τροποποιήσει τον τύπο μιας ακολουθίας με περισσότερα από ένα αντικείμενα αλλά μπορούμε να βάλουμε τη δήλωση σε μια path expression ώστε να δηλώσουμε το κάθε στοιχείο ένα προς ένα.

- **Typeswitch Expression**

Μια έκφραση typeswitch μοιάζει με την συνθήκη switch της γλώσσας C. Ενώ με τη συνθήκη switch, επιλέγουμε τη σωστή διαδικασία για την εκτέλεση με βάση τη τιμή ενός στοιχείου, με την έκφραση typeswitch επιλέγουμε με βάση τον τύπο της ακολουθίας και όχι την τιμή της. Η δομή της έκφρασης είναι:

"typesswitch(expression)
Case $VariableName as sequence type return expression
Case $VariableName as sequence type return expression
….
Default $VariableName return expression"

- **Treat Expression**

Αυτήν η έκφραση εκτελεί κάποιες εντολές μόνο εάν αυτές οι εντολές έχουν συγκεκριμένο τύπο. Η δομή της έκφρασης είναι: “expression treat as SequenceType”. Η έκφραση αυτή μπορεί να αντικαταστήσει μια έκφραση όπως:

```
"If (expression instance of SequenceType)
then expression
else()"
```

❖ **Arithmetic Expressions**

Στην XQuery, υπάρχουν 4 βασικοί αριθμητικοί τύποι: “xs:integer”, “xs:double”, “xs:decimal”, “xs:float”. Με αυτούς τους τύπους και τους αριθμητικούς τελεστές μπορούμε να συντάξουμε πολλές διαφορετικές αριθμητικές εκφράσεις όπως σύγκριση τιμών, πρόσθεση, αφαίρεση, πολλαπλασιασμό ή διαίρεση.

2.13 Query για SQL χρήστες

Αυτή η παράγραφος μας δίνει τις πληροφορίες που χρειαζόμαστε για να χρησιμοποιήσουμε την SQL καθώς και τις συσχετιζόμενες βάσεις δεδομένων. Επίσης συγκρίνει την SQL με την XQuery τόσο σε σχέση με τα μοντέλα δεδομένων όσο και με την σύνταξή τους. Τέλος μας τονίζει σημεία στα οποία μπορούμε να χρησιμοποιήσουμε την SQL με την XQuery μαζί, και περιγράφει τον ρόλο της SQL/XML.

2.13.1 Συγκρίνοντας την σύνταξη της SQL με την αντίστοιχη της XQuery

Αυτή η παράγραφος συγκρίνει την σύνταξη της SQL με την αντίστοιχη της XQuery έτσι ώστε να μας εξηγήσει για ποιους λόγους θα μπορούσαμε να χρησιμοποιήσουμε την XQuery. Υπάρχουν ομοιότητες μεταξύ της SQL και της XQuery, το οποίο δεν είναι σύμπτωση επειδή μερικοί από τους προγραμματιστές οι οποίοι εργάστηκαν για την ανάπτυξη της SQL εργάστηκαν και για την XQuery. Σε αυτή την παράγραφο δεν αναπτύσσεται όλη η σύνταξη της SQL, αλλά μόνο οι πιο συχνά χρησιμοποιούμενες δομές.

2.13.2 Ένα απλό Query

Για να συγκρίνουμε τα SQL και XQuery queries, θα ξεκινήσουμε με το απλό έγγραφο που περιέχει τον κατάλογο των προϊόντων. Ένα βασικό SQL query πιθανώς επιλέγει όλες τις τιμές από τον πίνακα οι οποίες ανταποκρίνονται σε κάποια κριτήρια, όπως για παράδειγμα στο ACC department. Η SQL έκφραση η οποία το πετυχαίνει αυτό είναι η εξής:

```
select * from catalog
where dept='ACC'
```

Στην XQuery, μπορούμε να χρησιμοποιήσουμε κατευθείαν ένα path expression για ένα τόσο απλό query, όπως για παράδειγμα:

```
doc("catalog2.xml")//product[@dept='ACC']
```

Εάν δεν θέλουμε να ταξινομήσουμε τα αποτελέσματα ή να δομήσουμε νέα στοιχεία, είναι συνήθως πιο απλό (και πιθανώς γρηγορότερο) να χρησιμοποιήσουμε ένα path expression. Όμως, μπορούμε να χρησιμοποιήσουμε μια πλήρες έκφραση FLWOR η οποία χρησιμοποιεί μια πρόταση where παρόμοια όπως στην SQL, για παράδειγμα:

```
for $prod in doc("catalog2.xml")//product
where $prod/@dept='ACC'
return $prod
```

Στην πρόταση where, χρειάζεται να ξεκινήσουμε την αναφορά στο dept χαρακτηριστικό με το $prod/ έτσι ώστε να δώσουμε σε αυτό κάποιο περιεχόμενο. Αυτή είναι η διαφορά με την SQL, στην οποία εάν υπάρχει μόνο μια στήλη dept στους πίνακες ή στον πίνακα μας του query, υποτίθεται ότι υπάρχει μόνο αυτή η μία στήλη. Στην XQuery, πρέπει να είμαστε σαφείς για το πού το dept χαρακτηριστικό εμφανίζεται, επειδή μπορεί να εμφανίζεται σε οποιοδήποτε επίπεδο του εγγράφου μας.
Τώρα ας υποθέσουμε ότι θέλουμε να ταξινομήσουμε τις τιμές σύμφωνα με το νούμερο προϊόντος. Στην SQL, απλά θα προσθέταμε μια πρόταση order by, όπως για παράδειγμα:

```
select * from catalog
where dept='ACC'
order by number
```

Μπορούμε επίσης να προσθέσουμε μια πρόταση order by στο FLWOR, ξανά δίνοντάς το ως περιεχόμενο, όπως στο:

```
for $prod in doc("catalog2.xml")//product
where $prod/@dept='ACC'
order by $prod/number
return $prod
```

2.13.3 Συνθήκες και Τελεστές

Οι συνθήκες και οι τελεστές χρησιμοποιούνται για να φιλτράρουν τα αποτελέσματα των queries. Αρκετές από τις συνθήκες και τους τελεστές οι οποίοι είναι διαθέσιμοι στην SQL μπορούν επίσης να χρησιμοποιηθούν και στην

XQuery, παρόλο που μερικές φορές χρειάζονται μια τροποποίηση την σύνταξη τους.

Συγκρίσεις

Το παράδειγμα πριν χρησιμοποιούσε το σύμβολο ισότητας = για να συγκρίνει την τιμή του department με το ACC. Η XQuery έχει τους ίδιους τελεστές ισότητας όπως και η SQL, οι οποίοι είναι: =, !=, <, <=, >, και >= . Η έκφραση BETWEEN στην SQL δεν υποστηρίζεται επακριβώς, αλλά μπορούμε να χρησιμοποιήσουμε δύο συγκρίσεις όπως στο:

```
for $prod in doc("catalog2.xml")//product
where $prod/number > 500 and $prod/number < 700
return $prod
```

Όπως στην SQL, τα quotes χρησιμοποιούνται για να περικλείσουν τιμές, αν και δεν χρησιμοποιούνται για περικλείσουν αριθμητικές τιμές.
Οι χαρακτήρες στην SQL μπορούν επίσης να συνταιριαστούν με wildcards χρησιμοποιώντας μια LIKE συνθήκη. Για παράδειγμα το query:

```
select * from catalog
where name LIKE 'F%'
```

το οποίο θα μας επιστρέψει τα προϊόντα των οποίων τα ονόματα ξεκινούν με το γράμμα F. Η XQuery μας παρέχει μια συνάρτηση η οποία θα μπορούσε να είναι ιδιαίτερα χρήσιμη σε αυτή τη συγκεκριμένη περίπτωση. Για παράδειγμα:

```
for $prod in doc("catalog2.xml")//product
where starts-with($prod/name,'F')
return $prod
```

Στην πιο γενική περίπτωση, οι συναρτήσεις αυτές μας δίνουν την δυνατότητα να ταιριάξουμε ένα χαρακτήρα με μία οποιαδήποτε συνηθισμένη έκφραση. Στις γενικές εκφράσεις, μια απλή τελεία (.) αντιπροσωπεύει έναν οποιοδήποτε χαρακτήρα, όπως ακριβώς η κάτω παύλα (_) στις LIKE συνθήκες. Προσθέτοντας έναν αστερίσκο μετά από μια τελεία (.*) έχουμε οποιαδήποτε επανάληψη οποιουδήποτε χαρακτήρα, όπως και με τον χαρακτήρα % στις LIKE συνθήκες. Οι χαρακτήρες ^ και $ μπορούν να χρησιμοποιηθούν για να μας υποδείξουν την αρχή και το τέλος ενός χαρακτήρα. Ένας άλλος τρόπος έκφρασης του προηγούμενου query είναι η:

```
for $prod in doc("catalog2.xml")//product
where matches($prod/name,'^F.*')
return $prod
```

Οι συνηθισμένες εκφράσεις είναι πολύ πιο ισχυρές από τις LIKE συνθήκες.

Αποτίμηση Like συνθηκών σε σύγκριση με τις regular εκφράσεις

Έκφραση Like	Ισοδύναμη regular έκφραση	Αποτίμηση τιμών
xyz	^xyz$	xyz
xyz_	^xyz.$	xyza
xyz%	^xyz.*$	xyz, xyza, xyzaaa
_xyz	^.xyz$	axyz
xyz%	^xyz	xyz, xyza, xyzaa
%xyz	xyz$	xyz, axyz, aaxyz
x_yz	^x.yz$	xayz
x%yz	^x.*yz$	xyz, xayz, xaayz

Κεφάλαιο 3

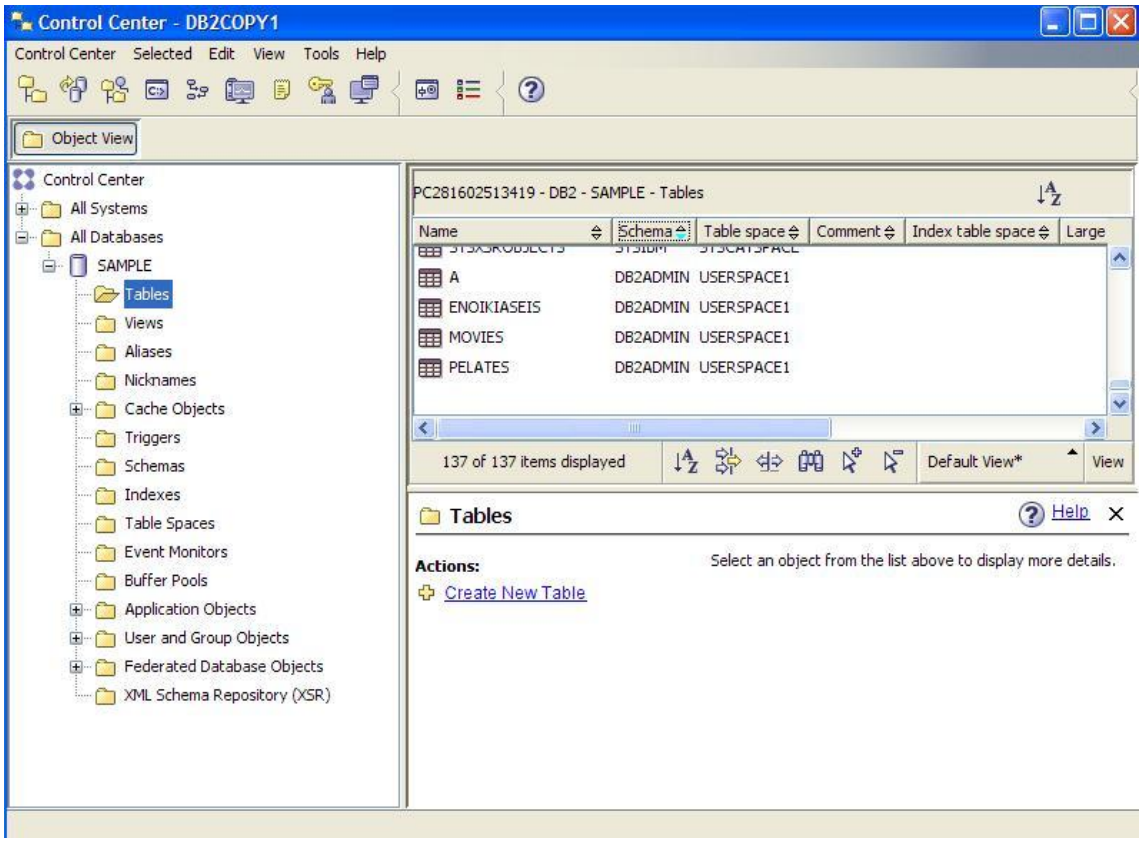
Εκτέλεση ερωτημάτων XQuery

Στόχος αυτού του κεφαλαίου είναι να παρουσιάσει στον αναγνώστη τον τρόπο που επιλέξαμε για να εκτελέσουμε τα ερωτήματα XQuery, καθώς και πώς εμφανίσαμε τα αποτελέσματα των ερωτημάτων.

3.1 Εισαγωγή στην εκτέλεση των ερωτημάτων XQuery

Για την εκτέλεση των ερωτημάτων XQuery χρησιμοποιήθηκε η DB2. Στην DB2 δημιουργήθηκε η βάση δεδομένων “Sample” στην οποία περιέχονται οι πίνακες που θα χρησιμοποιηθούν σαν βάση δεδομένων για την εκτέλεση των ερωτημάτων. Στη βάση δεδομένων “Sample” περιέχονται οι πίνακες: Movies ,Pelates και Enoikiaseis.

Ακολουθεί στην εικόνα 3.1 το περιβάλλον της DB2:

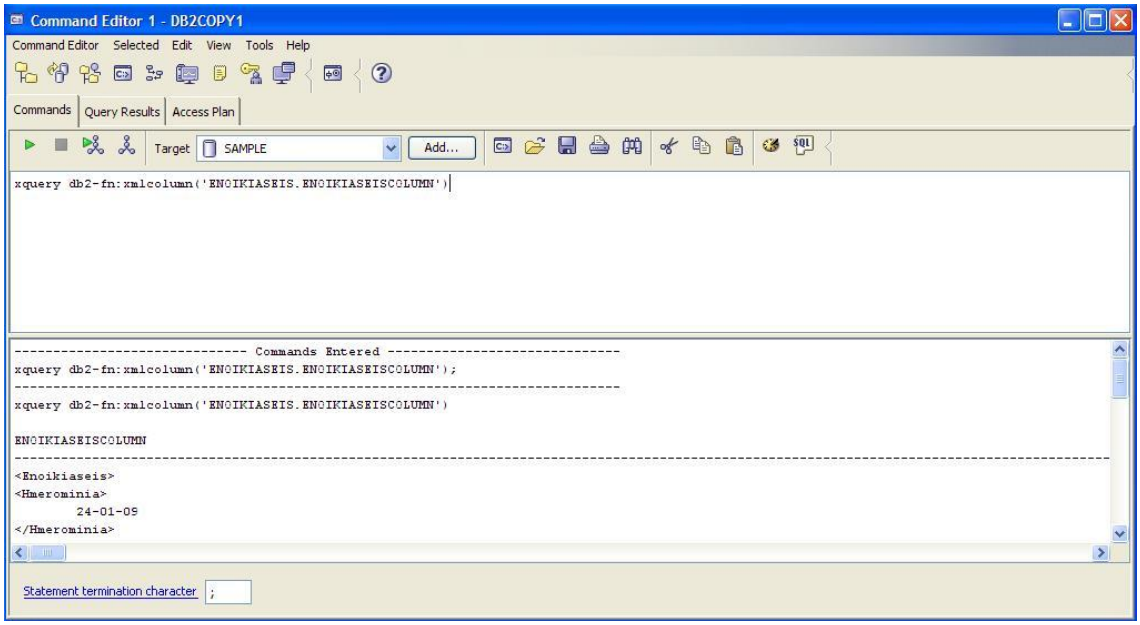


Εικόνα 3.1 *Περιβάλλον της DB2*

Μπορούμε να παρατηρήσουμε στα αριστερά του παραθύρου το λειτουργικό μας σύστημα (στο παράδειγμά μας είναι το PC281602513419) όπως και την βάση δεδομένων που έχουμε δημιουργήσει με όλες τις λεπτομέρειες της όπως: τους πίνακες, τα σχήματα κ.τ.λ..

Για να εκτελέσουμε ένα ερώτημα θα πρέπει να πατήσουμε στο κουμπί Command Editor ώστε να εμφανιστεί το παράθυρο στο οποίο μπορούμε να δημιουργήσουμε τα ερωτήματα που θέλουμε να εκτελέσουμε.

Στην εικόνα 3.2 παρουσιάζεται το περιβάλλον του Command Editor:



Εικόνα 3.2 *Command Editor*

Όπως μπορούμε να παρατηρήσουμε στο πρώτο πλαίσιο εισάγονται τα ερωτήματα που θέλουμε να εκτελεστούν, ενώ στο δεύτερο πλαίσιο εμφανίζονται τα αποτελέσματα της εκτέλεσης των ερωτημάτων.

3.2 Δημιουργία των πινάκων και εισαγωγή τιμών

Στην DB2 είναι απαραίτητο να δημιουργήσουμε τους πίνακες από τους οποίους θα αντλούμε τα δεδομένα για να τρέξουμε τα ερωτήματά μας. Παρακάτω θα παραθέσουμε τον κώδικα που απαιτείται για την δημιουργία των πινάκων.

```
create table Movies(
Movid INTEGER NOT NULL PRIMARY KEY,
Moviecolumn xml);
```

Όπως παρατηρούμε στο συγκεκριμένο παράδειγμα δημιουργούμε έναν πίνακα που τον ονομάζουμε Movies. Επίσης αυτός ο πίνακας περιέχει δύο στοιχεία το ένα είναι το Movid το οποίο αποτελεί τον κωδικό αριθμό της συγκεκριμένης ταινίας για να ξεχωρίζει από τις υπόλοιπες και το δεύτερο στοιχείο είναι η xml στήλη Moviecolumn στην οποία θα αποθηκευτούν όλα τα χαρακτηριστικά της κάθε ταινίας όπως για παράδειγμα ο τίτλος της, η κατηγορία της κ.α..

Για να εισάγουμε τις τιμές σε έναν πίνακα ο κώδικας που απαιτείται είναι ο εξής:

```
insert into Movies(Movid,Moviecolumn) values (1000,
'<Movie>
<Titlos>Kung Fu Panda</Titlos>
<Eidos>paidiki</Eidos>
<Katigoria>hmeras</Katigoria>
<Etairia>ODEON</Etairia>
</Movie>');
```

Χρησιμοποιώντας τον πίνακα του προηγούμενου παραδείγματος βλέπουμε πώς γίνεται η εισαγωγή των τιμών. Όπως παρατηρούμε εισάγουμε στο στοιχείο Movid του πίνακα Movies την τιμή 1000 ενώ στην συνέχεια για να εισάγουμε στην xml στήλη τα στοιχεία και τα δεδομένα θα πρέπει πρώτα να δημιουργήσουμε έναν κόμβο τον οποίο τον ονομάζουμε Movie. Στην xml στήλη δηλώνεται το στοιχείο Titlos με τιμή “Kung Fu Panda”, το στοιχείο Eidos με τιμή “paidiki”, το στοιχείο Katigoria με τιμή “hmeras” και τέλος το στοιχείο Etairia με τιμή “ODEON”.

Ακολουθεί ο κώδικας με τους πίνακες που δημιουργήσαμε στην DB2 για να γίνουν πιο κατανοητά αυτά που μόλις αναφέραμε.

Δημιουργία πίνακα Pelates:

```
create table Pelates(
Pelid INTEGER NOT NULL PRIMARY KEY,
Pelatescolumn xml);
```

Εισαγωγή τιμών στον πίνακα Pelates:

```
insert into Pelates(Pelid,Pelatescolumn) values (0002,
'<Pelates>
<Epwnimo>Papadopoulos</Epwnimo>
<Onoma>Kiriakos</Onoma>
<Dieuthinsi>Oxi</Dieuthinsi>
<Tilefwno>2310123457</Tilefwno>
</Pelates>');
```

Δημιουργία πίνακα Enoikiaseis ο οποίος έχει αναφορά στους δύο προηγούμενους πίνακες:

```
CREATE TABLE Enoikiaseis(
Movid INTEGER NOT NULL REFERENCES Movies(Movid),
Pelid INTEGER NOT NULL REFERENCES Pelates(Pelid),
Enoikiaseiscolumn xml,
PRIMARY KEY(Movid,Pelid));
```

Εισαγωγή τιμών στον πίνακα Enoikiaseis:

```
insert into Enoikiaseis(Movid,Pelid,Enoikiaseiscolumn) values (1000,0001,
'<Enoikiaseis>
<Hmerominia>24-01-09</Hmerominia>
</Enoikiaseis>');
```

3.3 Εκτέλεση των ερωτημάτων XQuery

Για να εκτελέσουμε ένα XQuery ερώτημα στην DB2 θα χρησιμοποιήσουμε την συνάρτηση db2-fn:xmlcolumn. Επίσης αναγκαίο για την εκτέλεση των ερωτημάτων μας είναι πριν από την δημιουργία των ερωτημάτων μας να υπάρχει η λέξη “xquery”. Όπως αναφέραμε απαραίτητη είναι και η συνάρτηση db2-fn:xmlcolumn η οποία ακολουθεί της λέξης “xquery”. Μετά θα πρέπει να ακολουθεί ο πίνακας από τον οποίο θέλουμε να αντλήσουμε τα δεδομένα μας. Αν δεν θέλουμε να εμφανίσουμε όλα τα χαρακτηριστικά του πίνακα θα πρέπει να δώσουμε την διαδρομή του χαρακτηριστικού που θέλουμε να εμφανίσουμε.

Θα δώσουμε τώρα μερικά παράδειγματα εκτέλεσης XQuery ερωτημάτων:

1. xquery db2-fn:xmlcolumn('MOVIES.MOVIECOLUMN')

Επεξήγηση:

Εμφανίζει όλες τις ταινίες που έχουν καταχωρηθεί στον πίνακα Movies

Αποτελέσματα:

```
MOVIECOLUMN
<Movie>
<Titlos>
        Kung Fu Panda
</Titlos>
<Eidos>
        paidiki
</Eidos>
<Katigoria>
        hmeras
</Katigoria>
<Etairia>
        ODEON
</Etairia>
</Movie>
<Movie>
<Titlos>
        Ice Age
</Titlos>
<Eidos>
        paidiki
</Eidos>
<Katigoria>
        trihmerou
</Katigoria>
<Etairia>
        Audio Visual
</Etairia>
</Movie>
```

<Movie>
<Titlos>
Horton
</Titlos>
<Eidos>
paidiki
</Eidos>
<Katigoria>
evdomadas
</Katigoria>
<Etairia>
Prooptiki
</Etairia>
</Movie>
<Movie>
<Titlos>
The Alibi
</Titlos>
<Eidos>
kwmwdia
</Eidos>
<Katigoria>
evdomadas
</Katigoria>
<Etairia>
Audio Visual
</Etairia>
</Movie>
<Movie>
<Titlos>
Get Smart
</Titlos>
<Eidos>
kwmwdia
</Eidos>
<Katigoria>
hmeras
</Katigoria>
<Etairia>
Odeon
</Etairia>
</Movie>
<Movie>
<Titlos>
Zohan
</Titlos>
<Eidos>
kwmwdia
</Eidos>
<Katigoria>
trihmerou
</Katigoria>
<Etairia>
Prooptiki
</Etairia>
</Movie>
<Movie>
<Titlos>
Righteous Kill
</Titlos>
<Eidos>
peripeteia
</Eidos>
<Katigoria>
hmeras
</Katigoria>
<Etairia>
Prooptiki
</Etairia>
</Movie>
<Movie>
<Titlos>
Cleaner
</Titlos>
<Eidos>
peripeteia

```
</Eidos>
<Katigoria>
	trihmerou
</Katigoria>
<Etairia>
	Odeon
</Etairia>
</Movie>
<Movie>
<Titlos>
	Count Of Mode Cristo
</Titlos>
<Eidos>
	peripeteia
</Eidos>
<Katigoria>
	evdomadas
</Katigoria>
<Etairia>
	Audio Visual
</Etairia>
</Movie>
<Movie>
<Titlos>
	The Name Of The Rose
</Titlos>
<Eidos>
	thriller
</Eidos>
<Katigoria>
	evdomadas
</Katigoria>
<Etairia>
	Audio Visual
</Etairia>
</Movie>
<Movie>
<Titlos>
	Seven
</Titlos>
<Eidos>
	thriller
</Eidos>
<Katigoria>
	trihmerou
</Katigoria>
<Etairia>
	Odeon
</Etairia>
</Movie>
<Movie>
<Titlos>
	The Strangers
</Titlos>
<Eidos>
	thriller
</Eidos>
<Katigoria>
	hmeras
</Katigoria>
<Etairia>
	Prooptiki
</Etairia>
</Movie>
```

2. xquery

```
for $x in db2-fn:xmlcolumn('MOVIES.MOVIECOLUMN')/Movie/Katigoria
return $x
```

Επεξήγηση:

Επιστρέφει για κάθε ταινία τι κατηγορία είναι

Αποτελέσματα:

```
<Katigoria>
hmeras
</Katigoria>
<Katigoria>
trihmerou
</Katigoria>
<Katigoria>
evdomadas
</Katigoria>
<Katigoria>
evdomadas
</Katigoria>
<Katigoria>
hmeras
</Katigoria>
<Katigoria>
trihmerou
</Katigoria>
<Katigoria>
hmeras
</Katigoria>
<Katigoria>
trihmerou
</Katigoria>
<Katigoria>
evdomadas
</Katigoria>
<Katigoria>
evdomadas
</Katigoria>
<Katigoria>
trihmerou
</Katigoria>
<Katigoria>
hmeras
</Katigoria>
```

3. xquery

```
for $x in db2-fn:xmlcolumn('MOVIES.MOVIECOLUMN')/Movie
where $x/Eidos="peripeteia"
return $x;
```

Επεξήγηση:

Επιστρέφει τις ταινίες που ανήκουν στο είδος 'peripeteia'

Αποτελέσματα:

```
<Movie>
<Titlos>
        Righteous Kill
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        hmeras
</Katigoria>
<Etairia>
        Prooptiki
</Etairia>
```

```
</Movie>
<Movie>
<Titlos>
        Cleaner
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        trihmerou
</Katigoria>
<Etairia>
        Odeon
</Etairia>
</Movie>
<Movie>
<Titlos>
        Count Of Mode Cristo
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        evdomadas
</Katigoria>
<Etairia>
        Audio Visual
</Etairia>
</Movie>
```

4. xquery

```
for $x in db2-fn:xmlcolumn('MOVIES.MOVIECOLUMN')/Movie
where $x/Katigoria="evdomadas" or $x/Eidos="peripeteia"
return $x;
```

Επεξήγηση:

Επιστρέφει τις ταινίες που ανήκουν στην κατηγορία εβδομάδας ή ανήκουν στο είδος περιπέτεια

Αποτελέσματα:

```
<Movie>
<Titlos>
        Horton
</Titlos>
<Eidos>
        paidiki
</Eidos>
<Katigoria>
        evdomadas
</Katigoria>
<Etairia>
        Prooptiki
</Etairia>
</Movie>
<Movie>
<Titlos>
        The Alibi
</Titlos>
<Eidos>
        kwmwdia
</Eidos>
<Katigoria>
        evdomadas
</Katigoria>
<Etairia>
        Audio Visual
</Etairia>
```

```
</Movie>
<Movie>
<Titlos>
        Righteous Kill
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        hmeras
</Katigoria>
<Etairia>
        Prooptiki
</Etairia>
</Movie>
<Movie>
<Titlos>
        Cleaner
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        trihmerou
</Katigoria>
<Etairia>
        Odeon
</Etairia>
</Movie>
<Movie>
<Titlos>
        Count Of Mode Cristo
</Titlos>
<Eidos>
        peripeteia
</Eidos>
<Katigoria>
        evdomadas
</Katigoria>
<Etairia>
        Audio Visual
</Etairia>
</Movie>
<Movie>
<Titlos>
        The Name Of The Rose
</Titlos>
<Eidos>
        thriller
</Eidos>
<Katigoria>
        evdomadas
</Katigoria>
<Etairia>
        Audio Visual
</Etairia>
</Movie>
```

5. xquery

```
for $x in db2-fn:xmlcolumn('MOVIES.MOVIECOLUMN')/Movie
where $x/Eidos="peripeteia" or $x/Katigoria="hmeras"
return <titlos>{$x/Titlos}</titlos>
```

Επεξήγηση:

Εμφανίζει στοιχεία "titlos" στα οποία περιέχονται οι τίτλοι των ταινιών που ανήκουν στο είδος περιπέτεια ή στην κατηγορία ημέρας

Αποτελέσματα:

```
<titlos>
<Titlos>
        Kung Fu Panda
</Titlos>
</titlos>
<titlos>
<Titlos>
        Get Smart
</Titlos>
</titlos>
<titlos>
<Titlos>
        Righteous Kill
</Titlos>
</titlos>
<titlos>
<Titlos>
        Cleaner
</Titlos>
</titlos>
<titlos>
<Titlos>
        Count Of Mode Cristo
</Titlos>
</titlos>
<titlos>
<Titlos>
        The Strangers
</Titlos>
</titlos>
```

Κεφάλαιο 4

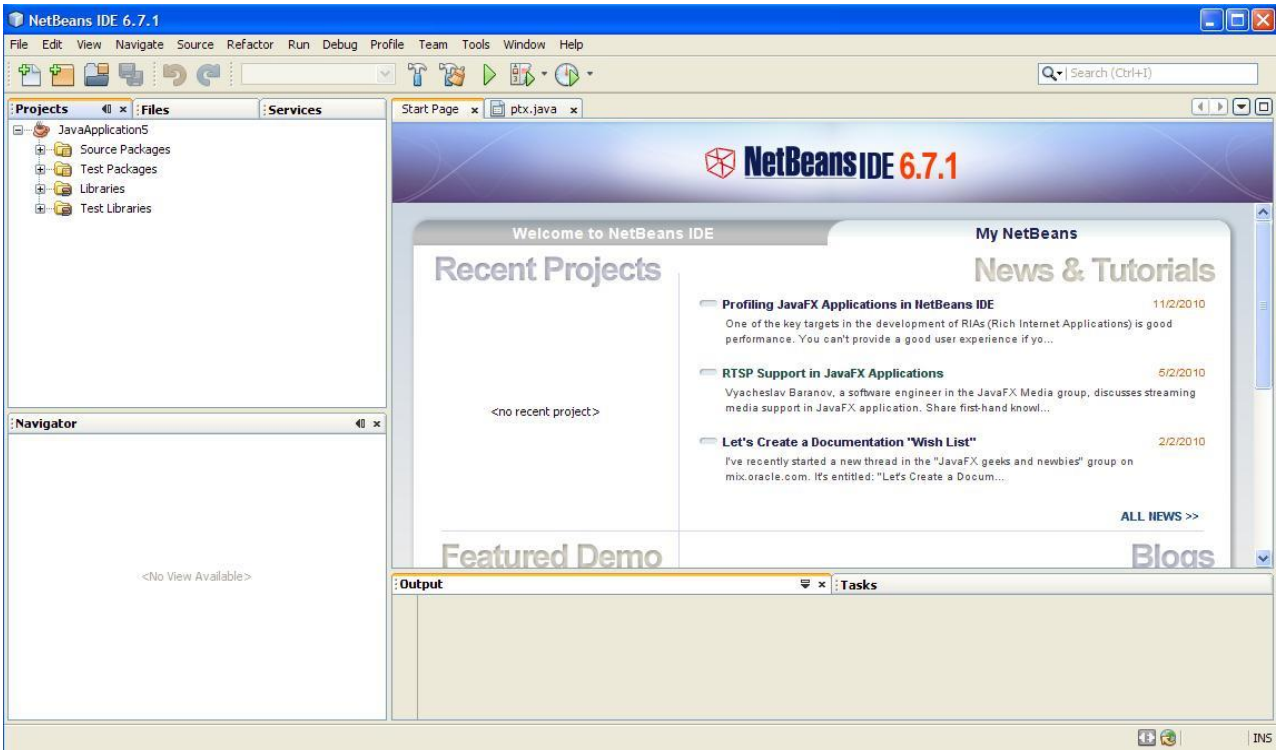
Παρουσίαση του προγράμματος NetBeans

Σε αυτό το κεφάλαιο παρουσιάζεται το πρόγραμμα NetBeans στο οποίο δημιουργήσαμε την εφαρμογή μας.

4.1 Παρουσίαση του NetBeans

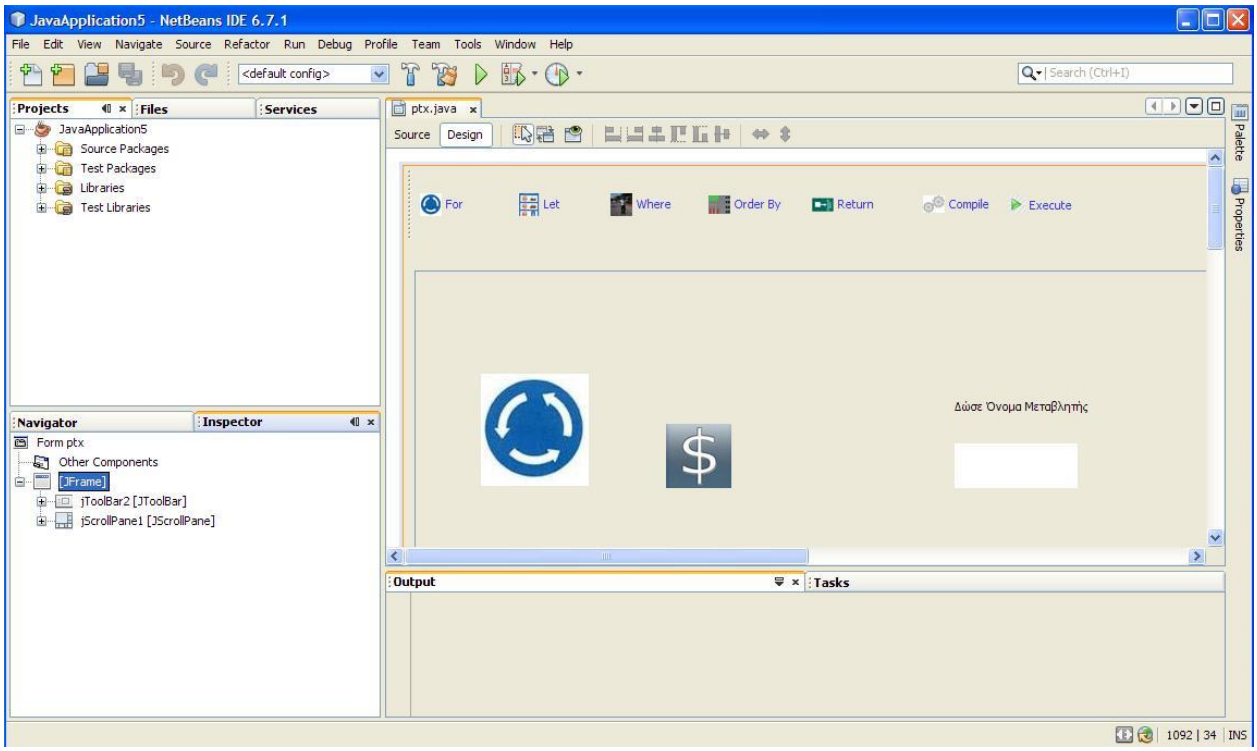
Για τη δημιουργία της εφαρμογής εργαστήκαμε στο περιβάλλον του NetBeans. Το NetBeans είναι ένα πρόγραμμα στο οποίο ο καθένας μπορεί να δημιουργήσει την εφαρμογή που επιθυμεί δίνοντάς την πληθώρα δυνατοτήτων. Βασικό πλεονέκτημα του NetBeans είναι ότι μας προσφέρει την δυνατότητα δημιουργίας διάφορων εργαλείων στην εφαρμογή μας όπως για παράδειγμα κουμπιά, toolbars, text areas κ.α. χωρίς να απαιτείται η συγγραφή κάποιου κώδικα. Δηλαδή ο κώδικας δημιουργείται αυτόματα από το NetBeans χρησιμοποιώντας τις κατάλληλες μεθόδους. Επίσης μπορούμε να δώσουμε στα εργαλεία λειτουργικότητα χωρίς πάλι να απαιτείται από τον χρήστη να γράψει κώδικα java swing αλλά όλα γίνονται αυτοματοποιημένα επιλέγοντας ο χρήστης τι ακριβώς θέλει να κάνει το συγκεκριμένο εργαλείο. Έτσι από τον χρήστη απαιτείται να γράψει πολύ λίγο κώδικα.

Ακολουθεί η εικόνα 4.1 που παρουσιάζει το περιβάλλον του NetBeans:



Εικόνα 4.1 *Περιβάλλον του NetBeans*

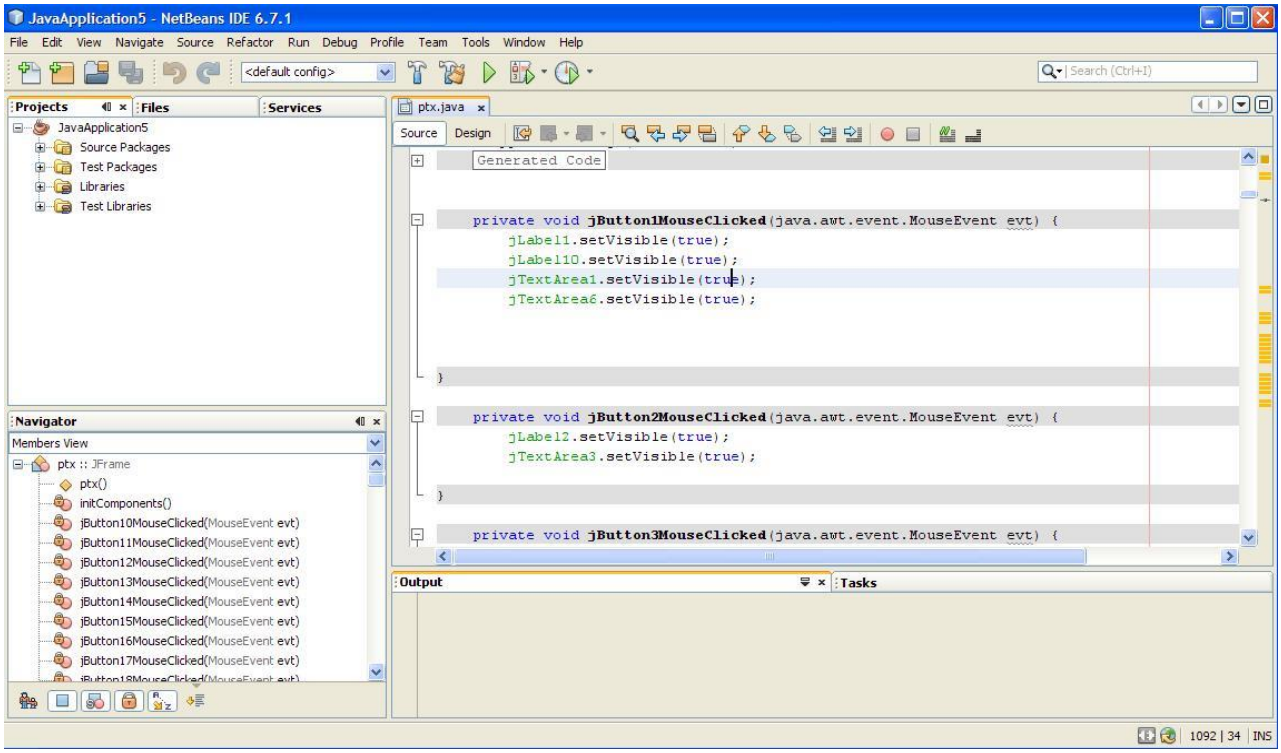
Αυτό είναι και το περιβάλλον μόλις εκτελέσουμε το NetBeans. Πάνω αριστερά όπως μπορούμε να δούμε βρίσκεται η εφαρμογή που έχουμε δημιουργήσει (“java application5”) με τα χαρακτηριστικά της. Μόλις ανοίξουμε την εφαρμογή που έχουμε δημιουργήσει το παράθυρο που εμφανίζεται είναι το παρακάτω:



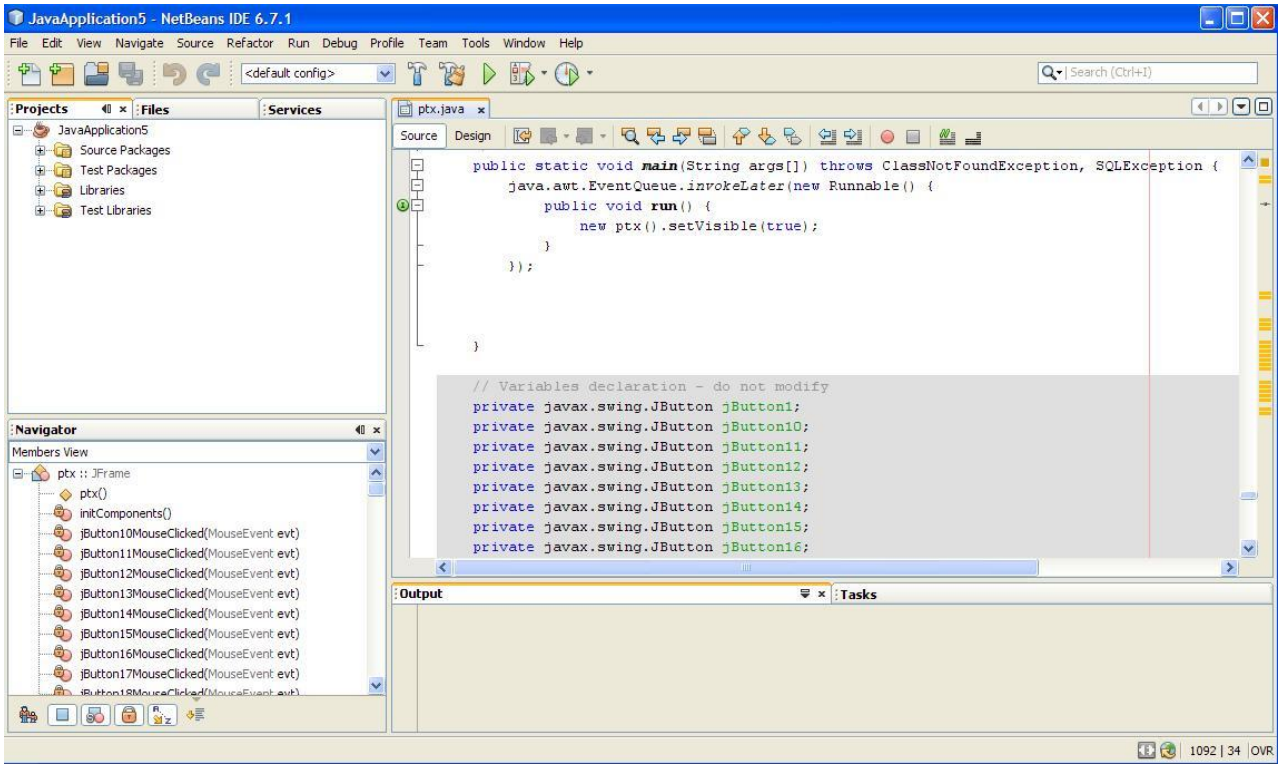
Εικόνα 4.2 *Αρχικό παράθυρο της εφαρμογής*

Όπως αναφέραμε και προηγουμένως πάνω αριστερά βρίσκεται το όνομα της εφαρμογής που έχουμε δημιουργήσει με τα χαρακτηριστικά της. Κάτω και αριστερά βρίσκονται τα εργαλεία που έχουμε δημιουργήσει στην εφαρμογή μας. Παρατηρούμε ότι το NetBeans έχει δύο όψεις, μία design (όπως μπορεί να φανεί στην προηγούμενη εικόνα) που είναι η παλέτα στην οποία δημιουργούμε τα εργαλεία της εφαρμογής μας, και μία source στην οποία περιέχεται ο κώδικας που έχει χρησιμοποιηθεί στην εφαρμογή μας.

Στις παρακάτω εικόνες εμφανίζεται η όψη source:



Εικόνα 4.3 *Όψη source*



Εικόνα 4.4 *Όψη source*

Κεφάλαιο 5

Παρουσίαση των γραφικών και του προγράμματος σύνταξης και εκτέλεσης ερωτημάτων XQuery

Σε αυτό το κεφάλαιο παρουσιάζεται η εφαρμογή που δημιουργήσαμε ώστε να βοηθήσουμε ακόμα και τον χρήστη που δεν είναι εξοικειωμένος με την σύνταξη και την εκτέλεση των ερωτημάτων XQuery να μπορέσει να συντάξει και να εκτελέσει το δικό του XQuery ερώτημα. Θα αναφερθούν λεπτομέρειες για τα χαρακτηριστικά της εφαρμογής μας καθώς και τον τρόπο λειτουργία της.

5.1 Μεταφορές - Metaphors

Είναι πολύ σημαντικό σε αυτό το κεφάλαιο να κατανοήσουμε πρώτα απ' όλα τι είναι ο metaphor και γιατί χρησιμοποιείται. Metaphor λοιπόν είναι η αναπαράσταση γραφικώς μιας πρότασης ή μιας ιδέας με ένα αντικείμενο από την καθημερινή ζωή. Όπως δηλαδή στη δική μας εφαρμογή τα metaphors είναι εμπνευσμένα από την οδική κυκλοφορία μπορούμε να χρησιμοποιήσουμε metaphors τα οποία μπορούν να έχουν σαν θέμα για παράδειγμα τα δομικά στοιχεία ενός Η/Υ ή ενός αυτοκινήτου κ.α.. Έχοντας στο μυαλό μας την λειτουργία που επιτελεί το συγκεκριμένο metaphor στην καθημερινή ζωή μπορούμε να καταλάβουμε και την λειτουργία που επιτελεί η συγκεκριμένη ιδέα ή πρόταση στον τομέα που αναφέρεται.
Κατά την επιλογή του θέματος του metaphor δεν υπάρχουν κάποια στάνταρ κριτήρια που θα πρέπει να ακολουθηθούν. Η επιλογή του θέματος του metaphor θα κριθεί πετυχημένη αν ακόμα και ένας απλός χρήστης καταλάβει την ιδέα που αναπαριστά το συγκεκριμένο metaphor.
Έτσι και εμείς για να μπορέσουμε να κάνουμε πιο εύκολη την σύνταξη ενός XQuery ερωτήματος χρησιμοποιήσαμε τα metaphors τα οποία στην εφαρμογή μας αντικατέστησαν τα βασικά στοιχεία της XQuery. Είναι πιο εύκολο για τον χρήστη που δεν έχει ιδιαίτερες γνώσεις πάνω στην XQuery όταν θέλει να δημιουργήσει ένα ερώτημα να έχει στο μυαλό του μια εικόνα παρά να έχει μια ολόκληρη έκφραση.
Τα metaphors τα οποία χρησιμοποιήσαμε έχουν σαν θέμα την οδική κυκλοφορία. Επιλέχθηκε το συγκεκριμένο θέμα γιατί όλοι γνωρίζουμε τους κανόνες της οδικής κυκλοφορίας και έτσι μπορούμε για παράδειγμα να συνδυάσουμε μια πινακίδα του κώδικα οδικής κυκλοφορίας με έναν όρο της XQuery.

Παρακάτω αναφέρονται τα metaphors που χρησιμοποιήθηκαν καθώς και οι όροι οι οποίοι αναπαριστώνται.

- ***For***

Ο όρος for, θέτει μια επανάληψη που επιτρέπει στο υπόλοιπο FLWOR να εκτιμηθεί πολλές φορές, μια φορά για κάθε αντικείμενο στην ακολουθία που επιστρέφεται από την έκφραση μετά τη δεσμευμένη λέξη *in*. Όπως ακριβώς λειτουργεί και ο κυκλικός κόμβος σε ένα δρόμο.

(Π-65)

- ***Let***

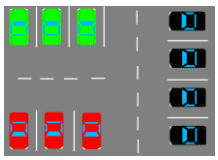
Ο όρος let είναι ένας εύκολος τρόπος να δεσμεύουμε μια μεταβλητή σε μια τιμή. Όπως μας θέτει τιμή στο όριο ταχύτητας η συγκεκριμένη πινακίδα ανάλογα με την περιοχή στην οποία κινούμαστε.

- ***Where***

Ο όρος where χρησιμοποιείται για να καθορίσει κριτήρια με τα οποία φιλτράρει τα αποτελέσματα του FLWOR. Ακριβώς σαν τον τροχονόμο που ρυθμίζει την κυκλοφορία και αποφασίζει ποια αυτοκίνητα μπορούν να κινηθούν.

- ***Return***

Ο όρος return αποτελείται από τη δεσμευμένη λέξη return ακολουθούμενη από την έκφραση που πρέπει να επιστραφεί. Όπως η έξοδος ενός αυτοκινητόδρομου η οποία μας βγάζει στον προορισμό μας.



- ***Order By***

Ο όρος order by χρησιμοποιείται για την ταξινόμηση των αποτελεσμάτων. Σαν ένα parking με παρκαρισμένα αυτοκίνητα ανάλογα με το χρώμα το οποίο έχουν.

- ***Count*** 

Η λειτουργία count χρησιμοποιείται για να καθορίσει τον αριθμό των αντικειμένων σε μια ακολουθία, έτσι και η συγκεκριμένη ηλεκτρονική πινακίδα που συναντάμε στους αυτοκινητοδρόμους μας μετράει την ταχύτητά μας και μας ενημερώνει πόσο είναι αυτή.

- ***Min*** 

Η λειτουργία min χρησιμοποιείται για να καθορίσει την ελάχιστη τιμή των αντικειμένων σε μια ακολουθία, έτσι και η συγκεκριμένη πινακίδα μας υποχρεώνει να κινούμαστε στον δρόμο με την ελάχιστη ταχύτητα των 30 χλμ.

- ***Max*** 

Η λειτουργία max χρησιμοποιείται για να καθορίσει την μέγιστη τιμή των αντικειμένων σε μια ακολουθία, έτσι και η συγκεκριμένη πινακίδα μας υποχρεώνει να κινούμαστε στον δρόμο με την μέγιστη ταχύτητα των 50 χλμ.

- ***Avg*** 

Η λειτουργία avg χρησιμοποιείται για να καθορίσει τον μέσο όρο των τιμών των αντικειμένων σε μια ακολουθία, όπως η συγκεκριμένη πινακίδα που μας ενημερώνει για τον μέσο όρο βάρους που πρέπει να υπάρχει ανά άξονα σ’ ένα φορτηγό.

- ***Sum*** 

Η λειτουργία sum χρησιμοποιείται για να καθορίσει την τελική τιμή των αντικειμένων σε μια ακολουθία. Το παγκόσμιο σύμβολο της πρόσθεσης είναι το “+” όπως δείχνει η παραπάνω πινακίδα.

- ***And***

Μια έκφραση υπόθεσης που κατασκευάστηκε με τον τελεστή “and” όπως “boolean1 and boolean2” επιστρέφει true αν και οι 2 τιμές είναι αληθείς. Έτσι και στο συγκεκριμένο σήμα η διέλευση επιτρέπεται στους πεζούς **και** στα ποδήλατα.

- ***Or***

Μια έκφραση υπόθεσης που κατασκευάστηκε με τον τελεστή “or” επιστρέφει true αν μια από τις 2 ή και οι 2 τιμές είναι αληθείς. Όπως στην συγκεκριμένη πινακίδα μπορούμε να επιλέξουμε ή τον ένα ή τον άλλο δρόμο (δεξιά ή αριστερή διαδρομή)

- ***If / Else***

Το “**if**” παίρνει μια έκφραση υπόθεσης. Αν η υπόθεση είναι αληθής, το σύστημα εκτελεί τις εντολές μέσα στο “**then**”. Αν όχι, τότε εκτελεί τις εντολές μέσα στον όρο “**else**”. Έτσι και στο συγκεκριμένο σήμα ακολουθούμε μια πορεία και μπορούμε ή να φτάσουμε στο τέλος της πορείας ή να ακολουθήσουμε μία εναλλακτική διαδρομή.

Όλα τα metaphors που παρουσιάσθηκαν περιέχονται στην εφαρμογή μας με γνώμονα πάντα να συμβάλλουν στην καλύτερη κατανόηση της XQuery καθώς και την πιο εύκολη σύνταξη και εκτέλεση των ερωτημάτων.

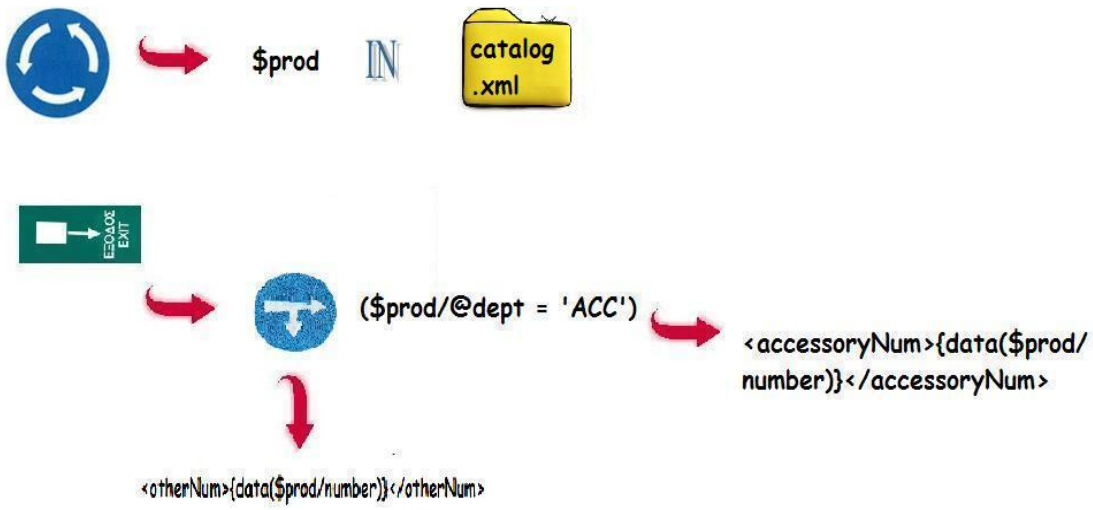
5.2 Γραφική Απεικόνιση Xquery Ερωτημάτων

Σε αυτή την παράγραφο παρουσιάζεται η γραφική απεικόνιση μερικών XQuery ερωτημάτων με την χρήση των metaphors ώστε να μας βοηθήσει να καταλάβουμε καλύτερα την λειτουργία των metaphors.

Παρακάτω παρουσιάζονται τα XQuery ερωτήματα καθώς και η γραφική τους αναπαράσταση.

Ερώτημα 1:

```
for $prod in (doc("catalog.xml")/catalog/product)
return if ($prod/@dept = 'ACC')
then <accessoryNum>{data($prod/number)}</accessoryNum>
else <otherNum>{data($prod/number)}</otherNum>
```



Εικόνα 5.1 *Ερώτημα 1*

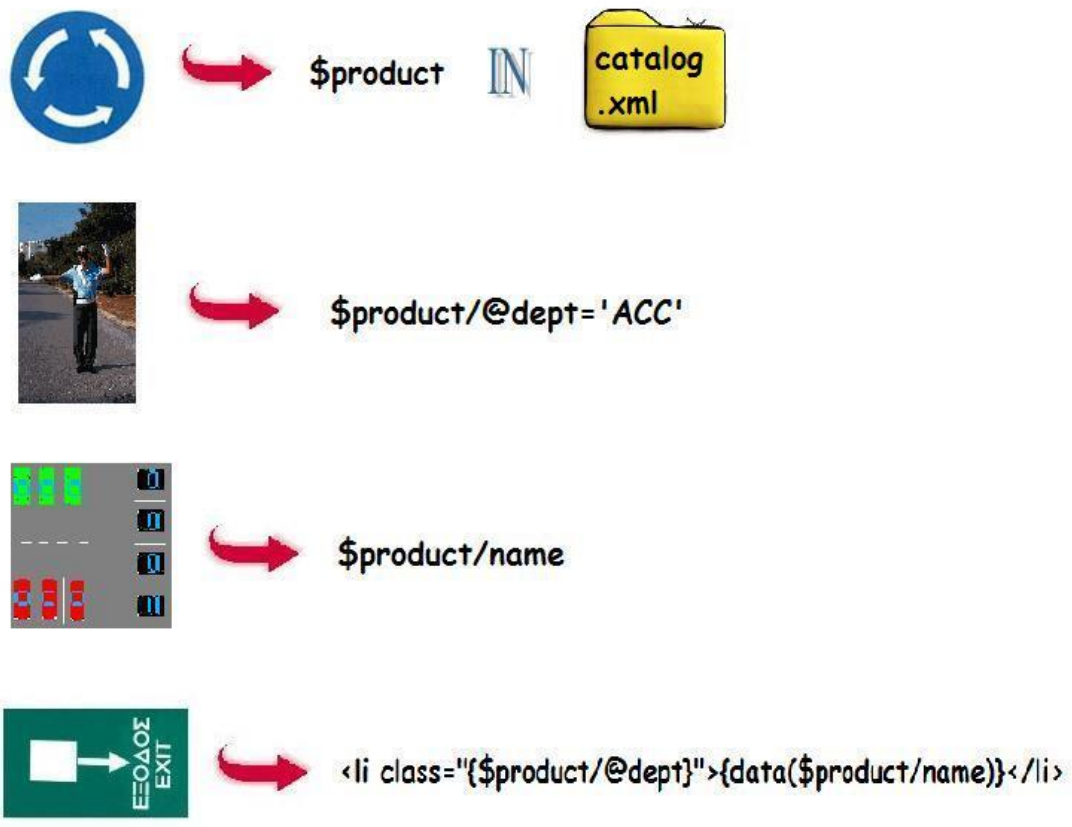
Αποτελέσματα:

```
<otherNum>557</otherNum>
<accessoryNum>563</accessoryNum>
<accessoryNum>443</accessoryNum>
<otherNum>784</otherNum>
```

Ερώτημα 2:

```
for $product in doc("catalog.xml")/catalog/product
where $product/@dept='ACC'
```

order by $product/name
return <li **class="{$product/@dept}"**>{data($product/name)}



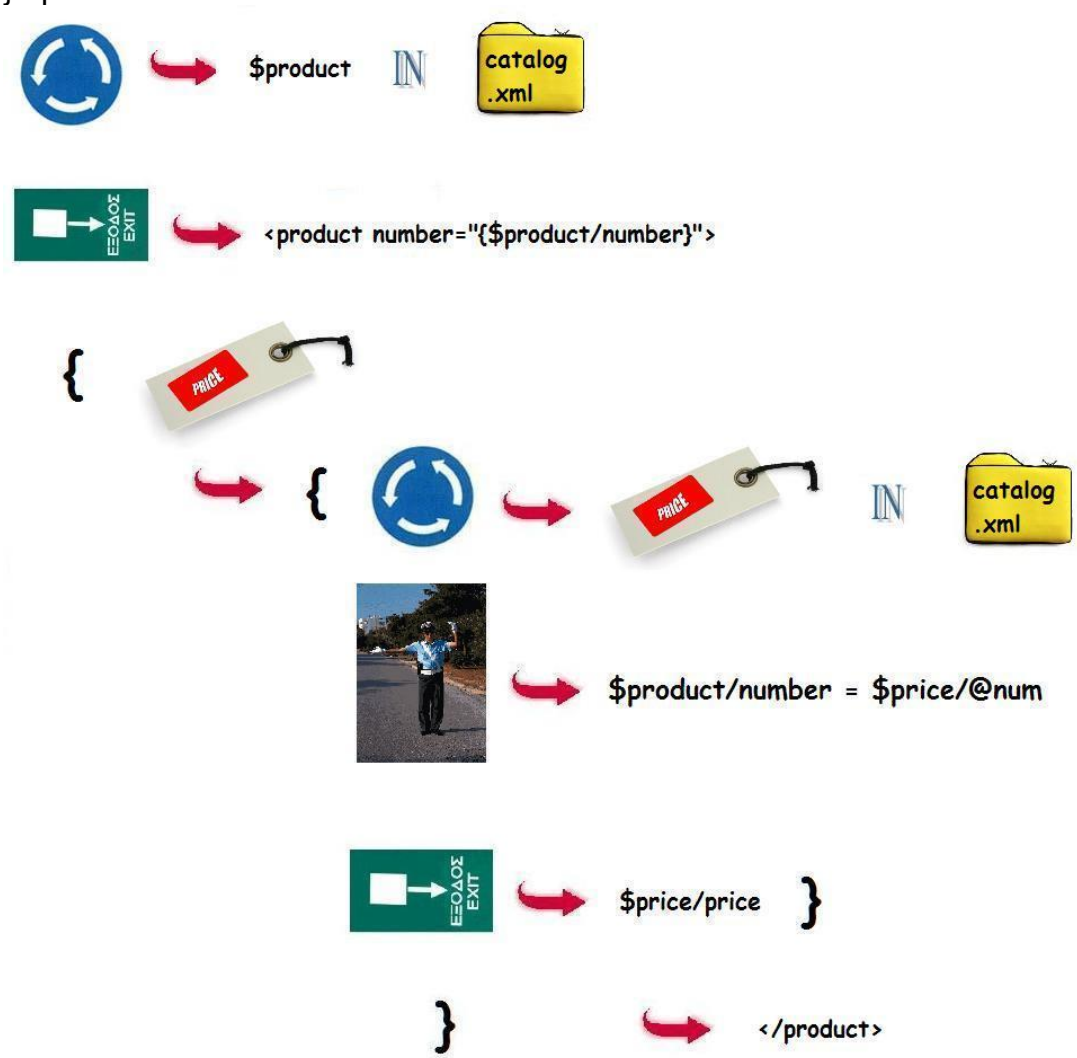
Εικόνα 5.2 *Ερώτημα 2*

Αποτελέσματα:

type="square">
<li class="ACC">Deluxe Travel Bag
<li class="ACC">Floppy Sun Hat

Ερώτημα 3:

```
for $product in doc("catalog.xml")//product
return <product number="{$product/number}">{
attribute price
{for $price in doc("prices.xml")//prices/priceList/prod
where $product/number = $price/@num
return $price/price}
}</product>
```



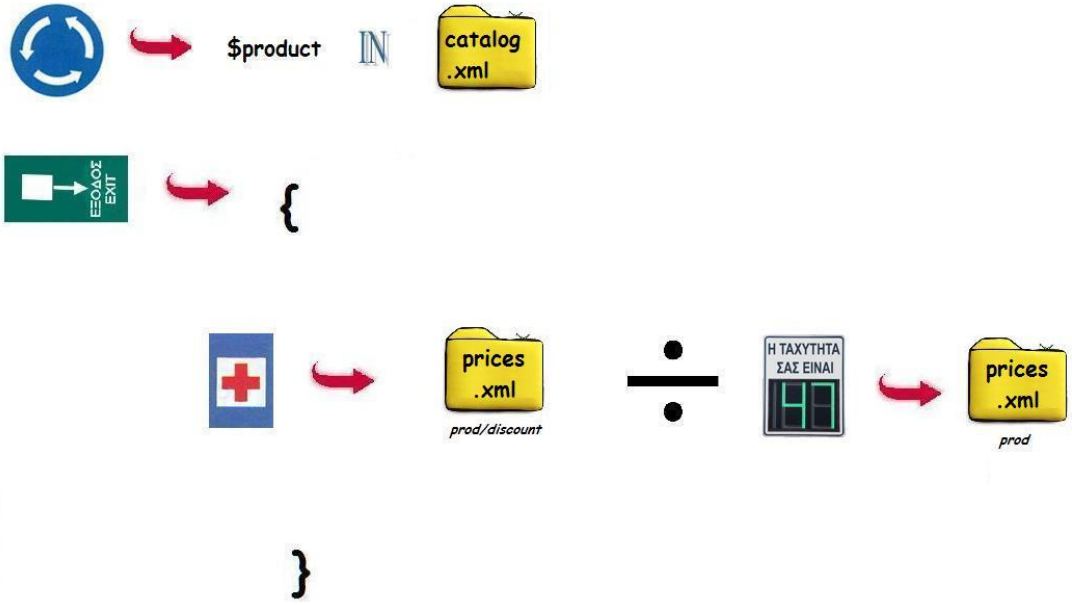
Εικόνα 5.3 *Ερώτημα 3*

Αποτελέσματα:

```
<product number="557" price="29.99"/>
<product number="563" price="69.99"/>
<product number="443" price="39.99"/>
<product number="784" price=""/>
```

Ερώτημα 4:

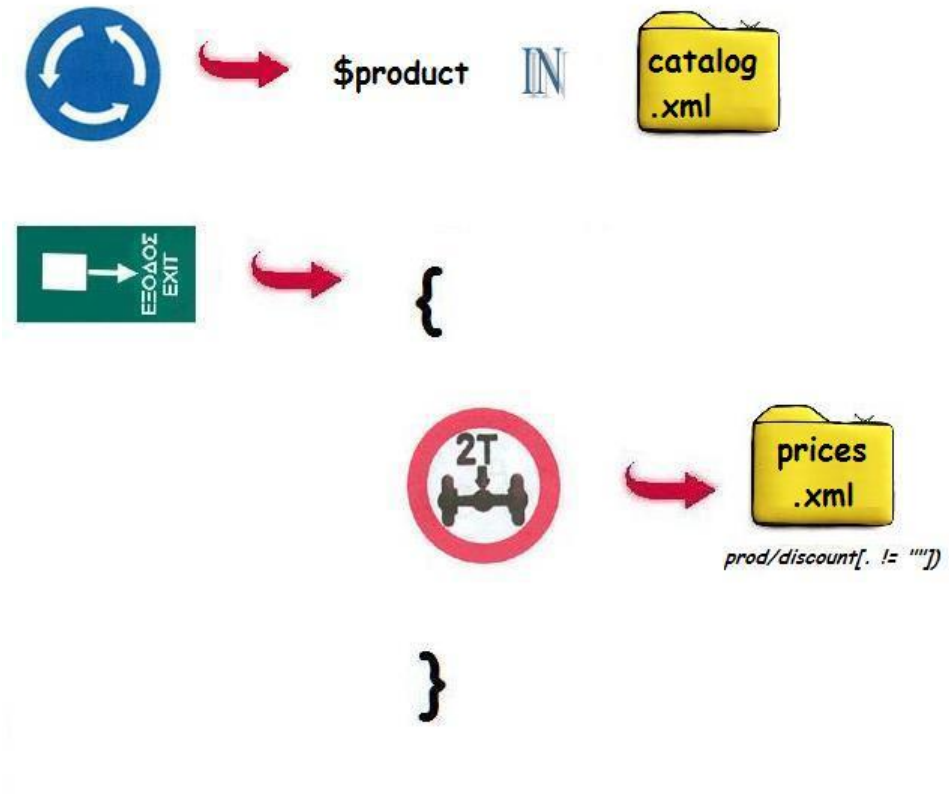
```
for $product in doc("catalog.xml")/catalog/product
return{
sum(doc("prices.xml")//prod/discount)
div count(doc("prices.xml")//prod)}
```



Εικόνα 5.4 *Ερώτημα 4*

Ερώτημα 5:

```
for $product in doc("catalog.xml")/catalog/product
return{
avg(doc("prices.xml")//prod/discount[. != ""])}
```

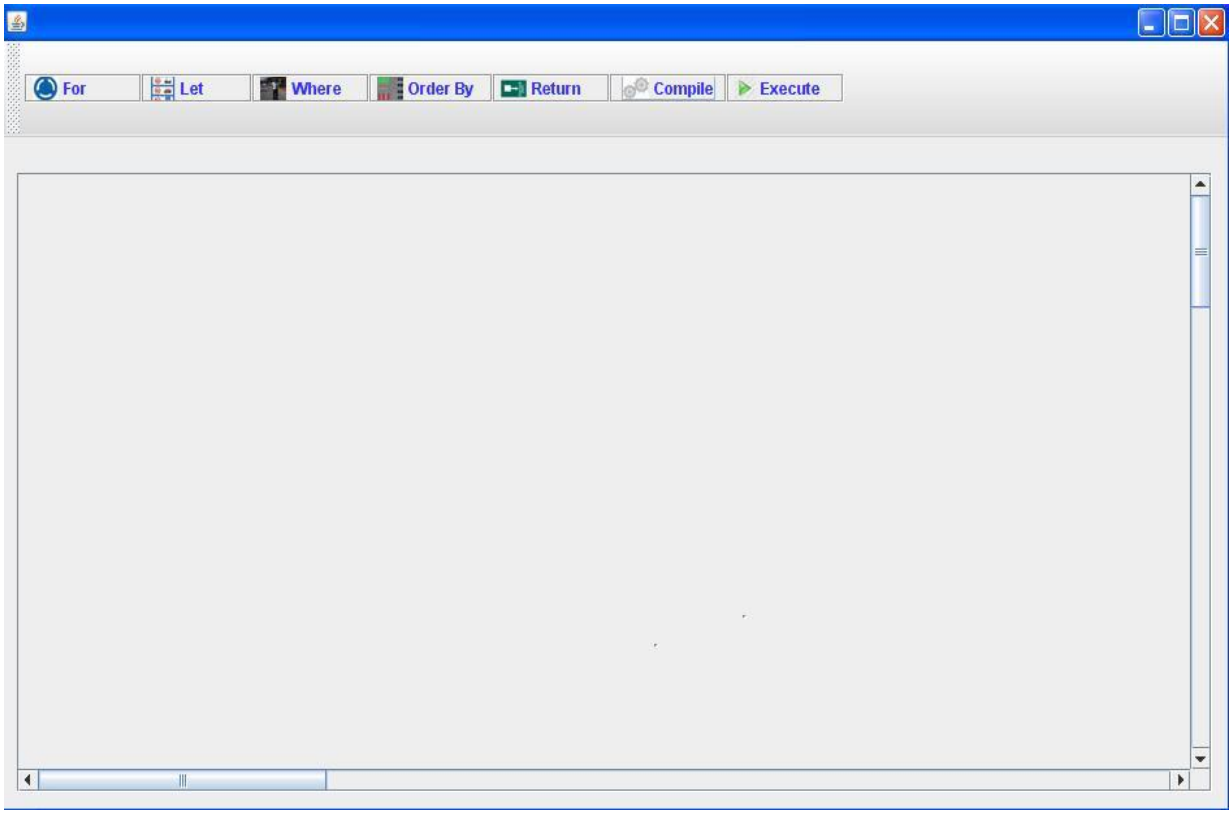


Εικόνα 5.5 *Ερώτημα 5*

5.3 Παρουσίαση του interface της εφαρμογής

Κατά την δημιουργία της εφαρμογής προσπαθήσαμε να κάνουμε το περιβάλλον όσο πιο φιλικό γίνεται για τον χρήστη, παρέχοντάς του ταυτόχρονα κάθε δυνατή βοήθεια ώστε να γίνει στο μέγιστο δυνατό βαθμό εύκολη η σύνταξη και η εκτέλεση των ερωτημάτων.
Για τον σκοπό αυτό χρησιμοποιήθηκαν ένας αριθμός κουμπιών ώστε να γίνει πιο αυτοματοποιημένη η σύνταξη του ερωτήματος και να απαιτείται από τον χρήστη να γράψει όσο το δυνατόν λιγότερο. Επίσης για να κάνουμε ακόμα πιο φιλικό το περιβάλλον χρησιμοποιήσαμε επεξηγήσεις στο τι πρέπει να κάνει σε κάθε σημείο της εφαρμογής ο χρήστης έτσι ώστε να μην φτάσει σε σύγχυση από το πλήθος των λειτουργιών που του προσφέρονται.
Επίσης για να απλουστεύσουμε την διαδικασία για τον χρήστη, στην εφαρμογή μας οι λειτουργίες προσφέρονται στον χρήστη σταδιακά ώστε βήμα – βήμα να φτάσει στη σύνταξη του ερωτήματος. Θεωρήσαμε ότι είναι πολύ δύσκολο για έναν χρήστη ο οποίος δεν είναι ιδιαίτερα εξοικειωμένος με την XQuery να του δοθούν από την αρχή όλα τα εργαλεία της εφαρμογής μας.

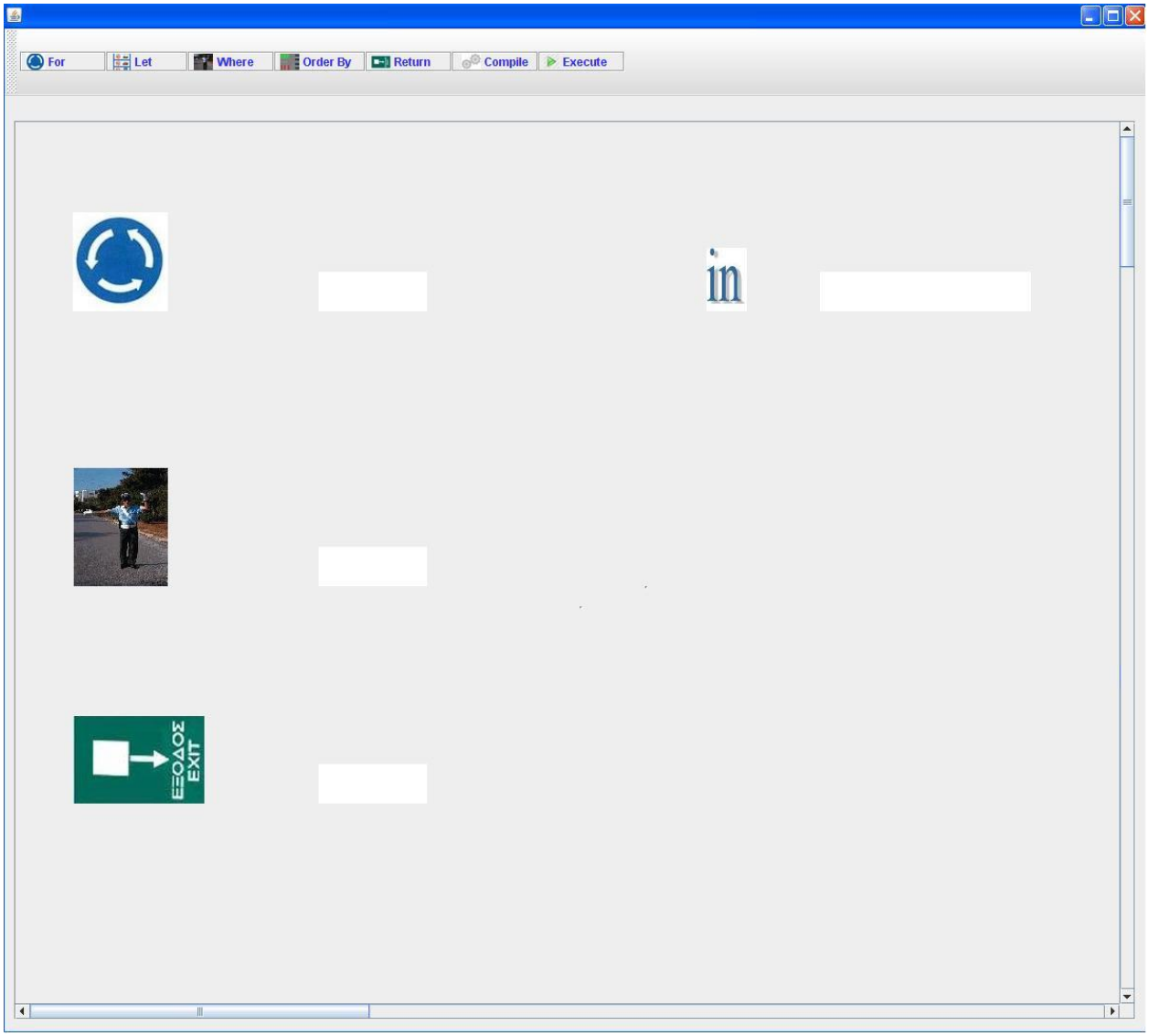
Στην εικόνα 5.6 που ακολουθεί φαίνεται το παράθυρο υποδοχής της εφαρμογής μας



Εικόνα 5.6 *Παράθυρο υποδοχής της εφαρμογής*

Όπως μπορούμε να παρατηρήσουμε μόλις εκτελείται η εφαρμογή μας χωρίζεται σε δύο κομμάτια. Σε αυτό που βρίσκεται στην κορυφή δίνεται στον χρήστη ένα toolbar στο οποίο περιέχονται εφτά κουμπιά τα οποία είναι και τα βασικά της εφαρμογής μας. Αυτό γίνεται γιατί όπως αναφέραμε πριν θέλουμε ο χρήστης να συντάξει σταδιακά το ερώτημα του. Ενώ στο δεύτερο κομμάτι συντίθεται ουσιαστικά το ερώτημά μας.
Στο toolbar αυτό υπάρχουν πέντε κουμπιά τα οποία είναι το for, το let, το where, το order by και το return τα οποία είναι τα βασικά στοιχεία για την σύνταξη ενός XQuery ερωτήματος. Πατώντας ένα από αυτά τα πέντε κουμπιά, προστίθεται και στο ερώτημά μας ο αντίστοιχος όρος.
Αν για παράδειγμα ο χρήστης στο ερώτημά του θέλει να έχει τους όρους for, where και return θα πατήσει στα αντίστοιχα κουμπιά. Έτσι θα σχηματιστεί ο σκελετός του ερωτήματός μας.

Για να φανεί τι ακριβώς εννοούμε παραθέτουμε την παρακάτω εικόνα



Εικόνα 5.7

Όταν πατάμε σε ένα κουμπί από το toolbar στην εφαρμογή μας εμφανίζεται το αντίστοιχο metaphor και ενεργοποιούνται και οι αντίστοιχες λειτουργίες για την συγκεκριμένη πρόταση. Για παράδειγμα στην πρόταση που περιέχεται το metaform for ενεργοποιούνται 2 text area καθώς και εμφανίζεται μια εικόνα που αναπαριστά την λέξη “in”. Οι εικόνες που εμφανίζονται σε κάθε πρόταση μας δείχνουν κάτι το οποίο είναι απαραίτητο για το ερώτημά μας και θα υπάρχει στο ερώτημα χωρίς να απαιτείται από τον χρήστη να κάνει τίποτα.

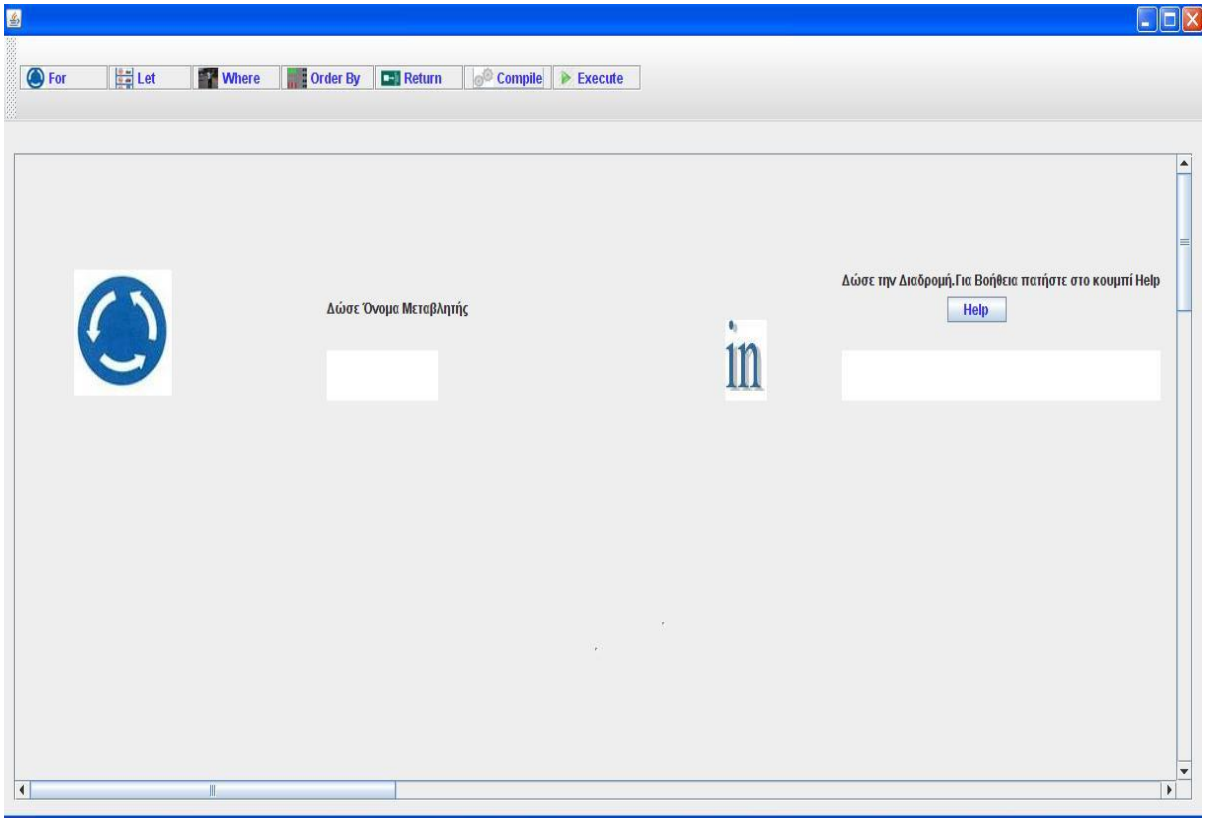
Για να επανέλθουμε στο αρχικό μας toolbar όπως θα παρατηρήσατε υπάρχουν δύο επιπλέον κουμπιά στα οποία δεν αναφερθήκαμε, το κουμπί compile και το κουμπί execute.
Το κουμπί compile ουσιαστικά δημιουργεί το ερώτημα σε ένα text αρχείο και το εμφανίζει στον χρήστη. Έτσι ο χρήστης έχει την δυνατότητα να δει ολοκληρωμένο το ερώτημά του καθώς και να αποθηκεύσει το ερώτημά του.

Το κουμπί execute είναι αυτό που ουσιαστικά στέλνει την εντολή στην db2 για να τρέξει το ερώτημα του χρήστη και εμφανίζει τα αποτελέσματα σε ένα xml αρχείο.
Σε κάθε κουμπί του toolbar όπως και σε όλα τα κουμπιά που χρησιμοποιούνται στην εφαρμογή μας εκτός από το όνομά του υπάρχει η αντίστοιχη εικόνα που το χαρακτηρίζει έτσι ώστε ο χρήστης να κατανοήσει καλύτερα την λειτουργία του συγκεκριμένου κουμπιού καθώς και να μείνει στη μνήμη του, στο βαθμό που είναι εφικτό.
Εκτός από το toolbar όπως αναφέραμε υπάρχει και το κομμάτι που δημιουργείται το ερώτημα μας και όπως είδαμε και με το παράδειγμα του όρου for ενεργοποιούνται οι λειτουργίες της κάθε πρότασης.
Αφού παρουσιάσαμε τα κύρια στοιχεία της εφαρμογής μας θα ξεκινήσουμε να αναφέρουμε με περισσότερες λεπτομέρειες τις λειτουργίες που προσφέρονται στον χρήστη σε κάθε πρόταση που περιέχει έναν συγκεκριμένο όρο. Για την καλύτερη κατανόηση θα αντιπαραθέσουμε εικόνες για κάθε έναν όρο.

5.3.1. Πρόταση For

Για τον όρο for μόλις πατήσουμε στο αντίστοιχο κουμπί του toolbar εμφανίζεται το αντίστοιχο metaphor και ενεργοποιούνται δύο text area όπως και το σύμβολο in το οποίο είναι και απαραίτητο για το ερώτημά μας.
Θεωρήσαμε πολύ δύσκολο για τον χρήστη να κατανοήσει τι πρέπει να γράψει στο κάθε text area χωρίς να υπάρχει η αντίστοιχη βοήθεια. Για τον λόγο αυτό ο χρήστης μόλις κάνει roll over με το ποντίκι πάνω στο text area αμέσως εμφανίζονται πάνω από το κάθε text area αντίστοιχη βοήθεια με το τι πρέπει να συμπληρώσει. Μόλις ο χρήστης συμπληρώσει το text area η βοήθεια εξαφανίζεται γιατί πλέον δεν είναι χρήσιμη για τον χρήστη ολοκληρώνοντας τον σκοπό της ύπαρξής της.

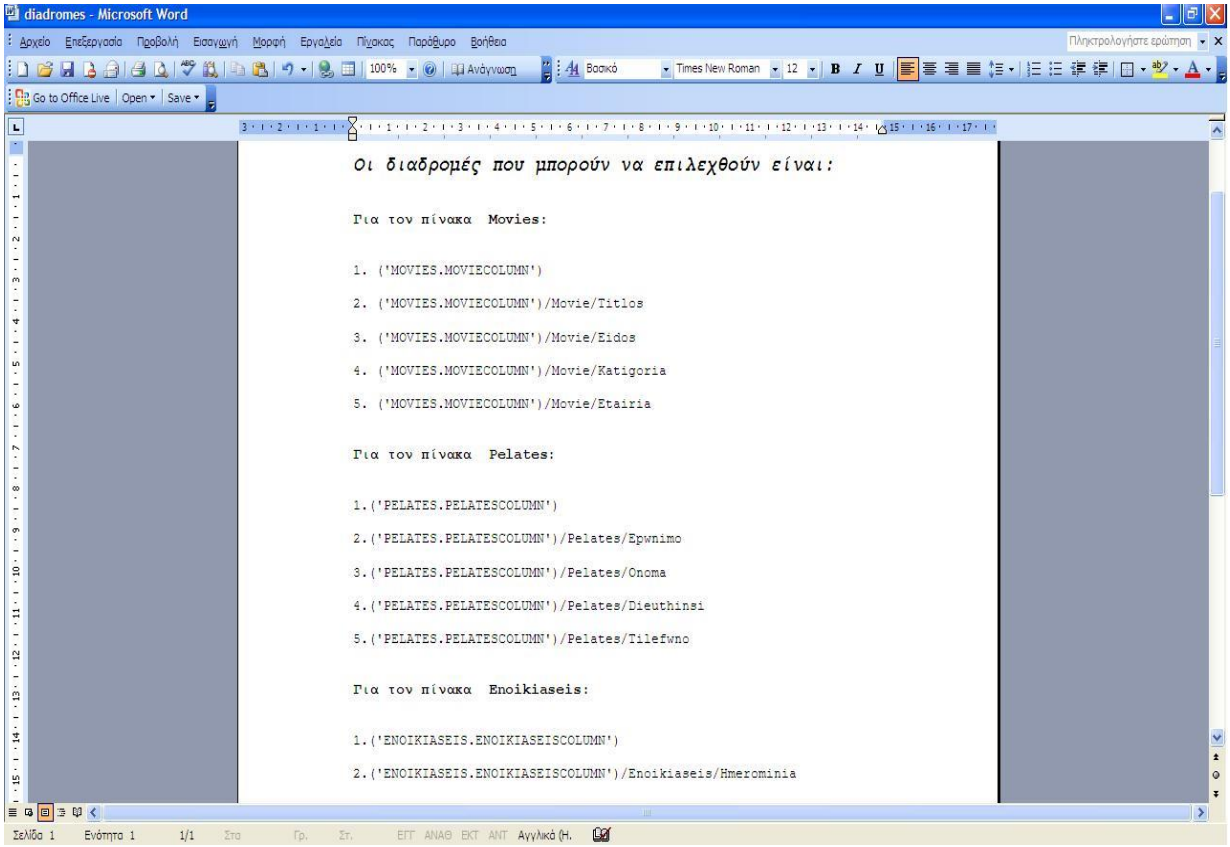
Στην επόμενη εικόνα φαίνεται η πρόταση for



Εικόνα 5.8 *Πρόταση for*

Στο πρώτο text area το μόνο που έχει να κάνει ο χρήστης όπως αναφέρει και η σχετική βοήθεια είναι να δώσει το όνομα της μεταβλητής που θέλει να χρησιμοποιήσει στο ερώτημά του. Σε περίπτωση που δεν θέλει να χρησιμοποιήσει μεταβλητή ο χρήστης μπορεί να αφήσει κενό το συγκεκριμένο text area και με αυτό τον τρόπο το συγκεκριμένο text area δεν συμμετέχει στο ερώτημα.
Όπως μπορούμε να δούμε στην προηγούμενη εικόνα για το δεύτερο text area εκτός από την επεξήγηση του τι πρέπει να συμπληρώσει ο χρήστης στο συγκεκριμένο πεδίο υπάρχει και το κουμπί help. Όταν ο χρήστης πατήσει πάνω στο κουμπί help ανοίγει το αρχείο diadromes.doc στο οποίο περιέχονται όλες οι δυνατές διαδρομές που μπορεί να επιλέξει ο χρήστης για κάθε πίνακα. Ο χρήστης το μόνο που έχει να κάνει είναι να επιλέξει την διαδρομή που θέλει να χρησιμοποιήσει και να την κάνει αντιγραφή – επικόλληση στο δεύτερο text area.
Για την εικόνα in όπως αναφέρθηκε και πριν είναι απαραίτητο στοιχείο του ερωτήματός μας που δεν μπορεί να παραληφθεί και για αυτό τον λόγο δεν μπορεί να τροποποιηθεί.

Στην επόμενη εικόνα φαίνεται το περιεχόμενο του diadromes.doc

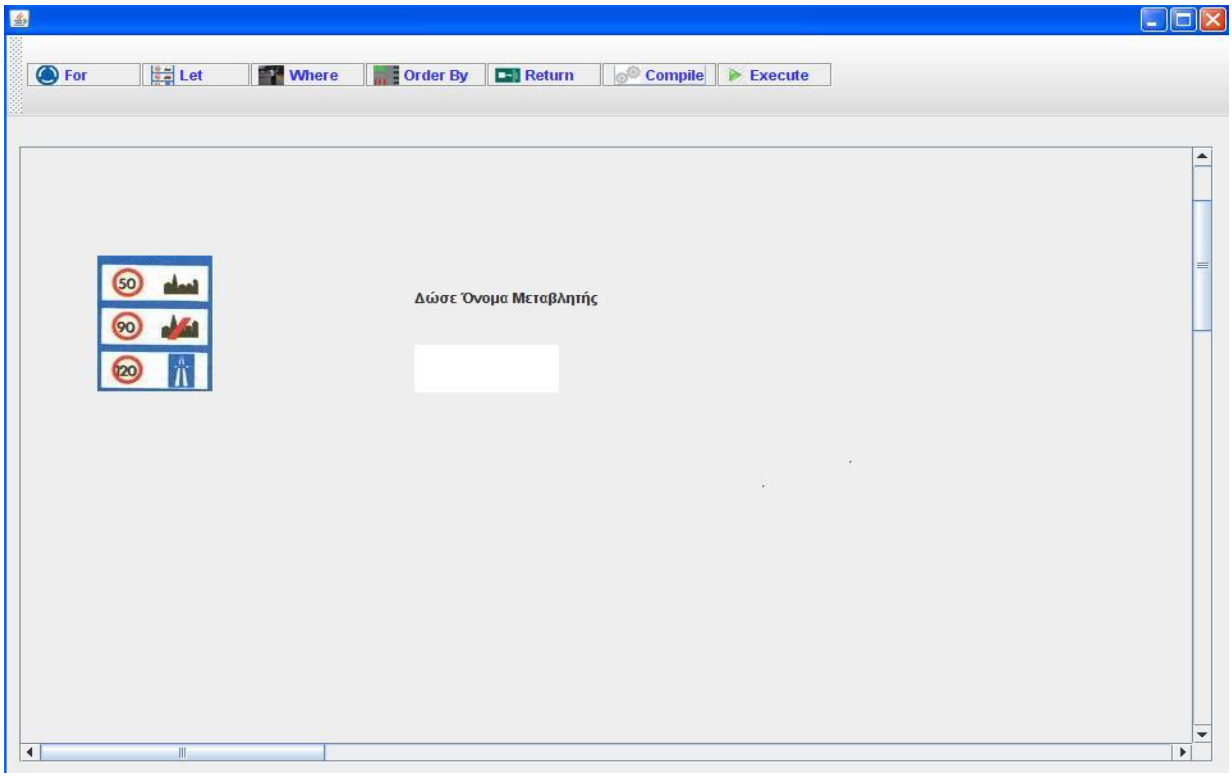


Εικόνα 5.9 *diadromes.doc*

5.3.2. Πρόταση Let

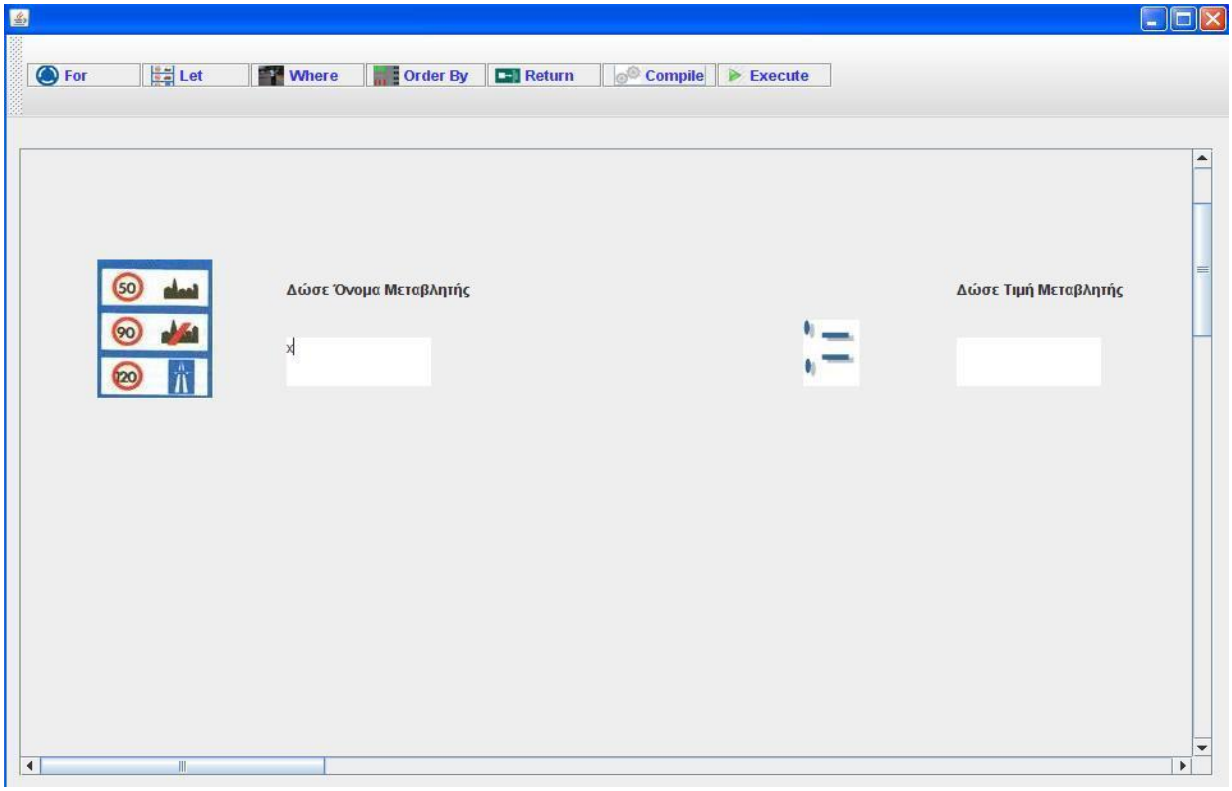
Για τον όρο let μόλις πατήσουμε το αντίστοιχο κουμπί από το toolbar εμφανίζεται το αντίστοιχο metaphor και ενεργοποιείται ένα text area στο οποίο η βοήθεια που αναφέρεται σε αυτό ζητάει από τον χρήστη να δώσει την τιμή της μεταβλητής στην οποία θα προσδώσει ένα χαρακτηριστικό.
Μόλις ο χρήστης δώσει την τιμή της μεταβλητής εμφανίζονται στην εφαρμογή μας η εικόνα του συμβόλου := το οποίο είναι απαραίτητο για το ερώτημά μας για να είναι δυνατή η εκχώρηση του χαρακτηριστικού στην μεταβλητή όπως επίσης εμφανίζεται ένα δεύτερο text area στο οποίο η βοήθεια μας ενημερώνει ότι σε αυτό το text area πρέπει να δοθεί το χαρακτηριστικό το οποίο θα εκχωρηθεί στην μεταβλητή που γράψαμε στο πρώτο text area.
Για να γίνουν πιο κατανοητά αυτά που αναφέραμε προηγουμένως θα ακολουθήσουν δύο εικόνες στις οποίες θα φαίνεται πώς δημιουργείται η πρόταση let.

Μόλις πατήσουμε το κουμπί let του toolbar εμφανίζεται το εξής:



Εικόνα 5.10 *Πρόταση let*

Αφού συμπληρώσουμε το πρώτο text area με το όνομα της μεταβλητής που επιθυμούμε εμφανίζεται το δεύτερο text area:



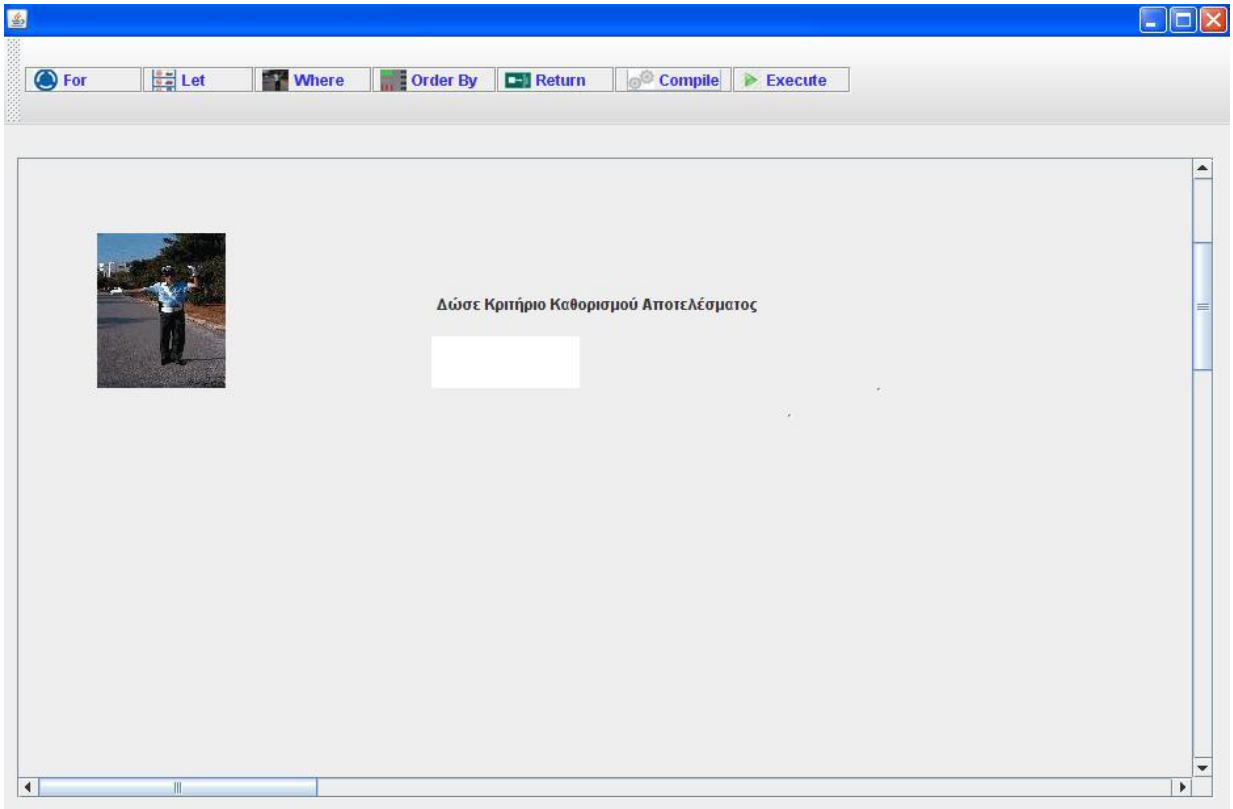
Εικόνα 5.11

5.3.3 Πρόταση Where

Για τον όρο where μόλις πατήσουμε στο αντίστοιχο κουμπί του toolbar εμφανίζεται στην εφαρμογή μας το metaphor του where και ένα text area όπου η βοήθεια μας ενημερώνει ότι πρέπει να δώσουμε το κριτήριο καθορισμού του αποτελέσματος. Εφόσον δώσουμε το κριτήριο στην εφαρμογή μας εμφανίζεται το σύμβολο του $ πριν το πρώτο text area, το οποίο όπως αναφέραμε και για τα προηγούμενα σύμβολα των άλλων προτάσεων είναι απαραίτητα και δεν μπορούν να απαλειφτούν από τις προτάσεις. Τα σύμβολα εμφανίζονται για να μάθει ο χρήστης το πώς συντάσσεται σωστά ένα ερώτημα ώστε να εξοικειωθεί και δεν τροποποιούνται ώστε στο να μειωθεί το ενδεχόμενο ο χρήστης να αμελήσει να γράψει στο ερώτημα του το συγκεκριμένο σύμβολο, γεγονός το οποίο θα προκαλούσε σφάλμα στο ερώτημά μας.
Επίσης όπως θα δούμε και στις εικόνες που θα ακολουθήσουν στην συγκεκριμένη πρόταση εμφανίζεται ένα βοηθητικό toolbar στο οποίο περιέχονται όλοι τελεστές συγκρίσεως όπως είναι ο =,<,!= κ.α.. Έτσι ο χρήστης

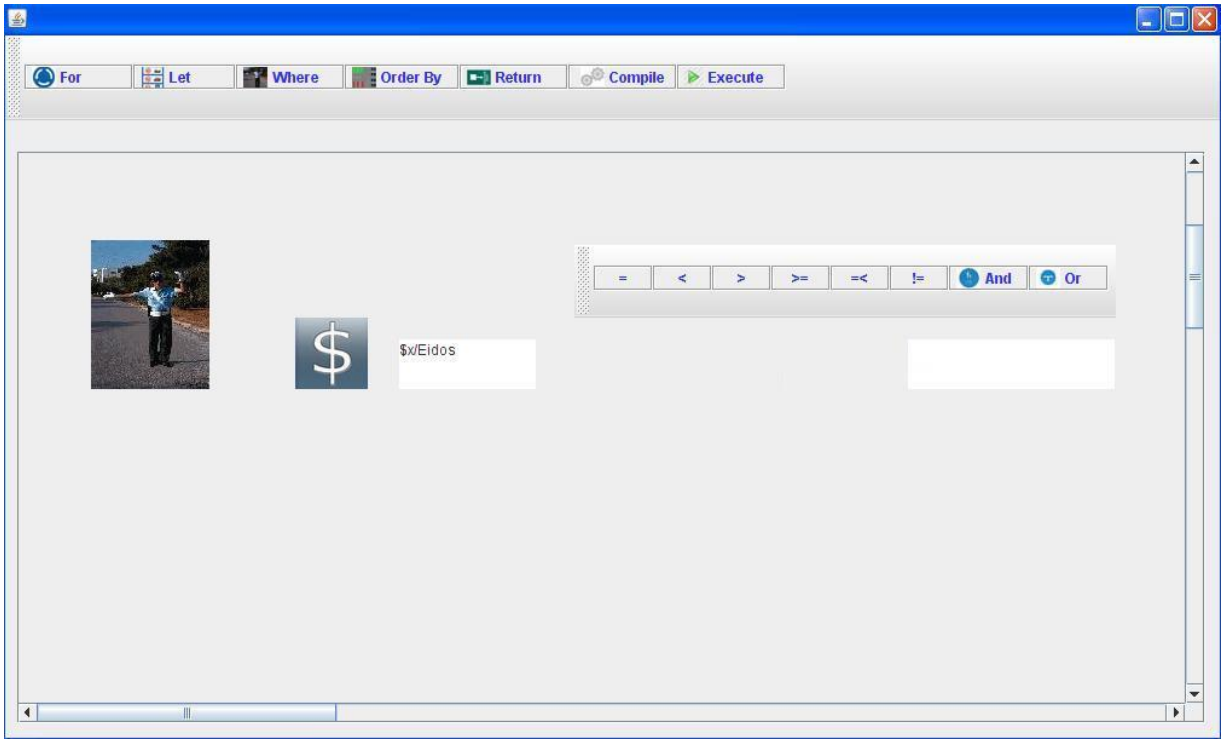
έχει την δυνατότητα να επιλέξει όποιον τελεστή επιθυμεί για να καθορίσει το κριτήριο του.
Στην πρόταση where ενεργοποιείται ένα δεύτερο text area στο οποίο όπως μας πληροφορεί η σχετική βοήθεια πρέπει να συμπληρώσουμε την τιμή του κριτηρίου.
Τέλος στην πρόταση where ο χρήστης έχει την δυνατότητα να συντάξει ένα σύνθετο κριτήριο το οποίο θα περιλαμβάνει δύο σκέλη χρησιμοποιώντας τον όρο and ή τον όρο or. Αυτή η δυνατότητα βρίσκεται στο toolbar της συγκεκριμένης πρότασης όπου μαζί με τους τελεστές συγκρίσεως που αναφέραμε πριν υπάρχουν και τα αντίστοιχα κουμπιά που οδηγούν στους όρους and και or.
Για να γίνουν πιο κατανοητά αυτά που αναφέραμε ακολουθούν τρεις εικόνες που δείχνουν τα βήματα της δημιουργίας της πρότασης where.

Πατώντας στο κουμπί where εμφανίζεται:



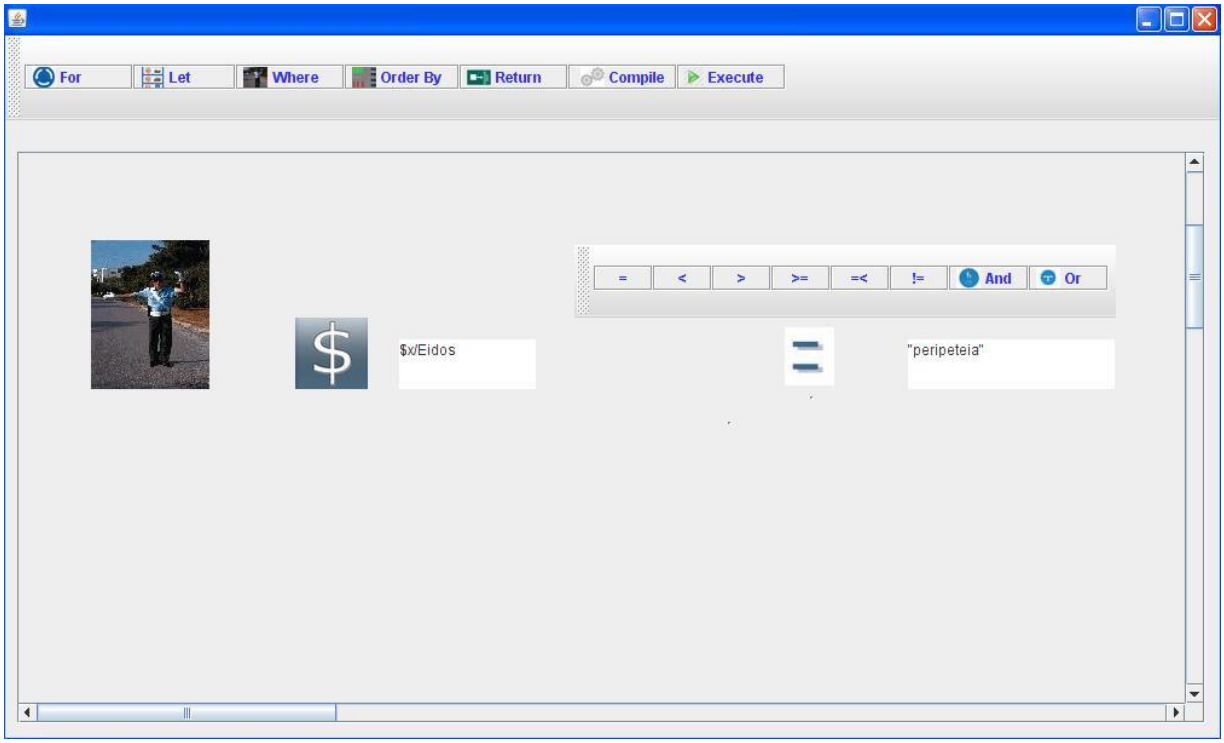
Εικόνα 5.12 *Πρόταση where*

Αφού δώσουμε το κριτήριο εμφανίζεται:



Εικόνα 5.13

Ολοκληρώνοντας την διαδικασία η πρόταση where τώρα:

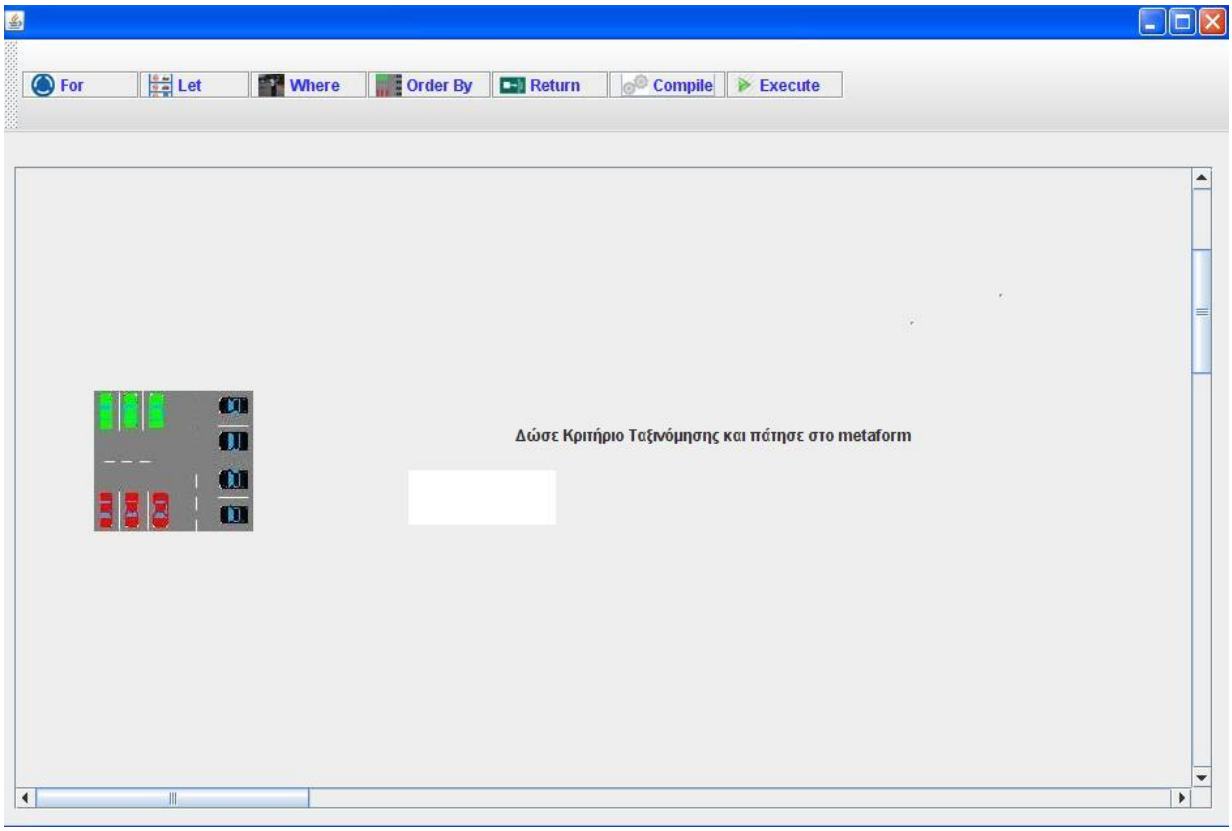


Εικόνα 5.14

5.3.4 Πρόταση Order By

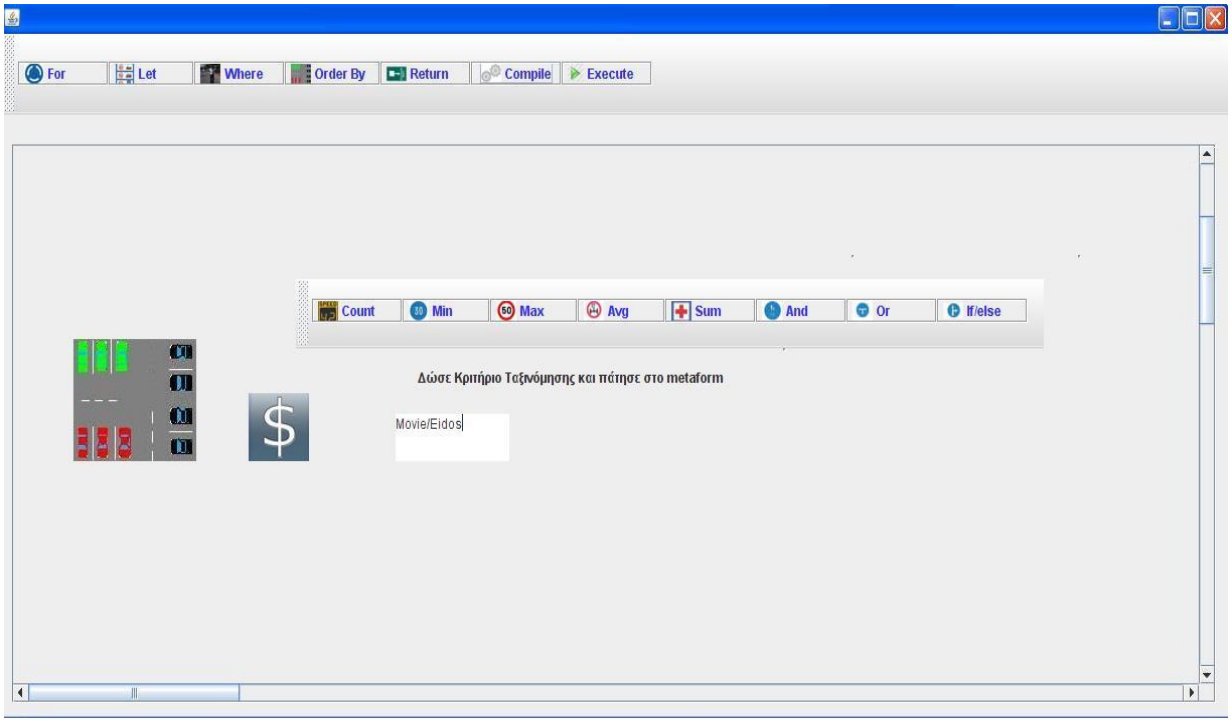
Για τον όρο order by μόλις πατήσουμε στο αντίστοιχο κουμπί εμφανίζεται το metaphor order by και ένα text area που η βοήθειά του μας πληροφορεί ότι πρέπει να πληκτρολογήσουμε το κριτήριο ταξινόμησης και ακολούθως να κάνουμε κλικ στην εικόνα του metaphor.
Κάνοντας κλικ πάνω στην εικόνα του metaphor βλέπουμε ότι ενεργοποιείται μια βοηθητική toolbar στην οποία υπάρχουν κουμπιά ένα για κάθε τελεστή που μπορεί να χρησιμοποιηθούν για την ταξινόμηση. Αυτοί οι τελεστές είναι οι:
count, min, max, avg, sum, and, or και if – else. Ο χρήστης μπορεί να επιλέξει ανάμεσα στους τελεστές που αναφέρθηκαν όποιον τελεστή επιθυμεί. Στην περίπτωση των τελεστών and, or και if – else για να μπορεί να είναι ολοκληρωμένη η συνθήκη ενεργοποιείται ένα δεύτερο text area στο οποίο θα δοθεί το δεύτερο κριτήριο.
Για να γίνουν πιο κατανοητά αυτά που αναφέραμε στις παρακάτω εικόνες που θα παραθέσουμε θα φαίνονται τα βήματα που περιγράψαμε.

Πατώντας στο κουμπί order by εμφανίζεται:



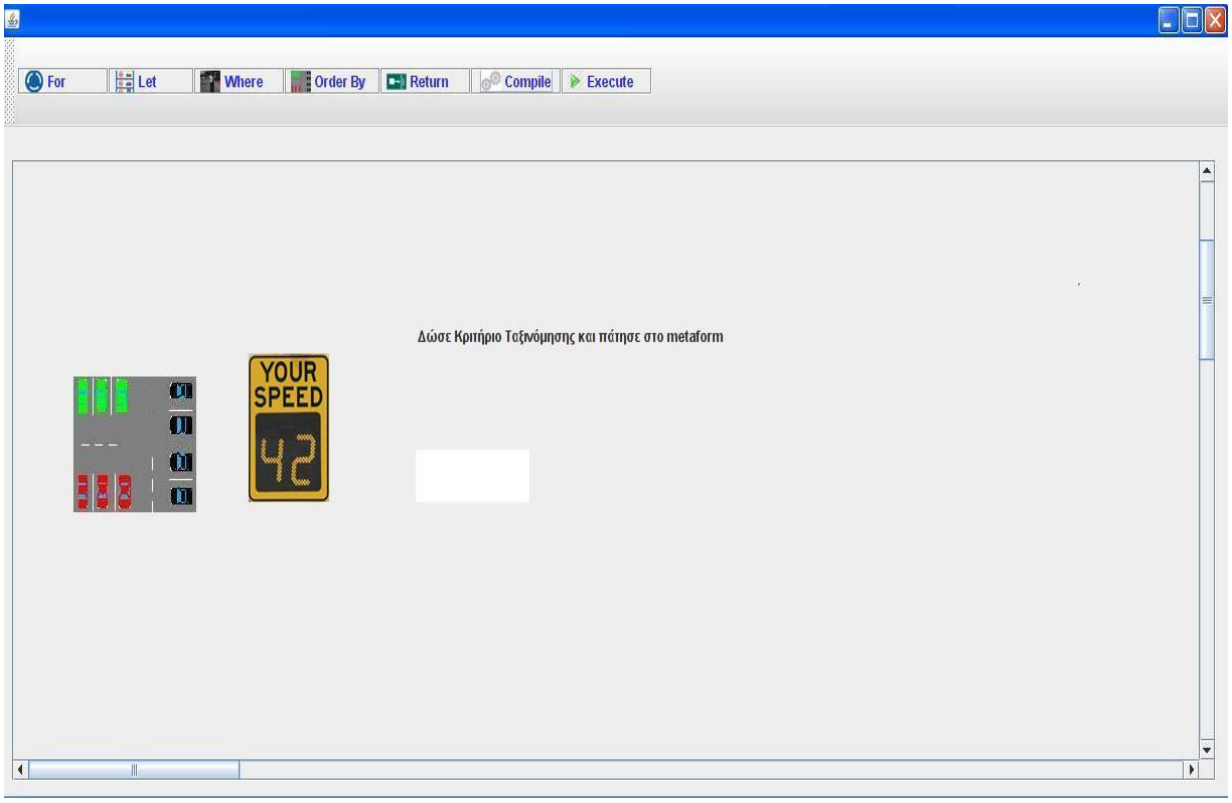
Εικόνα 5.15 *Πρόταση order by*

Αφού δώσουμε το κριτήριο ταξινόμησης και πατήσουμε στο metaphor:



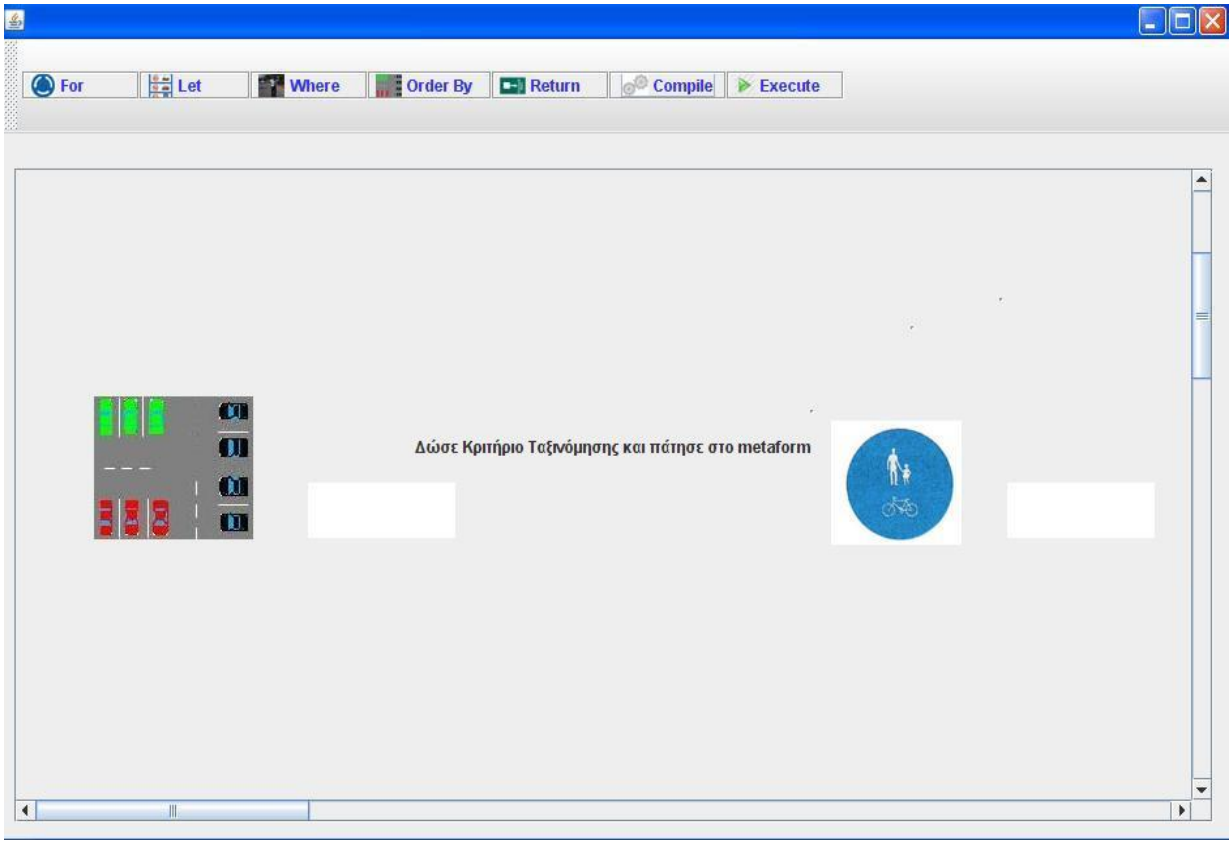
Εικόνα 5.16

Παράδειγμα πρότασης order by με τον τελεστή count:



Εικόνα 5.17

Παράδειγμα πρότασης order by με τον τελεστή and:



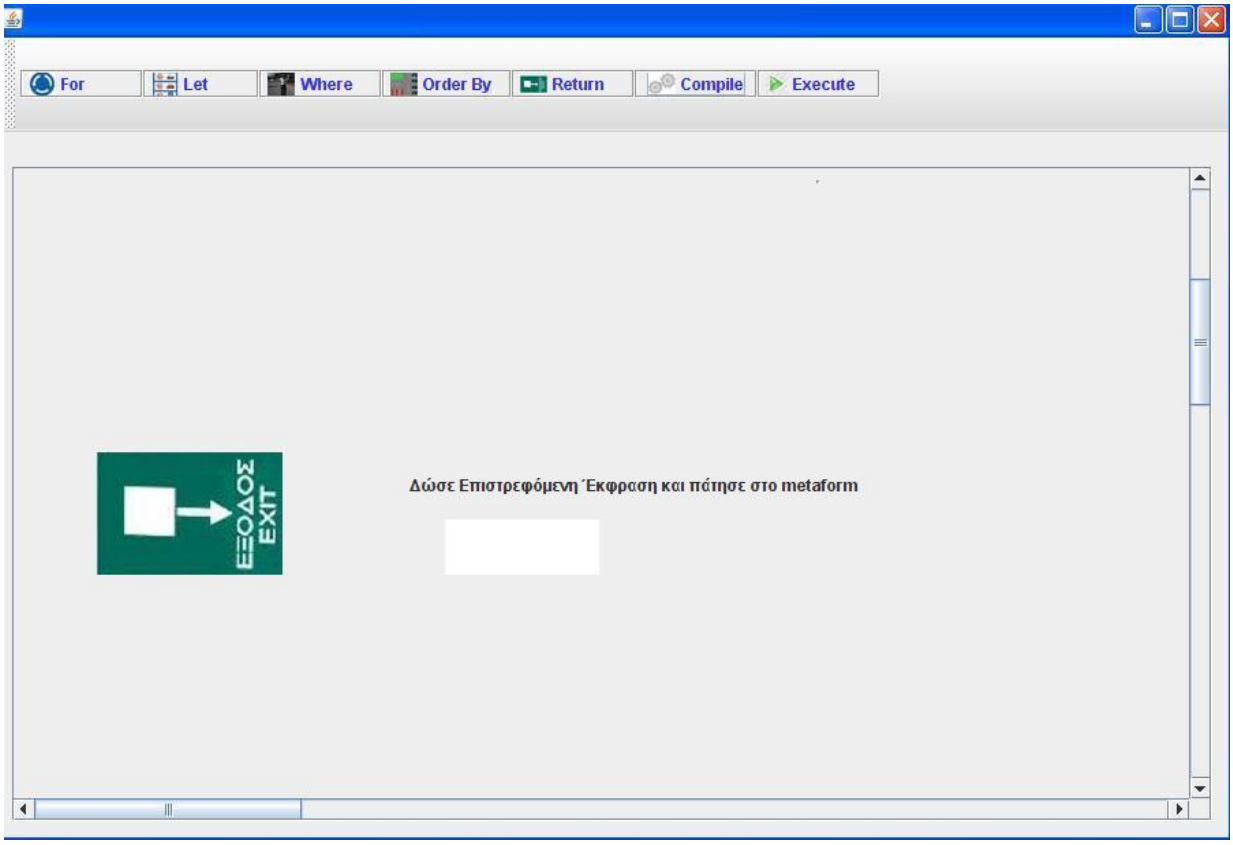
Εικόνα 5.18

5.3.5 Πρόταση Return

Για τον όρο return ισχύουν σχεδόν τα ίδια πράγματα που ισχύουν και στο order by. Η λογική μας ήταν να υπάρχει όσο το δυνατόν η ίδια μεθοδολογία για την σύνταξη μιας πρότασης ενός ερωτήματος έτσι ώστε ο χρήστης να εξοικειωθεί με την εφαρμογή μας. Για να επανέλθουμε στην πρόταση return, ο χρήστης μόλις πατήσει στο κουμπί return από το αρχικό toolbar στο κεντρικό πλαίσιο ενεργοποιείται το metaphor return και ένα text area στο οποίο όπως μας πληροφορεί η αντίστοιχη βοήθεια του text area θα πρέπει να δώσουμε την επιστρεφόμενη έκφραση. Αφού πληκτρολογήσουμε την επιστρεφόμενη έκφραση η βοήθεια του text area μας πληροφορεί ότι πρέπει να κάνουμε κλικ στο εικονίδιο του metaphor . Κάνοντας κλικ στο εικονίδιο του metaphor ενεργοποιείται μια βοηθητική toolbar στην οποία περιέχονται τα ίδια ακριβώς κουμπιά με την βοηθητική toolbar που υπήρχε και την πρόταση order by. Δηλαδή υπάρχουν τα κουμπιά: count, min, max, avg, sum, and, or και if – else τα οποία αντιστοιχούν και στους αντίστοιχους τελεστές που μπορούν να χρησιμοποιηθούν στην συγκεκριμένη πρόταση. Για τους τελεστές and, or και if – else ενεργοποιείται ένα δεύτερο text area το οποίο θα ολοκληρώσει την συγκεκριμένη πρόταση. Αξίζει να σημειωθεί πως σε περίπτωση που ο χρήστης μετανιώσει και θέλει να επιλέξει έναν άλλο τελεστή από αυτόν που αρχικά επέλεξε έχει την δυνατότητα να κάνει κλικ ξανά στο metaphor και έτσι ενεργοποιείται ξανά το βοηθητικό toolbar με τα κουμπιά των αντίστοιχων τελεστών ώστε να επιλέξει ξανά τον τελεστή που επιθυμεί.

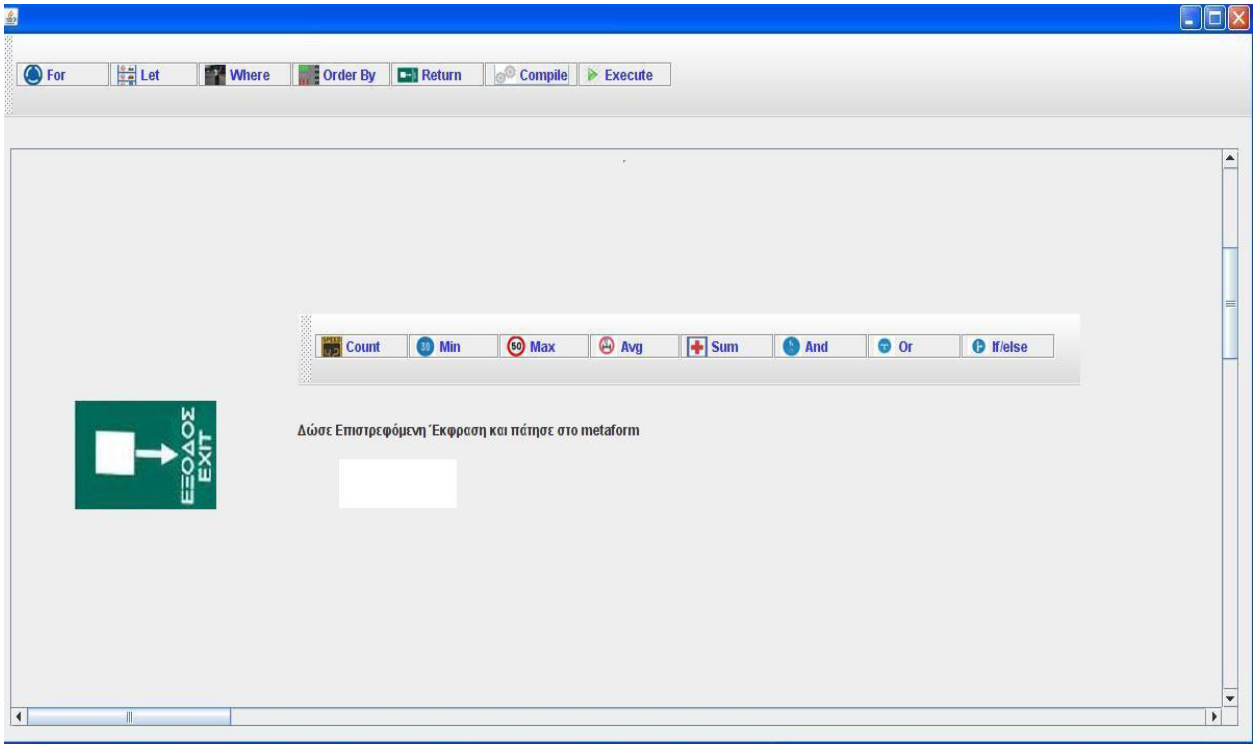
Ολοκληρώνοντας την αναφορά μας στην πρόταση return ακολουθούν εικονικά τα βήματα σύνταξης της πρότασης return

Πατώντας στο κουμπί return εμφανίζεται:



Εικόνα 5.19 Πρόταση return

Αφού δώσουμε την επιστρεφόμενη έκφραση και πατήσουμε στο metaphor:

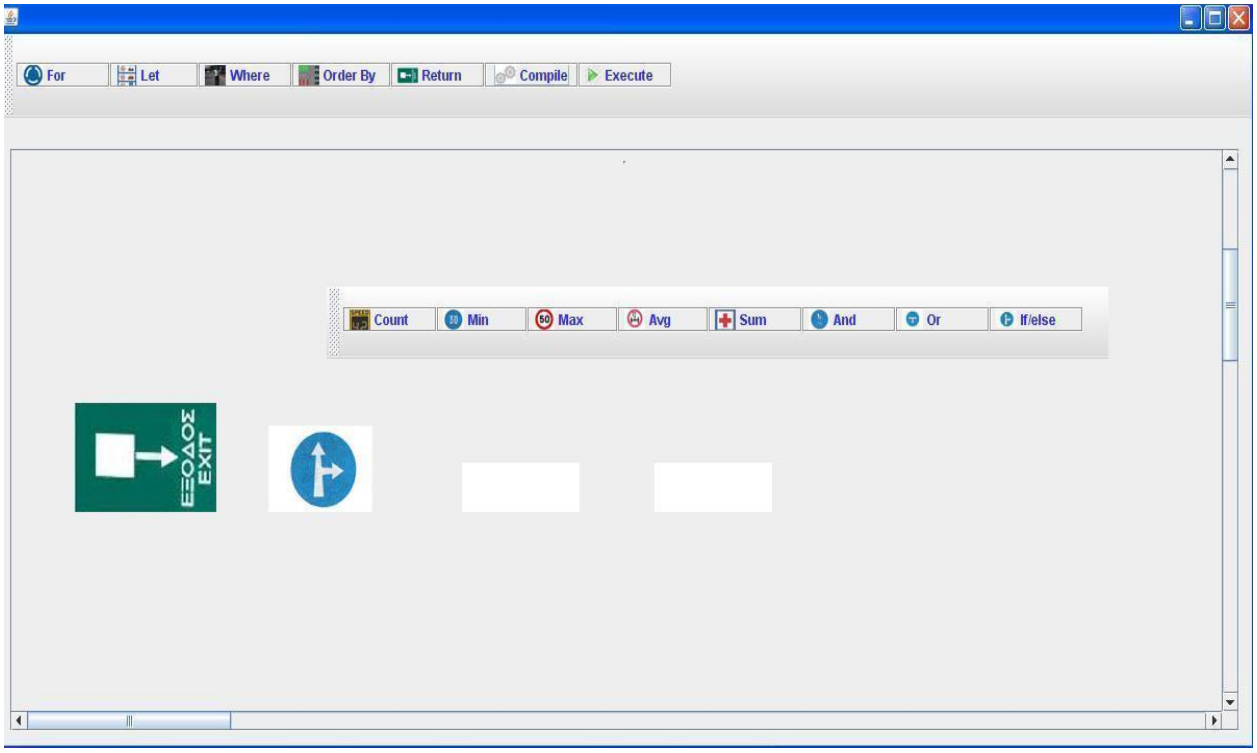


Εικόνα 5.20

Παράδειγμα πρότασης return με τον τελεστή count:

Εικόνα 5.21

Παράδειγμα πρότασης return με τον τελεστή if - else:

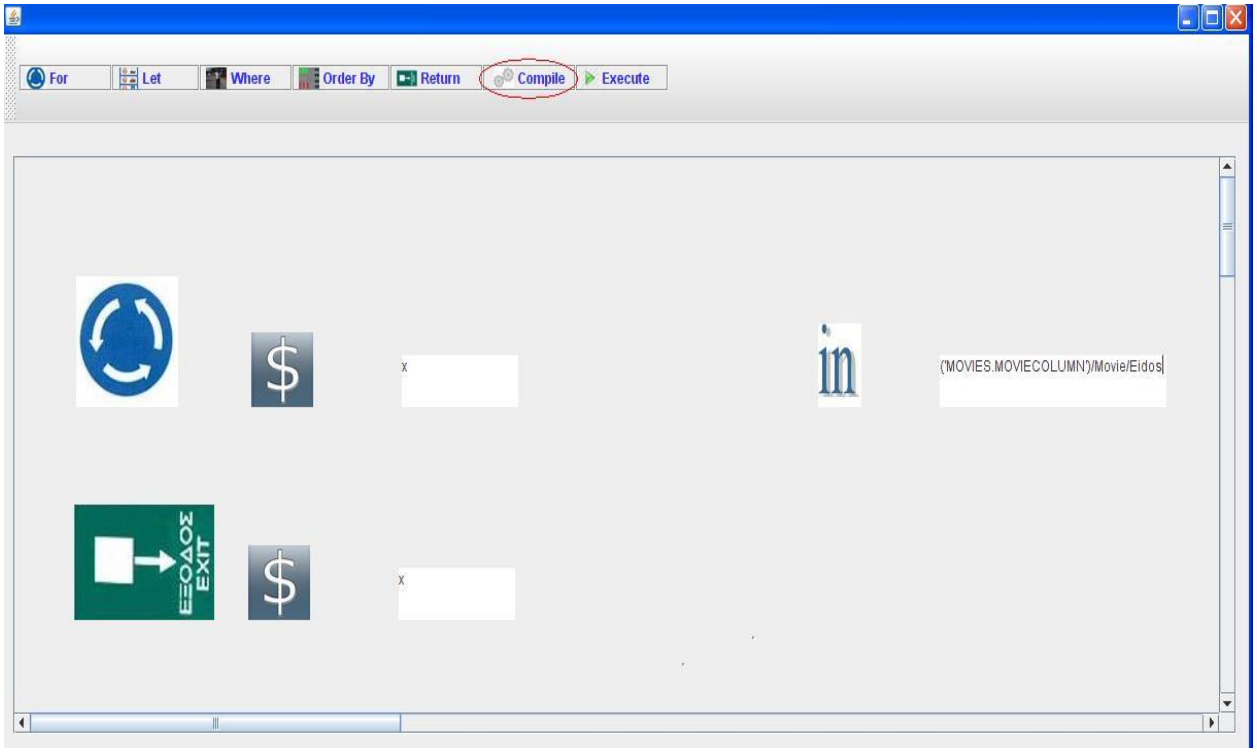


Εικόνα 5.22

5.3.6 Μεταγλώτιση Προγράμματος

Έχοντας συντάξει το ερώτημά μας αυτό που απομένει στον χρήστη είναι να πατήσει στο κουμπί compile έτσι ώστε να δημιουργηθεί το ερώτημα. Θεωρήσαμε ότι είναι καλό για τον χρήστη να βλέπει το ερώτημα που έχει συντάξει πλήρως σε ένα αρχείο text περιέχοντας όλους τους όρους ακόμα και αυτοί που ήταν κρυφοί στην εφαρμογή. Αυτό το κάναμε για να μπορέσει ο χρήστης με την ενασχόλησή του με την εφαρμογή να αποκτήσει ένα βαθμό οικειότητας με την XQuery έτσι ώστε αργότερα να μπορεί και μόνος του να συντάξει το δικό του ερώτημα.
Για να επανέλθουμε στην λειτουργία της εφαρμογής μας ο χρήστης όταν συντάξει το ερώτημα του πρέπει να πατήσει στο κουμπί compile του αρχικού toolbar για δύο λόγους. Ο πρώτος λόγος είναι για να δημιουργηθεί και να εμφανιστεί το αρχείο κειμένου που περιέχει το ερώτημα και κατά δεύτερον να δημιουργηθεί το String στο οποίο θα αποθηκευτεί το ερώτημα που θα σταλεί στην DB2.

Ακολουθεί ένα παράδειγμα πλήρους ερωτήματος XQuery ενώ στον κύκλο σημειώνεται το κουμπί compile:



Εικόνα 5.23

Όπως αναφέραμε πριν όταν πατάμε στο κουμπί compile αυτόματα ανοίγει ένα text αρχείο στο οποίο περιέχεται πλήρως το ερώτημά μας. Για να το καταφέρουμε αυτό χρησιμοποιήσαμε κώδικα java swing και καλέσαμε την μέθοδο Desktop.
Η μέθοδος Desktop είναι πολύ απλή και σαν όρισμα παίρνει μόνο το αρχείο που θέλουμε να ανοίξουμε. Ο κώδικας είναι ο εξής:

```
try {
    Desktop.getDesktop().open(file);
}

catch(Exception exception)  {
    System.out.println("Problem occour when to open the file");
}
```

Παρατηρούμε ότι η μέθοδος περιέχεται σε ένα block try – catch και σε περίπτωση που δημιουργηθεί κάποιο σφάλμα εμφανίζεται το αντίστοιχο μήνυμα

που μας ειδοποιεί ότι υπάρχει πρόβλημα στο να ανοιχτεί αυτό το αρχείο. Μέσα στην παρένθεση του open όπως αναφέρθηκε υπάρχει το όνομα του αρχείου που θέλουμε να ανοίξουμε.
Η μέθοδος Desktop χρησιμοποιήθηκε στην εφαρμογή μας σε όποιο σημείο θέλαμε να ανοίξουμε κάποιο αρχείο.
Για να μπορέσουμε να καλέσουμε την μέθοδο Desktop είναι αναγκαίο να υπάρχει στον κώδικά μας το αντίστοιχο input που στην συγκεκριμένη περίπτωση είναι το:

```
import java.awt.Desktop;
```

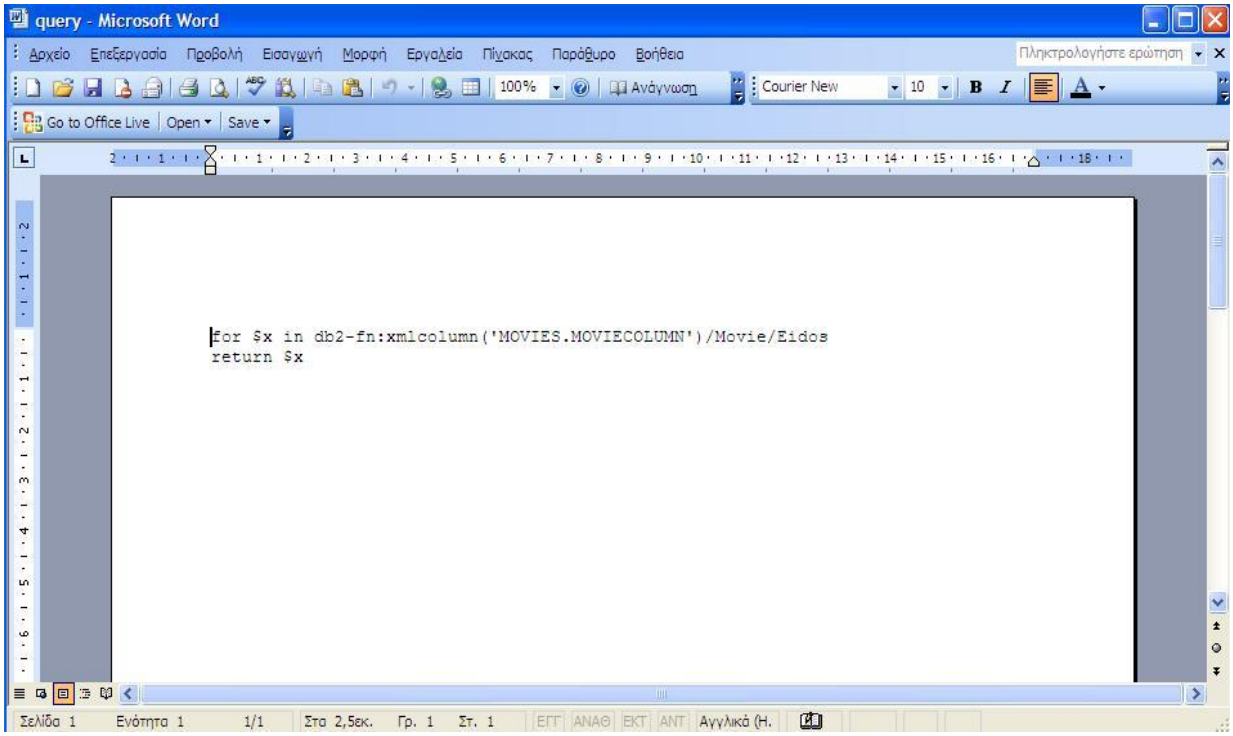
Επίσης για να μπορέσουμε να δημιουργήσουμε ένα αρχείο ο κώδικας που απαιτείται είναι ο εξής:

```
File file = new File("query.rtf");
```

Όπου στο συγκεκριμένο παράδειγμα δημιουργούμε ένα text αρχείο το οποίο το ονομάζουμε “query”.
Απαραίτητο στοιχείο στον κώδικά μας είναι το αντίστοιχο input το οποίο είναι το εξής:

```
import java.io.File;
```

Για να γίνουν ακόμα πιο κατανοητά αυτά που αναφέραμε για το παράδειγμα της προηγούμενης εικόνας το αρχείο που ανοίγει μόλις πατήσουμε το κουμπί compile είναι αυτό που φαίνεται στην επόμενη εικόνα:



Εικόνα 5.24

Με αυτό τον τρόπο ο χρήστης έχει την δυνατότητα να αποθηκεύσει το ερώτημα που έχει δημιουργήσει σε ένα αρχείο κειμένου. Το ερώτημα στέλνετε στο αρχείο αυτό μέσω ενός γράφου writer και χρησιμοποιώντας την μέθοδο write.
Ο γράφος που χρησιμοποιήθηκε είναι:

```
Writer output = null;
```

Ενώ ένα παράδειγμα του πώς μπορούμε να χρησιμοποιήσουμε την μέθοδο write για να στείλουμε κάποια δεδομένα σε ένα αρχείο είναι το εξής:

```
output = new BufferedWriter(new FileWriter(file));
output.write("for $");
```

Εξηγώντας το προηγούμενο παράδειγμα στην πρώτη σειρά δηλώνουμε σε ποιο αρχείο θέλουμε ο γράφος μας να γράψει, και στην δεύτερη σειρά γράφουμε το περιεχόμενο της παρενθέσεως στο συγκεκριμένο αρχείο.

Σημαντικό είναι να μην ξεχάσουμε στον κώδικά μας να συμπληρώσουμε τα κατάλληλα imports που στην συγκεκριμένη περίπτωση είναι τα εξής:

```
import java.io.FileWriter;
import java.io.BufferedWriter;
```

Το κουμπί compile εκτός από το να γράφει το ερώτημα σε ένα text αρχείο όπως προείπαμε δημιουργεί και το string το οποίο θα σταλεί στην DB2 για να εκτελεστεί. Για να γίνει όμως αυτό απαιτείται ένα μεγάλο μπλοκ κώδικα το οποίο θα αποθηκεύει σταδιακά σε ένα string το ερώτημα μας.
Είναι πολύ απλό να γίνει αυτό που αναφέραμε και θα παραθέσουμε το αντίστοιχο παράδειγμα για το πώς εργαστήκαμε:

```
selectString="xquery";
selectString=selectString+'\n';
selectString=selectString+"for $";
```

Το string το οποίο χρησιμοποιήσαμε το ονομάσαμε “selectString” και για να το δημιουργήσουμε προσθέταμε σε αυτό την τιμή που είχε από πριν με το καινούργιο στοιχείο που θέλαμε.
Επειδή στην εφαρμογή μας απαιτήθηκε σε ορισμένα σημεία ο χρήστης να γράψει σε κάποια text areas για να μπορέσουμε να αποθηκεύσουμε αυτό που ο χρήστης εισήγαγε στο text area χρησιμοποιήσαμε την μέθοδο getText().
Δηλαδή αν θέλαμε το περιεχόμενο jTextArea1 θα γράφαμε:

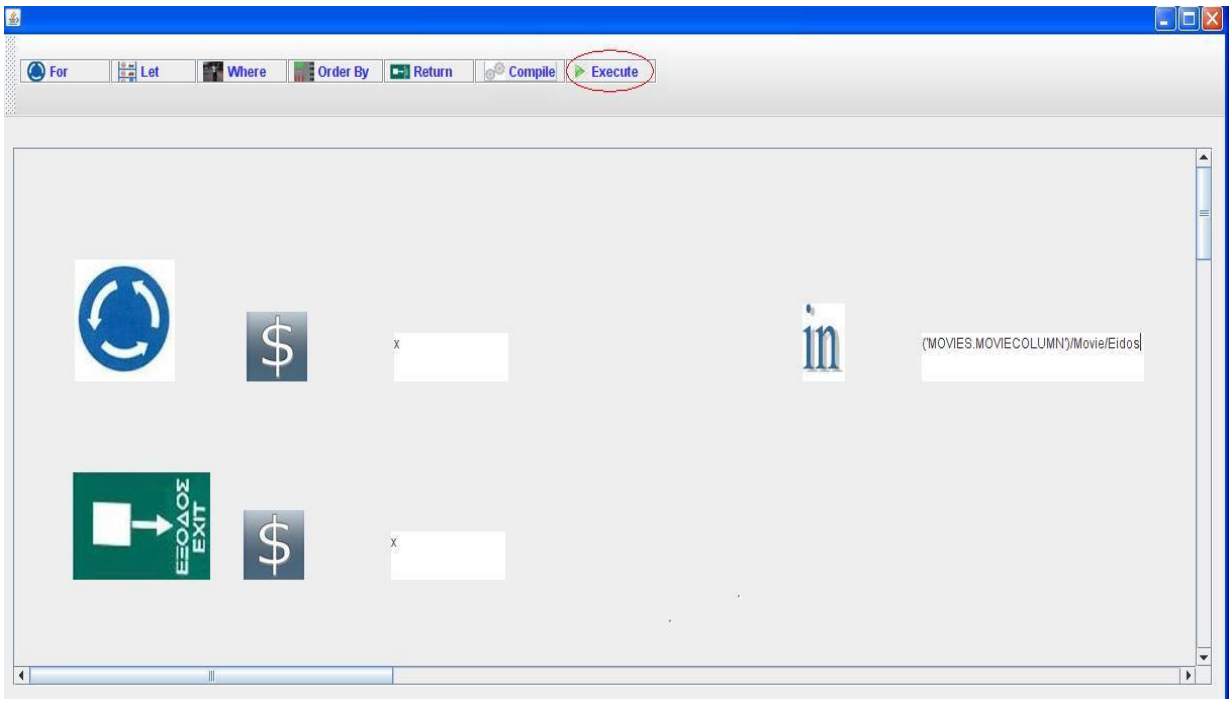
```
jTextArea1.getText();
```

5.3.7. Εκτέλεση Προγράμματος

Κομβικό σημείο στην εφαρμογή μας ήταν να καταφέρουμε να επικοινωνήσει η εφαρμογή μας με την DB2. Αυτό που θέλαμε να πετύχουμε είναι ο χρήστης να μην συμμετέχει στην επικοινωνία την εφαρμογής με την DB2 αλλά όλα να γίνονται αυτόματα.
Έτσι ο χρήστης το μόνο που έχει να κάνει είναι να πατήσει ένα κουμπί και αυτό δεν είναι άλλο από το κουμπί execute με αποτέλεσμα το ερώτημα που σύνταξε ο χρήστης να αποστέλλεται στην DB2. Το κουμπί execute εκτός από το να αποστέλλει το ερώτημα στην DB2 παίρνει τα αποτελέσματα του ερωτήματος από την DB2 και μας τα εμφανίζει σε ένα xml αρχείο το οποίο ονομάζεται result.xml. Έτσι η αλληλεπίδραση του χρήστη με την διεπαφή μειώνεται στο ελάχιστο δυνατό. Επιλέξαμε το xml αρχείο να ανοίγει αυτόματα με το πρόγραμμα Altova Xml Spy 2010 και αυτό γιατί το πρόγραμμα αυτό δίνει την δυνατότητα στον χρήστη να κάνει μια σειρά από ενέργειες στα αποτελέσματα του ερωτήματος.

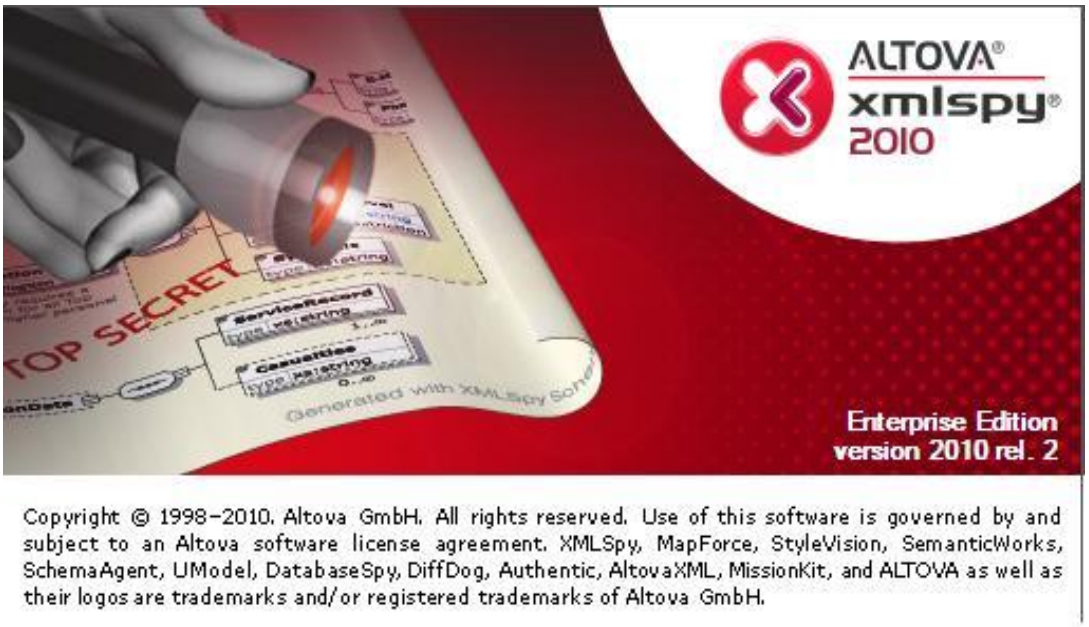
Θα ακολουθήσουν εικόνες με τα βήματα που απαιτούνται για να εκτελέσουμε ένα ερώτημα XQuery στην DB2.

Αφού έχουμε συντάξει το ερώτημά μας και έχουμε πατήσει το κουμπί compile δημιουργείται το ερώτημά μας και το string το οποίο θα αποσταλεί στην DB2. Όπως αναφέραμε και πριν το μόνο που απομένει είναι να πατήσουμε το κουμπί execute το οποίο σημειώνεται σε κόκκινο κύκλο.



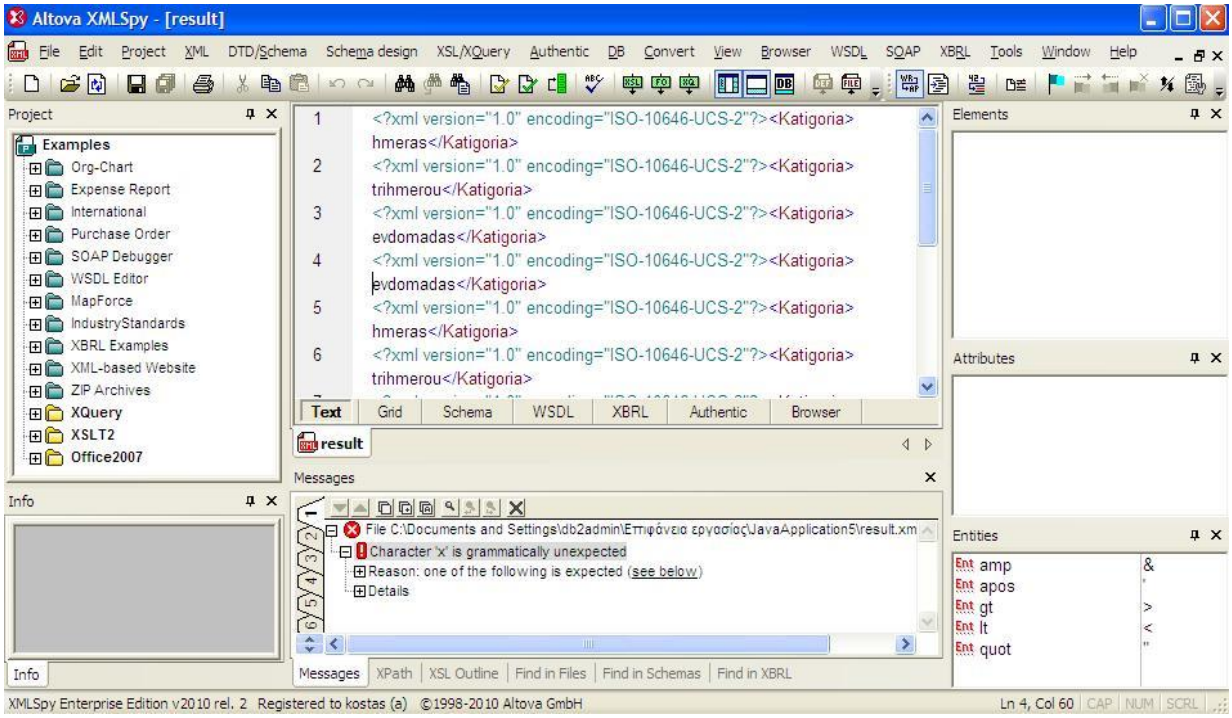
Εικόνα 5.25

Μόλις πατήσουμε το κουμπί execute ανοίγει αυτόματα το Altova Xml Spy παρουσιάζοντας μας τα αποτελέσματα.



Εικόνα 5.26

Και τα αποτελέσματα εμφανίζονται με αυτή την μορφή:



Εικόνα 5.27

Όπως παρατηρούμε στην προηγούμενη εικόνα ο χρήστης μπορεί να επιλέξει μια σειρά από ενέργειες πάνω στα δεδομένα όπως για παράδειγμα να δημιουργήσει το schema των αποτελεσμάτων.

Σε αυτό το σημείο θα αναλύσουμε πώς γίνονται οι λειτουργίες του κουμπιού execute.

Βασικό σημείο στην εφαρμογή μας ήταν καταφέρουμε να επικοινωνήσει με την DB2. Για να συμβεί αυτό απαιτείται στην αρχή του προγράμματός μας να δηλώσουμε κάποια στοιχεία τα οποία είναι τα εξής:

```
static String     driverClassName = "com.ibm.db2.jcc.DB2Driver" ;
static String     url             = "jdbc:db2://localhost:50000/SAMPLE" ;
static String     username = "db2admin";
static String     passwd = "********";
static Connection dbConnection    = null;
static ResultSet rs                 = null;
static String selectString = null;
static Statement statement=null;
```

Θα εξηγήσουμε την λειτουργία που εκτελεί το κάθε στοιχείο από αυτά που είναι απαραίτητο να δηλωθούν στην αρχή του προγράμματός μας. Το πρώτο στοιχείο που δηλώνεται είναι ο driver της DB2 που θα χρησιμοποιηθεί. Το δεύτερο στοιχείο είναι ο server στον οποίο θα δουλέψουμε όπου στην προκειμένη περίπτωση επιλέξαμε να είναι ο υπολογιστής μας. Όπως μπορούμε να προσέξουμε στο τέλος δηλώνεται η βάση δεδομένων που στο παράδειγμά μας είναι η “Sample”. Τα δύο επόμενα στοιχεία που δηλώνονται είναι το username και το password που χρησιμοποιούμε στο profile της DB2. Επόμενο στοιχείο είναι η σύνδεση μεταξύ της εφαρμογής μας και της DB2. Το στοιχείο ResultSet rs είναι το στοιχείο στο οποίο θα αποθηκευτεί το αποτέλεσμα του ερωτήματός μας. Έχουμε αναφέρει ότι το ερώτημά μας αποθηκεύεται σε ένα string που στην προκειμένη περίπτωση το ονομάζουμε selectString. Το τελευταίο στοιχείο είναι αυτό που θα χρησιμοποιήσει την διασύνδεση μεταξύ της εφαρμογής μας και της DB2 και μέσω αυτής θα εκτελεστεί το ερώτημά μας.

Δεν θα πρέπει να ξεχάσουμε να προσθέσουμε το αντίστοιχο import στο πρόγραμμά μας το οποίο είναι το εξής:

```
import java.sql.*;
```

Έχοντας αναφέρει τα στοιχεία που πρέπει να δηλωθούν καθώς και το import που πρέπει να χρησιμοποιηθεί θα αναφέρουμε τον κώδικα ο οποίος είναι απαραίτητος για γίνουν οι λειτουργίες που επιτελεί το κουμπί execute.
Για να καταφέρουμε να επικοινωνήσει η εφαρμογή μας με την DB2 το πρώτο πράγμα που πρέπει να κάνουμε είναι να καλέσουμε τον driver που θα χρησιμοποιήσουμε στην εφαρμογή μας.
Ακολουθεί ο κώδικας που πραγματοποιεί αυτό που μόλις αναφέραμε σε block try - catch ώστε να ελέγξουμε την περίπτωση σφάλματος.

```
try {
        Class.forName(driverClassName);
} catch (ClassNotFoundException ex) {
        Logger.getLogger(ptx.class.getName()).log(Level.SEVERE, null, ex);
}
```

Αφού καλέσαμε τον driver που θα χρησιμοποιήσουμε το επόμενο βήμα είναι να δημιουργήσουμε την σύνδεση μεταξύ της εφαρμογής μας και της DB2. Ο κώδικας που χρησιμοποιήσαμε είναι ο εξής:

```
dbConnection = DriverManager.getConnection (url, username, passwd);
statement=dbConnection.createStatement();
```

Με την δεύτερη σειρά δημιουργούμε μια έκφραση η οποία συνδέεται με την βάση δεδομένων της DB2 και αυτό γίνεται με την μέθοδο createStatement.

Όπως αναφέραμε και πριν θα πρέπει να δηλώσουμε ένα στοιχείο τύπου ResultSet στο οποίο θα αποθηκεύεται το αποτέλεσμα του ερωτήματός μας. Αυτό γίνεται με τον παρακάτω κώδικα:

```
rs = statement.executeQuery(selectString);
```

Η μέθοδος που χρησιμοποιούμε είναι η executeQuery η οποία παίρνει σαν όρισμα το string που περιέχει το ερώτημά μας το οποίο δεν είναι άλλο από το selectString.

Το μόνο που απομένει τώρα είναι να εμφανιστούν τα αποτελέσματα στην εφαρμογή μας και όπως έχουμε αναφέρει θα αποσταλούν τα αποτελέσματα στο

xml αρχείο που χρησιμοποιήσαμε. Θα παρουσιάσουμε το μπλοκ κώδικα που χρησιμοποιήθηκε και μετά θα το αναλύσουμε.

Ο κώδικας ο οποίος χρησιμοποιήθηκε είναι ο εξής:

```
while(rs.next()) {

        com.ibm.db2.jcc.DB2Xml xml = (com.ibm.db2.jcc.DB2Xml)
rs.getObject(1);


        System.out.println (xml.getDB2XmlString());

        apotelesma=apotelesma+xml.getDB2XmlString()+'\n';

}
```

Παρατηρούμε ότι ο κώδικάς μας ξεκινάει με μία επανάληψη while. Η επανάληψη αυτή μας διασφαλίζει ότι αν υπάρχουν παραπάνω από ένα αποτελέσματα θα εμφανιστούν και αυτά. Αν δεν υπήρχε η επανάληψη αυτή το μόνο αποτέλεσμα που θα εκτυπωνόταν θα ήταν το πρώτο απαλείφοντας τα υπόλοιπα. Αυτό όμως είναι κάτι το οποίο θέλουμε να αποφύγουμε και το διασφαλίζουμε με αυτή την επανάληψη.
Η επόμενη σειρά του συγκεκριμένου κώδικα ουσιαστικά αποθηκεύει τα xml αποτελέσματα σε ένα στοιχείο αντίστοιχου τύπου.
Οι δύο τελευταίες σειρές εμφανίζουν τα αποτελέσματα. Η πρώτη σειρά εκτυπώνει τα αποτελέσματά μας στην εφαρμογή μας ενώ η δεύτερη σειρά αποθηκεύει τα αποτελέσματα σε ένα string το οποίο το ονομάζουμε “apotelesma”. Το string είναι αυτό που θα σταλεί μέσω ενός γράφου στο xml αρχείο στο οποίο αποθηκεύουμε τα αποτελέσματά μας.
Για να το καταφέρουμε αυτό χρησιμοποιούμε τον ίδιο κώδικα που χρησιμοποιήθηκε στην αντίστοιχη λειτουργία που επιτελεί και το κουμπί compile και είναι ο εξής:

```
output2 = new BufferedWriter(new FileWriter(file2));

output2.write(apotelesma);

output2.close();
```

Όπως συμβαίνει με έναν γράφο αρχείου έτσι και στην περίπτωση που θέλουμε να συνδέσουμε μια εφαρμογή με την DB2 θα πρέπει να «κλείσουμε» τα στοιχεία τα οποία δεν είναι απαραίτητα και τα οποία ίσως να επαναχρησιμοποιηθούν μεταγενέστερα σε επόμενο ερώτημα.

Για να γίνει πιο κατανοητό παραθέτουμε τον κώδικα:

```
statement.close();
dbConnection.close();
```

Τέλος όπως και στο κουμπί compile έτσι και το κουμπί execute θέλουμε να ανοίγει αυτόματα ένα συγκεκριμένο αρχείο. Ο κώδικας που απαιτείται δεν θα μπορούσε να είναι διαφορετικός και είναι ο παρακάτω:

```
try {
        Desktop.getDesktop().open(file2);
}
catch(Exception exception)  {
        System.out.println("Problem occour when to open the file");
}
```

Με την παρουσίαση και του κουμπιού execute ολοκληρώθηκε η αναφορά μας στις λειτουργίες που μπορεί να επιτελέσει η εφαρμογή μας. Γνώμονάς μας ήταν ακόμα και ένας χρήστης ο οποίος δεν είναι εξοικειωμένος με την XQuery και την DB2 να μπορέσει να συντάξει και να εκτελέσει το δικό του XQuery ερώτημα και μέσω της αλληλεπίδρασης αυτής να αποκτήσει τις γνώσεις που απαιτούνται για να μπορεί και μόνος του να εκτελεί τα ερωτήματά του. Η λογική που χρησιμοποιήσαμε στην διεπαφή μας ήταν ότι στον χρήστη θα πρέπει να δίνονται σταδιακά τα εργαλεία και όχι όλα εξ’ αρχής για να μπορέσει βήμα – βήμα να φτάσει στο επιθυμητό αποτέλεσμα.

Συμπέρασμα

Σκοπός της εφαρμογής μας ήταν να δώσουμε την δυνατότητα ακόμα και στον απλό χρήστη που δεν έχει τις ιδιαίτερες γνώσεις πάνω στην XQuery και στον προγραμματισμό πάνω σε αυτήν, να καταφέρει να συντάξει το δικό του ερώτημα, να το εκτελέσει και να αποθηκεύσει τα αποτελέσματα του ερωτήματός του. Αν και εκ των προτέρων ήταν αρκετά δύσκολο να το καταφέρουμε πιστεύουμε ότι σε μεγάλο βαθμό πετύχαμε τον σκοπό που είχαμε θέσει. Η εφαρμογή μας μπορεί να χρησιμοποιηθεί και για εκπαιδευτικούς σκοπούς ώστε φοιτητές μέσω της ενασχόλησής τους με αυτήν να μάθουν τα δομικά στοιχεία και τον τρόπο σύνταξης της XQuery. Θα θέλαμε να τονίσουμε ότι η εφαρμογή μας έχει ακόμα περιθώρια βελτίωσης. Η εφαρμογή μας θα αναπτυχθεί ακόμα περισσότερο ώστε να κάνει ακόμα πιο αυτοματοποιημένη την διαδικασία σύνταξης ερωτημάτων καθώς και θα αποκτήσει ακόμα περισσότερες δυνατότητες. Μια από αυτές τις δυνατότητες που θέλουμε να προσθέσουμε στην εφαρμογή μας είναι να μπορεί ο χρήστης να συντάξει ένα εμφωλευμένο ερώτημα. Δυστυχώς στο επίπεδο που είναι η εφαρμογή μας, ο χρήστης δεν μπορεί να συνθέσει ένα εμφωλευμένο ερώτημα. Αυτό είναι ένα μεγάλο στοίχημα για εμάς που θέλουμε να κερδίσουμε όταν θα αναβαθμίσουμε την εφαρμογή μας με περισσότερες λειτουργίες. Επίσης πιστεύουμε ότι το διεπιφάνεια της εφαρμογής θα μπορούσε να είναι ακόμα πιο “φιλικό” για τον χρήστη και θα εργαστούμε γι’ αυτό για να το φτάσουμε στο μέγιστο δυνατό σημείο. Η πτυχιακή που αναλάβαμε μας βοήθησε να αποκτήσουμε γνώσεις πάνω στον προγραμματισμό που δεν θα μπορούσαμε να τις αποκτήσουμε αν παρακολουθούσαμε κάποιο αντίστοιχο μάθημα. Έτσι καθώς ψάχναμε στοιχεία για την πτυχιακή μας μάθαμε αρκετά πράγματα για την XQuery όπως επίσης για τον προγραμματισμό Java Swing. Επίσης μελετήσαμε το πώς πρέπει να δομηθεί η διεπιφάνεια μιας εφαρμογής έτσι ώστε να είναι φιλικό προς τον χρήστη,. Κλείνοντας, μέσω της πτυχιακής μας εμπλουτίσαμε τις γνώσεις μας και επίσης μας έδωσε το έναυσμα να ασχοληθούμε ακόμα περισσότερο με τον συγκεκριμένο τομέα της Πληροφορικής που επιλέξαμε να ακολουθήσουμε.

Βιβλιογραφία

Priscilla Walmsley (April 2007), *XQuery – Search across a variety of xml data*, O'Reilly, United States of America.

Peter Fankhauser, Philip Wadler, *XQuery Tutorial*
Fraunhofer IPSI, Avaya Labs.

Kay, Michael (2004), *Xpath 2.0 Programmer's Reference,* Wprox.

Katz, Howard (2003), *XQuery from the Experts*, Addison-Wesley.

D. Braga and A. Campi. (2003) *A Graphical Environment to Query XML Data with XQuery,* In Fourth Intl. Conf. on Web Information Systems Engineering (WISE'03).

Z. Qin, B. B. Yao, Y. Liu and M. McCool, (August, 2004), *A Graphical XQuery Language Using Nested Windows*, School of Computer Science, University of Waterloo.

Robert Eckstein, Marc Loy & Dave Wood, (2000), *Java Swing,* O'Reilly, United States of America.

W3C, (August 2003), *XML Query Use Cases*.
http://www.w3.org/TR/xmlquery-use-cases.

Li X. XGI, (2006), *A graphical interface for XQuery creation and XML schema visualization [Masters],*: University of Washington.
http://sigpubs.biostr.washington.edu/archive/00000198/

World Wide Web Consortium, (2001), *XQuery*.
http://www.w3.org/TR/xquery

W3C XML Query page
http://www.w3.org/XML/Query.html

Χρήσιμα forum και sites:

http://www.java-forums.org
http://forums.netbeans.org
http://www.ibm.com

Παράρτημα

Ακολουθεί όλος ο κώδικας της εφαρμογής μας:

```
package my.jdbc.pkg;
import java.awt.Desktop;
import java.io.BufferedWriter;
import javax.swing.JFileChooser;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.Writer;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.sql.*;
import java.lang.*;
/*
 * ptx.java
 *
 * Created on 5 Ιουν 2009, 6:20:37 μμ
 */

/**
 *
 * @author Administrator
 */


public class ptx extends javax.swing.JFrame {

    /** Creates new form ptx */
    public ptx() {
        initComponents();
    }

    static String     driverClassName = "com.ibm.db2.jcc.DB2Driver" ;
    static String     url             = "jdbc:db2://localhost:50000/SAMPLE" ;
    static String     username = "db2admin";
    static String     passwd = "2021985";
    static Connection dbConnection    = null;
    static ResultSet rs                         = null;
    static String selectString = null;
    static String selectString2 = null;
    static Statement statement=null;
    static String j="";

     Writer output = null;
     Writer output2 = null;

     File file = new File("query.rtf");
     File file2 = new File("result.xml");
     File file3 = new File("diadromes.doc");

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jToolBar2 = new javax.swing.JToolBar();
        jButton1 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jButton5 = new javax.swing.JButton();
        jButton14 = new javax.swing.JButton();
        jButton29 = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jTextArea1 = new javax.swing.JTextArea();
        jScrollPane2 = new javax.swing.JScrollPane();
        jScrollPane3 = new javax.swing.JScrollPane();
        jScrollPane4 = new javax.swing.JScrollPane();
        jScrollPane5 = new javax.swing.JScrollPane();
        jScrollPane6 = new javax.swing.JScrollPane();
```

```
jToolBar1 = new javax.swing.JToolBar();
jButton6 = new javax.swing.JButton();
jButton7 = new javax.swing.JButton();
jButton8 = new javax.swing.JButton();
jButton9 = new javax.swing.JButton();
jButton10 = new javax.swing.JButton();
jButton11 = new javax.swing.JButton();
jButton12 = new javax.swing.JButton();
jButton13 = new javax.swing.JButton();
jToolBar3 = new javax.swing.JToolBar();
jButton15 = new javax.swing.JButton();
jButton16 = new javax.swing.JButton();
jButton17 = new javax.swing.JButton();
jButton18 = new javax.swing.JButton();
jButton19 = new javax.swing.JButton();
jButton20 = new javax.swing.JButton();
jButton21 = new javax.swing.JButton();
jButton22 = new javax.swing.JButton();
jLabel6 = new javax.swing.JLabel();
jLabel7 = new javax.swing.JLabel();
jLabel10 = new javax.swing.JLabel();
jLabel8 = new javax.swing.JLabel();
jTextArea6 = new javax.swing.JTextArea();
jLabel11 = new javax.swing.JLabel();
jTextArea2 = new javax.swing.JTextArea();
jLabel12 = new javax.swing.JLabel();
jTextArea7 = new javax.swing.JTextArea();
jLabel13 = new javax.swing.JLabel();
jTextArea3 = new javax.swing.JTextArea();
jLabel14 = new javax.swing.JLabel();
jTextArea8 = new javax.swing.JTextArea();
jLabel15 = new javax.swing.JLabel();
jTextArea4 = new javax.swing.JTextArea();
jTextArea5 = new javax.swing.JTextArea();
jLabel16 = new javax.swing.JLabel();
jLabel17 = new javax.swing.JLabel();
jTextArea9 = new javax.swing.JTextArea();
jTextArea10 = new javax.swing.JTextArea();
jLabel18 = new javax.swing.JLabel();
jLabel19 = new javax.swing.JLabel();
jLabel20 = new javax.swing.JLabel();
jLabel21 = new javax.swing.JLabel();
jLabel22 = new javax.swing.JLabel();
jToolBar4 = new javax.swing.JToolBar();
jButton23 = new javax.swing.JButton();
jButton25 = new javax.swing.JButton();
jButton24 = new javax.swing.JButton();
jButton26 = new javax.swing.JButton();
jButton27 = new javax.swing.JButton();
jButton28 = new javax.swing.JButton();
jButton31 = new javax.swing.JButton();
jButton32 = new javax.swing.JButton();
jLabel23 = new javax.swing.JLabel();
jLabel24 = new javax.swing.JLabel();
jLabel25 = new javax.swing.JLabel();
jLabel26 = new javax.swing.JLabel();
jLabel9 = new javax.swing.JLabel();
jButton30 = new javax.swing.JButton();
jLabel27 = new javax.swing.JLabel();
jTextArea11 = new javax.swing.JTextArea();
jLabel28 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jToolBar2.setRollover(true);

jButton1.setForeground(new java.awt.Color(51, 51, 255));
jButton1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/for3.jpg"))); // NOI18N
jButton1.setText("For");
jButton1.setFocusable(false);
jButton1.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton1.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton1.setMaximumSize(new java.awt.Dimension(90, 21));
jButton1.setMinimumSize(new java.awt.Dimension(180, 40));
jButton1.setPreferredSize(new java.awt.Dimension(40, 20));
jButton1.setRequestFocusEnabled(false);
jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton1MouseClicked(evt);
    }
});
jToolBar2.add(jButton1);

jButton3.setForeground(new java.awt.Color(51, 51, 255));
jButton3.setIcon(new javax.swing.ImageIcon(getClass().getResource("/let3.jpg"))); // NOI18N
jButton3.setText("Let");
```

```
jButton3.setFocusable(false);
jButton3.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton3.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton3.setMaximumSize(new java.awt.Dimension(85, 21));
jButton3.setMinimumSize(new java.awt.Dimension(49, 21));
jButton3.setPreferredSize(new java.awt.Dimension(49, 21));
jButton3.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton3MouseClicked(evt);
    }
});
jToolBar2.add(jButton3);

jButton2.setForeground(new java.awt.Color(51, 51, 255));
jButton2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/where3.jpg"))); // NOI18N
jButton2.setText("Where");
jButton2.setFocusable(false);
jButton2.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton2.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton2.setMaximumSize(new java.awt.Dimension(90, 21));
jButton2.setMinimumSize(new java.awt.Dimension(51, 21));
jButton2.setPreferredSize(new java.awt.Dimension(51, 21));
jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton2MouseClicked(evt);
    }
});
jToolBar2.add(jButton2);

jButton4.setForeground(new java.awt.Color(51, 51, 255));
jButton4.setIcon(new javax.swing.ImageIcon(getClass().getResource("/order by3.jpg"))); // NOI18N
jButton4.setFocusable(false);
jButton4.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton4.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton4.setLabel("Order By");
jButton4.setMaximumSize(new java.awt.Dimension(95, 21));
jButton4.setMinimumSize(new java.awt.Dimension(65, 21));
jButton4.setPreferredSize(new java.awt.Dimension(65, 21));
jButton4.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton4MouseClicked(evt);
    }
});
jToolBar2.add(jButton4);

jButton5.setForeground(new java.awt.Color(51, 51, 255));
jButton5.setIcon(new javax.swing.ImageIcon(getClass().getResource("/return3.jpg"))); // NOI18N
jButton5.setText("Return");
jButton5.setFocusable(false);
jButton5.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton5.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton5.setMaximumSize(new java.awt.Dimension(90, 21));
jButton5.setMinimumSize(new java.awt.Dimension(51, 21));
jButton5.setPreferredSize(new java.awt.Dimension(51, 21));
jButton5.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton5MouseClicked(evt);
    }
});
jToolBar2.add(jButton5);

jButton14.setForeground(new java.awt.Color(51, 51, 255));
jButton14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/granazi2.jpg"))); // NOI18N
jButton14.setText("Compile");
jButton14.setMaximumSize(new java.awt.Dimension(90, 21));
jButton14.setMinimumSize(new java.awt.Dimension(51, 21));
jButton14.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton14MouseClicked(evt);
    }
});
jToolBar2.add(jButton14);

jButton29.setForeground(new java.awt.Color(51, 51, 255));
jButton29.setIcon(new javax.swing.ImageIcon(getClass().getResource("/execute2.jpg"))); // NOI18N
jButton29.setText("Execute");
jButton29.setFocusable(false);
jButton29.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton29.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton29.setMaximumSize(new java.awt.Dimension(90, 21));
jButton29.setMinimumSize(new java.awt.Dimension(51, 21));
jButton29.setPreferredSize(new java.awt.Dimension(51, 21));
jButton29.setVerticalTextPosition(javax.swing.SwingConstants.BOTTOM);
jButton29.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton29MouseClicked(evt);
```

```
    }
});
jToolBar2.add(jButton29);

jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/for2.JPG"))); // NOI18N
jLabel1.setVisible(false);

jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/where2.JPG"))); // NOI18N
jLabel2.setVisible(false);

jLabel3.setIcon(new javax.swing.ImageIcon(getClass().getResource("/let2.JPG"))); // NOI18N
jLabel3.setVisible(false);

jLabel4.setIcon(new javax.swing.ImageIcon(getClass().getResource("/order by2.JPG"))); // NOI18N
jLabel4.setVisible(false);
jLabel4.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jLabel4MouseClicked(evt);
    }
});

jLabel5.setIcon(new javax.swing.ImageIcon(getClass().getResource("/return22.jpg"))); // NOI18N
jLabel5.setVisible(false);
jLabel5.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jLabel5MouseClicked(evt);
    }
});

jTextArea1.setVisible(false);
jTextArea1.setColumns(20);
jTextArea1.setRows(5);
jTextArea1.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea1MouseMoved(evt);
    }
});
jTextArea1.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea1KeyTyped(evt);
    }
});

jToolBar1.setRollover(true);
jToolBar1.setVisible(false);

jButton6.setForeground(new java.awt.Color(51, 51, 255));
jButton6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/count3.jpg"))); // NOI18N
jButton6.setText("Count");
jButton6.setFocusable(false);
jButton6.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton6.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton6.setMaximumSize(new java.awt.Dimension(90, 21));
jButton6.setMinimumSize(new java.awt.Dimension(49, 21));
jButton6.setPreferredSize(new java.awt.Dimension(49, 21));
jButton6.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton6MouseClicked(evt);
    }
});
jToolBar1.add(jButton6);

jButton7.setForeground(new java.awt.Color(51, 51, 255));
jButton7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/min3.jpg"))); // NOI18N
jButton7.setText("Min");
jButton7.setFocusable(false);
jButton7.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton7.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton7.setMaximumSize(new java.awt.Dimension(87, 21));
jButton7.setMinimumSize(new java.awt.Dimension(49, 21));
jButton7.setPreferredSize(new java.awt.Dimension(49, 21));
jButton7.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton7MouseClicked(evt);
    }
});
jToolBar1.add(jButton7);

jButton8.setForeground(new java.awt.Color(51, 51, 255));
jButton8.setIcon(new javax.swing.ImageIcon(getClass().getResource("/max3.jpg"))); // NOI18N
jButton8.setText("Max");
jButton8.setFocusable(false);
jButton8.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton8.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton8.setMaximumSize(new java.awt.Dimension(87, 21));
jButton8.setMinimumSize(new java.awt.Dimension(49, 21));
```

```
jButton8.setPreferredSize(new java.awt.Dimension(49, 21));
jButton8.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton8MouseClicked(evt);
    }
});
jToolBar1.add(jButton8);

jButton9.setForeground(new java.awt.Color(51, 51, 255));
jButton9.setIcon(new javax.swing.ImageIcon(getClass().getResource("/avg3.jpg"))); // NOI18N
jButton9.setText("Avg");
jButton9.setFocusable(false);
jButton9.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton9.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton9.setMaximumSize(new java.awt.Dimension(87, 21));
jButton9.setMinimumSize(new java.awt.Dimension(49, 21));
jButton9.setPreferredSize(new java.awt.Dimension(49, 21));
jButton9.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton9MouseClicked(evt);
    }
});
jToolBar1.add(jButton9);

jButton10.setForeground(new java.awt.Color(51, 51, 255));
jButton10.setIcon(new javax.swing.ImageIcon(getClass().getResource("/sum3.jpg"))); // NOI18N
jButton10.setText("Sum");
jButton10.setFocusable(false);
jButton10.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton10.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton10.setMaximumSize(new java.awt.Dimension(90, 21));
jButton10.setMinimumSize(new java.awt.Dimension(49, 21));
jButton10.setPreferredSize(new java.awt.Dimension(49, 21));
jButton10.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton10MouseClicked(evt);
    }
});
jToolBar1.add(jButton10);

jButton11.setForeground(new java.awt.Color(51, 51, 255));
jButton11.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and3.jpg"))); // NOI18N
jButton11.setText("And");
jButton11.setFocusable(false);
jButton11.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton11.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton11.setMaximumSize(new java.awt.Dimension(90, 21));
jButton11.setMinimumSize(new java.awt.Dimension(49, 21));
jButton11.setPreferredSize(new java.awt.Dimension(49, 21));
jButton11.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton11MouseClicked(evt);
    }
});
jToolBar1.add(jButton11);

jButton12.setForeground(new java.awt.Color(51, 51, 255));
jButton12.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or3.jpg"))); // NOI18N
jButton12.setText("Or");
jButton12.setFocusable(false);
jButton12.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton12.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton12.setMaximumSize(new java.awt.Dimension(90, 21));
jButton12.setMinimumSize(new java.awt.Dimension(49, 21));
jButton12.setPreferredSize(new java.awt.Dimension(49, 21));
jButton12.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton12MouseClicked(evt);
    }
});
jToolBar1.add(jButton12);

jButton13.setForeground(new java.awt.Color(51, 51, 255));
jButton13.setIcon(new javax.swing.ImageIcon(getClass().getResource("/if else3.jpg"))); // NOI18N
jButton13.setText("If/else");
jButton13.setFocusable(false);
jButton13.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton13.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton13.setMaximumSize(new java.awt.Dimension(90, 21));
jButton13.setMinimumSize(new java.awt.Dimension(49, 21));
jButton13.setPreferredSize(new java.awt.Dimension(49, 21));
jButton13.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton13MouseClicked(evt);
    }
});
```

```
jToolBar1.add(jButton13);

jToolBar3.setRollover(true);
jToolBar3.setVisible(false);

jButton15.setForeground(new java.awt.Color(51, 51, 255));
jButton15.setIcon(new javax.swing.ImageIcon(getClass().getResource("/count3.jpg"))); // NOI18N
jButton15.setText("Count");
jButton15.setFocusable(false);
jButton15.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton15.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton15.setMaximumSize(new java.awt.Dimension(90, 21));
jButton15.setMinimumSize(new java.awt.Dimension(49, 21));
jButton15.setPreferredSize(new java.awt.Dimension(49, 21));
jButton15.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton15MouseClicked(evt);
    }
});
jToolBar3.add(jButton15);

jButton16.setForeground(new java.awt.Color(51, 51, 255));
jButton16.setIcon(new javax.swing.ImageIcon(getClass().getResource("/min3.jpg"))); // NOI18N
jButton16.setText("Min");
jButton16.setFocusable(false);
jButton16.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton16.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton16.setMaximumSize(new java.awt.Dimension(87, 21));
jButton16.setMinimumSize(new java.awt.Dimension(49, 21));
jButton16.setPreferredSize(new java.awt.Dimension(49, 21));
jButton16.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton16MouseClicked(evt);
    }
});
jToolBar3.add(jButton16);

jButton17.setForeground(new java.awt.Color(51, 51, 255));
jButton17.setIcon(new javax.swing.ImageIcon(getClass().getResource("/max3.jpg"))); // NOI18N
jButton17.setText("Max");
jButton17.setFocusable(false);
jButton17.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton17.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton17.setMaximumSize(new java.awt.Dimension(87, 21));
jButton17.setMinimumSize(new java.awt.Dimension(49, 21));
jButton17.setPreferredSize(new java.awt.Dimension(49, 21));
jButton17.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton17MouseClicked(evt);
    }
});
jToolBar3.add(jButton17);

jButton18.setForeground(new java.awt.Color(51, 51, 255));
jButton18.setIcon(new javax.swing.ImageIcon(getClass().getResource("/avg3.jpg"))); // NOI18N
jButton18.setText("Avg");
jButton18.setFocusable(false);
jButton18.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton18.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton18.setMaximumSize(new java.awt.Dimension(87, 21));
jButton18.setMinimumSize(new java.awt.Dimension(49, 21));
jButton18.setPreferredSize(new java.awt.Dimension(49, 21));
jButton18.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton18MouseClicked(evt);
    }
});
jToolBar3.add(jButton18);

jButton19.setForeground(new java.awt.Color(51, 51, 255));
jButton19.setIcon(new javax.swing.ImageIcon(getClass().getResource("/sum3.jpg"))); // NOI18N
jButton19.setText("Sum");
jButton19.setFocusable(false);
jButton19.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton19.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton19.setMaximumSize(new java.awt.Dimension(90, 21));
jButton19.setMinimumSize(new java.awt.Dimension(49, 21));
jButton19.setPreferredSize(new java.awt.Dimension(49, 21));
jButton19.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton19MouseClicked(evt);
    }
});
jToolBar3.add(jButton19);

jButton20.setForeground(new java.awt.Color(51, 51, 255));
```

```
jButton20.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and3.jpg"))); // NOI18N
jButton20.setText("And");
jButton20.setFocusable(false);
jButton20.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton20.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton20.setMaximumSize(new java.awt.Dimension(90, 21));
jButton20.setMinimumSize(new java.awt.Dimension(49, 21));
jButton20.setPreferredSize(new java.awt.Dimension(49, 21));
jButton20.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton20MouseClicked(evt);
    }
});
jToolBar3.add(jButton20);

jButton21.setForeground(new java.awt.Color(51, 51, 255));
jButton21.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or3.jpg"))); // NOI18N
jButton21.setText("Or");
jButton21.setFocusable(false);
jButton21.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton21.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton21.setMaximumSize(new java.awt.Dimension(90, 21));
jButton21.setMinimumSize(new java.awt.Dimension(49, 21));
jButton21.setPreferredSize(new java.awt.Dimension(49, 21));
jButton21.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton21MouseClicked(evt);
    }
});
jToolBar3.add(jButton21);

jButton22.setForeground(new java.awt.Color(51, 51, 255));
jButton22.setIcon(new javax.swing.ImageIcon(getClass().getResource("/if else3.jpg"))); // NOI18N
jButton22.setText("If/else");
jButton22.setFocusable(false);
jButton22.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton22.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton22.setMaximumSize(new java.awt.Dimension(90, 21));
jButton22.setMinimumSize(new java.awt.Dimension(49, 21));
jButton22.setPreferredSize(new java.awt.Dimension(49, 21));
jButton22.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton22MouseClicked(evt);
    }
});
jToolBar3.add(jButton22);

jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/dolar2.jpg"))); // NOI18N
jLabel6.setVisible(false);

jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/dolar2.jpg"))); // NOI18N
jLabel7.setVisible(false);

jLabel10.setFont(new java.awt.Font("Tahoma", 0, 36));
jLabel10.setIcon(new javax.swing.ImageIcon(getClass().getResource("/in2.JPG"))); // NOI18N
jLabel10.setVisible(false);

jLabel8.setIcon(new javax.swing.ImageIcon(getClass().getResource("/dolar2.jpg"))); // NOI18N
jLabel8.setVisible(false);

jTextArea6.setColumns(20);
jTextArea6.setRows(5);
jTextArea6.setVisible(false);
jTextArea6.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea6MouseMoved(evt);
    }
});
jTextArea6.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea6KeyTyped(evt);
    }
});

jLabel11.setIcon(new javax.swing.ImageIcon(getClass().getResource("/dolar2.jpg"))); // NOI18N
jLabel11.setVisible(false);

jTextArea2.setColumns(20);
jTextArea2.setRows(5);
jTextArea2.setVisible(false);
jTextArea2.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea2MouseMoved(evt);
    }
});
jTextArea2.addKeyListener(new java.awt.event.KeyAdapter() {
```

```
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea2KeyTyped(evt);
    }
});

jLabel12.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ison.JPG"))); // NOI18N
jLabel12.setVisible(false);

jTextArea7.setVisible(false);
jTextArea7.setColumns(20);
jTextArea7.setRows(5);
jTextArea7.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea7MouseMoved(evt);
    }
});
jTextArea7.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea7KeyTyped(evt);
    }
});

jLabel13.setIcon(new javax.swing.ImageIcon(getClass().getResource("/dolar2.jpg"))); // NOI18N
jLabel13.setVisible(false);

jTextArea3.setColumns(20);
jTextArea3.setRows(5);
jTextArea3.setVisible(false);
jTextArea3.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea3MouseMoved(evt);
    }
});
jTextArea3.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea3KeyTyped(evt);
    }
});

jLabel14.setDoubleBuffered(true);
jLabel14.setVisible(false);

jTextArea8.setColumns(20);
jTextArea8.setRows(5);
jTextArea8.setVisible(false);
jTextArea8.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea8MouseMoved(evt);
    }
});
jTextArea8.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea8KeyTyped(evt);
    }
});

jLabel15.setText("Δώσε Όνομα Μεταβλητής");
jLabel15.setVisible(false);

jTextArea4.setColumns(20);
jTextArea4.setRows(5);
jTextArea4.setVisible(false);
jTextArea4.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea4MouseMoved(evt);
    }
});
jTextArea4.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea4KeyTyped(evt);
    }
});

jTextArea5.setColumns(20);
jTextArea5.setRows(5);
jTextArea5.setVisible(false);
jTextArea5.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
    public void mouseMoved(java.awt.event.MouseEvent evt) {
        jTextArea5MouseMoved(evt);
    }
});
jTextArea5.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        jTextArea5KeyTyped(evt);
    }
});
```

```
jLabel16.setText("jLabel16");
jLabel16.setVisible(false);

jLabel17.setText("jLabel17");
jLabel17.setVisible(false);

jTextArea9.setColumns(20);
jTextArea9.setRows(5);
jTextArea9.setVisible(false);

jTextArea10.setColumns(20);
jTextArea10.setRows(5);
jTextArea10.setVisible(false);

jLabel18.setVisible(false);

jLabel19.setVisible(false);

jLabel20.setText("Δώσε Όνομα Μεταβλητής");
jLabel20.setVisible(false);

jLabel21.setText("Δώσε Τιμή Μεταβλητής");
jLabel21.setVisible(false);

jLabel22.setText("Δώσε Κριτήριο Καθορισμού Αποτελέσματος");
jLabel22.setVisible(false);

jToolBar4.setRollover(true);
jToolBar4.setVisible(false);

jButton23.setForeground(new java.awt.Color(51, 51, 255));
jButton23.setText("=");
jButton23.setFocusable(false);
jButton23.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton23.setMaximumSize(new java.awt.Dimension(50, 21));
jButton23.setMinimumSize(new java.awt.Dimension(30, 21));
jButton23.setPreferredSize(new java.awt.Dimension(49, 21));
jButton23.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton23MouseClicked(evt);
    }
});
jToolBar4.add(jButton23);

jButton25.setForeground(new java.awt.Color(51, 51, 255));
jButton25.setText("<");
jButton25.setFocusable(false);
jButton25.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton25.setMaximumSize(new java.awt.Dimension(50, 21));
jButton25.setMinimumSize(new java.awt.Dimension(30, 21));
jButton25.setPreferredSize(new java.awt.Dimension(49, 21));
jButton25.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton25MouseClicked(evt);
    }
});
jToolBar4.add(jButton25);

jButton24.setForeground(new java.awt.Color(51, 51, 255));
jButton24.setText(">");
jButton24.setFocusable(false);
jButton24.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton24.setMaximumSize(new java.awt.Dimension(50, 21));
jButton24.setMinimumSize(new java.awt.Dimension(30, 21));
jButton24.setPreferredSize(new java.awt.Dimension(49, 21));
jButton24.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton24MouseClicked(evt);
    }
});
jToolBar4.add(jButton24);

jButton26.setForeground(new java.awt.Color(51, 51, 255));
jButton26.setText(">=");
jButton26.setFocusable(false);
jButton26.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton26.setMaximumSize(new java.awt.Dimension(50, 21));
jButton26.setMinimumSize(new java.awt.Dimension(30, 21));
jButton26.setPreferredSize(new java.awt.Dimension(49, 21));
jButton26.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton26MouseClicked(evt);
    }
});
jToolBar4.add(jButton26);
```

```
jButton27.setForeground(new java.awt.Color(51, 51, 255));
jButton27.setText("=<");
jButton27.setFocusable(false);
jButton27.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton27.setMaximumSize(new java.awt.Dimension(50, 21));
jButton27.setMinimumSize(new java.awt.Dimension(30, 21));
jButton27.setPreferredSize(new java.awt.Dimension(49, 21));
jButton27.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton27MouseClicked(evt);
    }
});
jToolBar4.add(jButton27);

jButton28.setForeground(new java.awt.Color(51, 51, 255));
jButton28.setText("!=");
jButton28.setFocusable(false);
jButton28.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
jButton28.setMaximumSize(new java.awt.Dimension(50, 21));
jButton28.setMinimumSize(new java.awt.Dimension(30, 21));
jButton28.setPreferredSize(new java.awt.Dimension(49, 21));
jButton28.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton28MouseClicked(evt);
    }
});
jToolBar4.add(jButton28);

jButton31.setForeground(new java.awt.Color(51, 51, 255));
jButton31.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and3.jpg"))); // NOI18N
jButton31.setText("And");
jButton31.setFocusable(false);
jButton31.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton31.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton31.setMaximumSize(new java.awt.Dimension(90, 21));
jButton31.setMinimumSize(new java.awt.Dimension(49, 21));
jButton31.setPreferredSize(new java.awt.Dimension(49, 21));
jButton31.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton31MouseClicked(evt);
    }
});
jToolBar4.add(jButton31);

jButton32.setForeground(new java.awt.Color(51, 51, 255));
jButton32.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or3.jpg"))); // NOI18N
jButton32.setText("Or");
jButton32.setFocusable(false);
jButton32.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton32.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton32.setMaximumSize(new java.awt.Dimension(90, 21));
jButton32.setMinimumSize(new java.awt.Dimension(49, 21));
jButton32.setPreferredSize(new java.awt.Dimension(49, 21));
jButton32.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jButton32MouseClicked(evt);
    }
});
jToolBar4.add(jButton32);

jLabel23.setText("Δώσε Τιμή Κριτηρίου");
jLabel23.setVisible(false);

jLabel24.setText("jLabel24");
jLabel24.setVisible(false);

jLabel25.setText("Δώσε Επιστρεφόμενη Έκφραση και πάτησε στο metaform");
jLabel25.setVisible(false);

jLabel26.setText("Δώσε Κριτήριο Ταξινόμησης και πάτησε στο metaform");
jLabel26.setVisible(false);

jLabel9.setText("Δώσε την Διαδρομή.Για Βοήθεια πατήστε στο κουμπί Help");
jLabel9.setVisible(false);

jButton30.setForeground(new java.awt.Color(51, 51, 255));
jButton30.setText("Help");
jButton30.setFocusable(false);
jButton30.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jButton30.setHorizontalTextPosition(javax.swing.SwingConstants.RIGHT);
jButton30.setMaximumSize(new java.awt.Dimension(90, 21));
jButton30.setMinimumSize(new java.awt.Dimension(51, 21));
jButton30.setPreferredSize(new java.awt.Dimension(51, 21));
jButton30.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
```

```
            jButton30MouseClicked(evt);
        }
    });
    jButton30.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton30ActionPerformed(evt);
        }
    });
    jButton30.setVisible(false);

    jLabel27.setText(" ");
    jLabel27.setVisible(false);

    jTextArea11.setVisible(false);
    jTextArea11.setColumns(20);
    jTextArea11.setRows(5);
    jTextArea11.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
        public void mouseMoved(java.awt.event.MouseEvent evt) {
            jTextArea11MouseMoved(evt);
        }
    });
    jTextArea11.addKeyListener(new java.awt.event.KeyAdapter() {
        public void keyTyped(java.awt.event.KeyEvent evt) {
            jTextArea11KeyTyped(evt);
        }
    });

    jLabel28.setText("jLabel28");
    jLabel28.setVisible(false);

    org.jdesktop.layout.GroupLayout jPanel1Layout = new org.jdesktop.layout.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(
        jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
        .add(jPanel1Layout.createSequentialGroup()
            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                .add(jPanel1Layout.createSequentialGroup()
                    .add(60, 60, 60)
                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                .add(jLabel5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 137,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(jLabel4))
                            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED))
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                                .add(jLabel1)
                                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                    .add(jLabel3)
                                    .add(jLabel2)))
                            .add(38, 38, 38)))
                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(33, 33, 33)
                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                    .add(jLabel8)
                                    .add(jLabel11)
                                    .add(jLabel6)
                                    .add(jLabel7))
                                .add(86, 86, 86)
                                .add(jTextArea4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                    .add(jPanel1Layout.createSequentialGroup()
                                        .add(jTextArea5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                        .add(38, 38, 38)
                                        .add(jLabel19)
                                        .add(34, 34, 34)
                                        .add(jTextArea10, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                        .add(119, 119, 119)
                                        .add(jLabel17))
                                    .add(jPanel1Layout.createSequentialGroup()
                                        .add(272, 272, 272)
                                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                            .add(jPanel1Layout.createSequentialGroup()
                                                .add(0, 0, 0)
                                                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                                                    .add(jPanel1Layout.createSequentialGroup()
                                                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                                            .add(jPanel1Layout.createSequentialGroup()
```

```
                                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                        .add(jScrollPane3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                                    .add(jPanel1Layout.createSequentialGroup()
                                        .add(16, 16, 16)
                                        .add(jLabel18)))
                                .add(35, 35, 35)
                                .add(jTextArea9, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .add(jLabel16)
                                .add(515, 515, 515))
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(68, 68, 68)
                                .add(jScrollPane2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(802, 802, 802)
                                .add(jScrollPane6, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
                        .add(15, 15, 15)
                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(78, 78, 78)
                                .add(jScrollPane5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(67, 67, 67)
                                .add(jScrollPane4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(132, 132, 132)
                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jLabel10)
                            .add(jLabel12))
                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(76, 76, 76)
                                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                                    .add(jTextArea7, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .add(jLabel21)
                                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jTextArea6)
                                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel9,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(183, 183, 183)
                                .add(jButton30, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 60,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                .add(jPanel1Layout.createSequentialGroup()
                    .add(47, 47, 47)
                    .add(jLabel14, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 43,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(110, 110, 110)
                            .add(jLabel23))
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(59, 59, 59)
                            .add(jTextArea8, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 171,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                            .add(53, 53, 53)
                            .add(jLabel27)
                            .add(28, 28, 28)
                            .add(jTextArea11, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 148,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))))
            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                .add(jTextArea2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(jTextArea1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(jTextArea3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 113,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel15)
                .add(jLabel20))))
        .add(jLabel13)))
    .add(jPanel1Layout.createSequentialGroup()
        .add(80, 80, 80)
        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
            .add(jToolBar1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 786,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
            .add(jToolBar3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 786,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
    .add(jPanel1Layout.createSequentialGroup()
        .add(123, 123, 123)
        .add(jLabel22))
```

```
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(264, 264, 264)
                        .add(jToolBar4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 448,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(188, 188, 188)
                        .add(jLabel24))
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(170, 170, 170)
                        .add(jLabel25))
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(201, 201, 201)
                        .add(jLabel26))))
                .add(jPanel1Layout.createSequentialGroup()
                    .add(809, 809, 809)
                    .add(jLabel28)))
                .addContainerGap(2073, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1Layout.createSequentialGroup()
                .add(11, 11, 11)
                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel8)
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(83, 83, 83)
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                .add(jLabel1)
                                .add(jPanel1Layout.createSequentialGroup()
                                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                        .add(jPanel1Layout.createSequentialGroup()
                                            .add(23, 23, 23)
                                            .add(jLabel15))
                                        .add(jPanel1Layout.createSequentialGroup()
                                            .add(jLabel9)
                                            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                            .add(jButton30, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, 23, Short.MAX_VALUE)
                                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                                        .add(jTextArea1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                        .add(jTextArea6, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))))
                        .add(jLabel10))
                    .add(89, 89, 89)
                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanel1Layout.createSequentialGroup()
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                .add(jLabel20)
                                .add(jLabel21))
                            .add(33, 33, 33)
                            .add(jTextArea2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel12)
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jTextArea7, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel11)
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel3))
                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(34, 34, 34)
                            .add(jLabel24)
                            .add(7, 7, 7)
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                                .add(jLabel2)
                                .add(jLabel13)
                                .add(jPanel1Layout.createSequentialGroup()
                                    .add(jLabel22)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jTextArea3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                        .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanel1Layout.createSequentialGroup()
                            .add(18, 18, 18)
                            .add(jToolBar4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 60,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                                .add(jLabel14, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 53,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(jPanel1Layout.createSequentialGroup()
                                    .add(jLabel23)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                        .add(jTextArea8, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
```

```
                        .add(jLabel27)
                        .add(jTextArea11, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))))
            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
               .add(jPanel1Layout.createSequentialGroup()
                  .add(jScrollPane2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(18, 18, 18)
                  .add(jToolBar3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 60,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                     .add(org.jdesktop.layout.GroupLayout.TRAILING, jPanel1Layout.createSequentialGroup()
                        .add(jScrollPane3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(97, 97, 97))
                     .add(jPanel1Layout.createSequentialGroup()
                        .add(63, 63, 63)
                        .add(jLabel16))
                     .add(jPanel1Layout.createSequentialGroup()
                        .add(66, 66, 66)
                        .add(jLabel18))))
               .add(jPanel1Layout.createSequentialGroup()
                  .add(73, 73, 73)
                  .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING, false)
                     .add(jLabel4)
                     .add(jPanel1Layout.createSequentialGroup()
                        .add(25, 25, 25)
                        .add(jLabel26)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                           .add(jTextArea4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                           .add(jLabel6)
                           .add(jTextArea9, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .add(5, 5, 5)))))
            .add(90, 90, 90)
            .add(jToolBar1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 60,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
            .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
               .add(jPanel1Layout.createSequentialGroup()
                  .add(12, 12, 12)
                  .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                     .add(jScrollPane4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                     .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                        .add(jLabel7)
                        .add(jLabel5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 91,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(jPanel1Layout.createSequentialGroup()
                           .add(jLabel25)
                           .add(18, 18, 18)
                           .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                              .add(jTextArea5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jTextArea10, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 41,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabel19)))))
                  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, 139, Short.MAX_VALUE)
                  .add(jScrollPane5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(110, 110, 110)
                  .add(jScrollPane6, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(1289, 1289, 1289))
               .add(jPanel1Layout.createSequentialGroup()
                  .add(74, 74, 74)
                  .add(jLabel17)
                  .add(171, 171, 171)
                  .add(jLabel28)
                  .addContainerGap())))
      );

      jScrollPane1.setViewportView(jPanel1);

      org.jdesktop.layout.GroupLayout layout = new org.jdesktop.layout.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(
         layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
         .add(jToolBar2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 1426, Short.MAX_VALUE)
         .add(layout.createSequentialGroup()
            .add(10, 10, 10)
            .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 1406, Short.MAX_VALUE)
            .addContainerGap())
```

```
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(jToolBar2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 70,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(26, 26, 26)
                .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 1445, Short.MAX_VALUE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>


    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel1.setVisible(true);
        jLabel10.setVisible(true);
        jTextArea1.setVisible(true);
        jTextArea6.setVisible(true);



}

    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel2.setVisible(true);
        jTextArea3.setVisible(true);

}

    private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel3.setVisible(true);
        jTextArea2.setVisible(true);
}

    private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel4.setVisible(true);
        jTextArea4.setVisible(true);
}

    private void jButton5MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel5.setVisible(true);
        jTextArea5.setVisible(true);
}

    private void jLabel4MouseClicked(java.awt.event.MouseEvent evt) {
        jToolBar3.setVisible(true);
    }

    private void jLabel5MouseClicked(java.awt.event.MouseEvent evt) {
        jToolBar1.setVisible(true);
    }

    private void jButton6MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/count22.JPG"))); // NOI18N
        jLabel7.setVisible(true);
        jToolBar1.setVisible(false);
        jLabel17.setText("1");
        jLabel19.setVisible(false);
        jTextArea10.setVisible(false);
    }

    private void jButton7MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/min2.JPG"))); // NOI18N
        jLabel7.setVisible(true);
        jToolBar1.setVisible(false);
        jLabel17.setText("2");
        jLabel19.setVisible(false);
        jTextArea10.setVisible(false);
    }

    private void jButton8MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/max22.JPG"))); // NOI18N
        jLabel7.setVisible(true);
        jToolBar1.setVisible(false);
        jLabel17.setText("3");
        jLabel19.setVisible(false);
        jTextArea10.setVisible(false);
    }

    private void jButton9MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/avg2.JPG"))); // NOI18N
        jLabel7.setVisible(true);
        jToolBar1.setVisible(false);
        jLabel17.setText("4");
```

```
    jLabel19.setVisible(false);
    jTextArea10.setVisible(false);
}

private void jButton10MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/sum2.JPG"))); // NOI18N
    jLabel7.setVisible(true);
    jToolBar1.setVisible(false);
    jLabel17.setText("5");
    jLabel19.setVisible(false);
    jTextArea10.setVisible(false);
}

private void jButton11MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel7.setVisible(false);
    jLabel19.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and2.JPG"))); // NOI18N
    jLabel19.setVisible(true);
    jToolBar1.setVisible(false);
    jLabel17.setText("6");
    jTextArea10.setVisible(true);
}

private void jButton12MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel7.setVisible(false);
    jLabel19.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or2.JPG"))); // NOI18N
    jLabel19.setVisible(true);
    jToolBar1.setVisible(false);
    jLabel17.setText("7");
    jTextArea10.setVisible(true);
}

private void jButton13MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel19.setVisible(false);
    jLabel7.setIcon(new javax.swing.ImageIcon(getClass().getResource("/if else2.JPG"))); // NOI18N
    jLabel7.setVisible(true);
    jToolBar1.setVisible(false);
    jLabel17.setText("8");
    jTextArea10.setVisible(true);
}

private void jButton15MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/count22.JPG"))); // NOI18N
    jLabel6.setVisible(true);
    jToolBar3.setVisible(false);
    jLabel16.setText("1");
    jLabel18.setVisible(false);
    jTextArea9.setVisible(false);
}

private void jButton16MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/min2.JPG"))); // NOI18N
    jLabel6.setVisible(true);
    jToolBar3.setVisible(false);
    jLabel16.setText("2");
    jLabel18.setVisible(false);
    jTextArea9.setVisible(false);
}

private void jButton17MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/max22.JPG"))); // NOI18N
    jLabel6.setVisible(true);
    jToolBar3.setVisible(false);
    jLabel16.setText("3");
    jLabel18.setVisible(false);
    jTextArea9.setVisible(false);
}

private void jButton18MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/avg2.JPG"))); // NOI18N
    jLabel6.setVisible(true);
    jToolBar3.setVisible(false);
    jLabel16.setText("4");
    jLabel18.setVisible(false);
    jTextArea9.setVisible(false);
}

private void jButton19MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/sum2.JPG"))); // NOI18N
    jLabel6.setVisible(true);
    jToolBar3.setVisible(false);
    jLabel16.setText("5");
    jLabel18.setVisible(false);
    jTextArea9.setVisible(false);
}

private void jButton20MouseClicked(java.awt.event.MouseEvent evt) {
```

```
        jLabel6.setVisible(false);
        jLabel18.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and2.JPG"))); // NOI18N
        jLabel18.setVisible(true);
        jToolBar3.setVisible(false);
        jLabel16.setText("6");
        jTextArea9.setVisible(true);
    }

    private void jButton21MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel6.setVisible(false);
        jLabel18.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or2.JPG"))); // NOI18N
        jLabel18.setVisible(true);
        jToolBar3.setVisible(false);
        jLabel16.setText("7");
        jTextArea9.setVisible(true);
    }

    private void jButton22MouseClicked(java.awt.event.MouseEvent evt) {
        jLabel18.setVisible(false);
        jLabel6.setIcon(new javax.swing.ImageIcon(getClass().getResource("/if else2.JPG"))); // NOI18N
        jLabel6.setVisible(true);
        jToolBar3.setVisible(false);
        jLabel16.setText("8");
        jTextArea9.setVisible(true);
    }

    private void jTextArea1MouseMoved(java.awt.event.MouseEvent evt) {
      jLabel15.setVisible(true);
    }

    private void jTextArea1KeyTyped(java.awt.event.KeyEvent evt) {
      jLabel15.setVisible(false);
      jLabel8.setVisible(true);
    }

    private void jTextArea2KeyTyped(java.awt.event.KeyEvent evt) {
        jLabel11.setVisible(true);
        jLabel12.setVisible(true);
        jTextArea7.setVisible(true);
        jLabel20.setVisible(false);
    }

    private void jTextArea3KeyTyped(java.awt.event.KeyEvent evt) {
        jLabel13.setVisible(true);
        jLabel14.setVisible(true);
        jTextArea8.setVisible(true);
        jLabel22.setVisible(false);
        jToolBar4.setVisible(true);
    }

    private void jTextArea4KeyTyped(java.awt.event.KeyEvent evt) {

        if(jLabel16.getText()=="6" || jLabel16.getText()=="7" || jLabel16.getText()=="8")
            jLabel6.setVisible(false);

        else
        jLabel6.setVisible(true);

        jLabel26.setVisible(false);

    }

    private void jTextArea5KeyTyped(java.awt.event.KeyEvent evt) {

        if(jLabel17.getText()=="6" || jLabel16.getText()=="7" || jLabel16.getText()=="8")
            jLabel7.setVisible(false);

        else
        jLabel7.setVisible(true);
        jLabel25.setVisible(false);
    }

    private void jButton14MouseClicked(java.awt.event.MouseEvent evt) {

try {
            output = new BufferedWriter(new FileWriter(file));
        if(jLabel1.isVisible()==true){

            output.write("for $");
            selectString="xquery";
            selectString=selectString+'\n';
            selectString=selectString+"for $";
            jTextArea1.write(output);
            selectString=selectString+jTextArea1.getText();
            output.write(" in db2-fn:xmlcolumn");
            selectString=selectString+" in db2-fn:xmlcolumn";
```

```
    jTextArea6.write(output);
    selectString=selectString+jTextArea6.getText();
    output.write('\n');
    selectString=selectString+'\n';

}

if(jLabel3.isVisible()==true){
    output.write("let $");
    selectString=selectString+"let $";
    jTextArea2.write(output);
    selectString=selectString+jTextArea2.getText();
    output.write(" := ");
    selectString=selectString+" := ";
    jTextArea7.write(output);
    selectString=selectString+jTextArea7.getText();
    output.write('\n');
    selectString=selectString+'\n';

}

if(jLabel2.isVisible()==true){
    output.write("where $");
    selectString=selectString+"where $";
    jTextArea3.write(output);
    selectString=selectString+jTextArea3.getText();

    if(jLabel24.getText() == "1"){
    output.write(" = ");
    selectString=selectString+" = ";
    }

    else if(jLabel24.getText() == "2"){
    output.write(" > ");
    selectString=selectString+" > ";
    }

    else if(jLabel24.getText() == "3"){
    output.write(" < ");
    selectString=selectString+" < ";
    }

    else if(jLabel24.getText() == "4"){
    output.write(" >= ");
    selectString=selectString+" >= ";
    }

    else if(jLabel24.getText() == "5"){
    output.write(" =< ");
    selectString=selectString+" =< ";
   }

    else {
    output.write(" != ");
    selectString=selectString+" != ";
    }

    jTextArea8.write(output);
    selectString=selectString+jTextArea8.getText();

    if (jLabel28.getText() == "1"){

    output.write(" and ");
    selectString=selectString+" and ";
    jTextArea11.write(output);
    selectString=selectString+jTextArea11.getText();
    }

    else if (jLabel28.getText() == "2"){

    output.write(" or ");
    selectString=selectString+" or ";
    jTextArea11.write(output);
    selectString=selectString+jTextArea11.getText();
    }

    output.write('\n');
    selectString=selectString+'\n';


}

if(jLabel4.isVisible()==true){
        output.write("order by ");
        selectString=selectString+"order by ";
```

```
if(jLabel16.getText() == "1"){
    output.write("count ");
    selectString=selectString+"count ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "2"){
    output.write("min ");
    selectString=selectString+"min ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "3"){
    output.write("max ");
    selectString=selectString+"max ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "4"){
    output.write("avg ");
    selectString=selectString+"avg ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "5"){
    output.write("sum ");
    selectString=selectString+"sum ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "6"){
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write(" and ");
    selectString=selectString+" and ";
    jTextArea9.write(output);
    selectString=selectString+jTextArea9.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "7"){
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write(" or ");
    selectString=selectString+" or ";
    jTextArea9.write(output);
    selectString=selectString+jTextArea9.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else if (jLabel16.getText() == "8"){
    output.write("if ");
    selectString=selectString+"if ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    jTextArea9.write(output);
    selectString=selectString+jTextArea9.getText();
    output.write('\n');
    selectString=selectString+'\n';
}

else {
    output.write("$ ");
    selectString=selectString+"$ ";
    jTextArea4.write(output);
    selectString=selectString+jTextArea4.getText();
    output.write('\n');
    selectString=selectString+'\n';
}
```

```
}

    if(jLabel5.isVisible()==true){
        output.write("return ");
        selectString=selectString+"return ";

    if(jLabel17.getText() == "1"){
        output.write("count ");
        selectString=selectString+"count ";
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "2"){
        output.write("min ");
        selectString=selectString+"min ";
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "3"){
        output.write("max ");
        selectString=selectString+"max ";
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "4"){
        output.write("avg ");
        selectString=selectString+'\n';
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "5"){
        output.write("sum ");
        selectString=selectString+"sum ";
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "6"){
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write(" and ");
        selectString=selectString+" and ";
        jTextArea10.write(output);
        selectString=selectString+jTextArea10.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "7"){
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        output.write(" or ");
        selectString=selectString+" or ";
        jTextArea10.write(output);
        selectString=selectString+jTextArea10.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }

    else if (jLabel17.getText() == "8"){
        output.write("if ");
        selectString=selectString+"if ";
        jTextArea5.write(output);
        selectString=selectString+jTextArea5.getText();
        jTextArea10.write(output);
        selectString=selectString+jTextArea10.getText();
        output.write('\n');
        selectString=selectString+'\n';
    }
```

```
        else {
            output.write("$");
            selectString=selectString+"$";
            jTextArea5.write(output);
            selectString=selectString+jTextArea5.getText();
            output.write('\n');

        }

    }

        output.close();

        }
        catch (IOException ex) {
        Logger.getLogger(ptx.class.getName()).log(Level.SEVERE, null, ex);

    }

    try {
            Desktop.getDesktop().open(file);
          }
          catch(Exception exception)  {
            System.out.println("Problem occour when to open the file");
          }


}

private void jTextArea2MouseMoved(java.awt.event.MouseEvent evt) {
    jLabel20.setVisible(true);
}

private void jTextArea7KeyTyped(java.awt.event.KeyEvent evt) {
    jLabel21.setVisible(false);
}

private void jTextArea7MouseMoved(java.awt.event.MouseEvent evt) {
    jLabel21.setVisible(true);
}

private void jTextArea3MouseMoved(java.awt.event.MouseEvent evt) {
    jLabel22.setVisible(true);
}

private void jButton23MouseClicked(java.awt.event.MouseEvent evt) {
   jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/isotita.JPG")));
   jLabel24.setText("1");
}

private void jButton24MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/megalitero.JPG")));
    jLabel24.setText("2");
}

private void jButton25MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/mikrotero.JPG")));
    jLabel24.setText("3");
}

private void jButton26MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/megaliso.JPG")));
    jLabel24.setText("4");
}

private void jButton27MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/mikriso.JPG")));
    jLabel24.setText("5");
}

private void jButton28MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel14.setIcon(new javax.swing.ImageIcon(getClass().getResource("/diaforo.JPG")));
    jLabel24.setText("6");
}

private void jTextArea8KeyTyped(java.awt.event.KeyEvent evt) {
    jLabel23.setVisible(false);
}

private void jTextArea8MouseMoved(java.awt.event.MouseEvent evt) {
    jLabel23.setVisible(true);
}

private void jTextArea5MouseMoved(java.awt.event.MouseEvent evt) {
   jLabel25.setVisible(true);
}
```

```
private void jTextArea4MouseMoved(java.awt.event.MouseEvent evt) {
   jLabel26.setVisible(true);
}

private void jButton29MouseClicked(java.awt.event.MouseEvent evt) {

    selectString2=selectString;
    try
{
        try {
            Class.forName(driverClassName);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(ptx.class.getName()).log(Level.SEVERE, null, ex);
        }
    dbConnection = DriverManager.getConnection (url, username, passwd);
     statement=dbConnection.createStatement();

    rs = statement.executeQuery(selectString2);

    while(rs.next()) {
        com.ibm.db2.jcc.DB2Xml xml = (com.ibm.db2.jcc.DB2Xml) rs.getObject(1);

        System.out.println (xml.getDB2XmlString());

        j=j+xml.getDB2XmlString()+'\n';

            try {
                output2 = new BufferedWriter(new FileWriter(file2));

                output2.write(j);

                output2.close();
            } catch (IOException ex) {
                Logger.getLogger(ptx.class.getName()).log(Level.SEVERE, null, ex);
            }

    }



    statement.close();
    dbConnection.close();

    }

 catch (SQLException ex) {
            Logger.getLogger(ptx.class.getName()).log(Level.SEVERE, null, ex);
        }

    try {
                Desktop.getDesktop().open(file2);
            }
            catch(Exception exception)  {
                System.out.println("Problem occour when to open the file");
            }
}

private void jTextArea6MouseMoved(java.awt.event.MouseEvent evt) {
    jLabel9.setVisible(true);
    jButton30.setVisible(true);
}

private void jTextArea6KeyTyped(java.awt.event.KeyEvent evt) {
    jLabel9.setVisible(false);
    jButton30.setVisible(false);
}

private void jButton30MouseClicked(java.awt.event.MouseEvent evt) {
    try {
                Desktop.getDesktop().open(file3);
            }
            catch(Exception exception)  {
                System.out.println("Problem occour when to open the file");
            }
}

private void jButton30ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void jButton31MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel27.setVisible(true);
    jLabel27.setIcon(new javax.swing.ImageIcon(getClass().getResource("/and2.JPG")));
    jTextArea11.setVisible(true);
    jLabel28.setText("1");
```

```
}

private void jButton32MouseClicked(java.awt.event.MouseEvent evt) {
    jLabel27.setVisible(true);
    jLabel27.setIcon(new javax.swing.ImageIcon(getClass().getResource("/or2.JPG")));
    jTextArea11.setVisible(true);
    jLabel28.setText("2");
}

private void jTextArea11MouseMoved(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

private void jTextArea11KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
}


/**
* @param args the command line arguments
*/
public static void main(String args[]) throws ClassNotFoundException, SQLException {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new ptx().setVisible(true);
        }
    });



}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton10;
private javax.swing.JButton jButton11;
private javax.swing.JButton jButton12;
private javax.swing.JButton jButton13;
private javax.swing.JButton jButton14;
private javax.swing.JButton jButton15;
private javax.swing.JButton jButton16;
private javax.swing.JButton jButton17;
private javax.swing.JButton jButton18;
private javax.swing.JButton jButton19;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton20;
private javax.swing.JButton jButton21;
private javax.swing.JButton jButton22;
private javax.swing.JButton jButton23;
private javax.swing.JButton jButton24;
private javax.swing.JButton jButton25;
private javax.swing.JButton jButton26;
private javax.swing.JButton jButton27;
private javax.swing.JButton jButton28;
private javax.swing.JButton jButton29;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton30;
private javax.swing.JButton jButton31;
private javax.swing.JButton jButton32;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton6;
private javax.swing.JButton jButton7;
private javax.swing.JButton jButton8;
private javax.swing.JButton jButton9;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel14;
private javax.swing.JLabel jLabel15;
private javax.swing.JLabel jLabel16;
private javax.swing.JLabel jLabel17;
private javax.swing.JLabel jLabel18;
private javax.swing.JLabel jLabel19;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel20;
private javax.swing.JLabel jLabel21;
private javax.swing.JLabel jLabel22;
private javax.swing.JLabel jLabel23;
private javax.swing.JLabel jLabel24;
private javax.swing.JLabel jLabel25;
private javax.swing.JLabel jLabel26;
private javax.swing.JLabel jLabel27;
private javax.swing.JLabel jLabel28;
```

```
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JScrollPane jScrollPane4;
    private javax.swing.JScrollPane jScrollPane5;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea10;
    private javax.swing.JTextArea jTextArea11;
    private javax.swing.JTextArea jTextArea2;
    private javax.swing.JTextArea jTextArea3;
    private javax.swing.JTextArea jTextArea4;
    private javax.swing.JTextArea jTextArea5;
    private javax.swing.JTextArea jTextArea6;
    private javax.swing.JTextArea jTextArea7;
    private javax.swing.JTextArea jTextArea8;
    private javax.swing.JTextArea jTextArea9;
    private javax.swing.JToolBar jToolBar1;
    private javax.swing.JToolBar jToolBar2;
    private javax.swing.JToolBar jToolBar3;
    private javax.swing.JToolBar jToolBar4;
    // End of variables declaration

}
```