



ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ  
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ  
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

### ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

« Πληροφοριακό σύστημα για αλυσίδα σούπερ μάρκετ σε XML »

« Information system for supermarket sequence in XML »

Γρυπάρη Μαργαρίτα

Καλαμάτα Νίκη

A.M. 2467

A.M. 2641



Επόπτης καθηγητής : Παπαδάκης Νικόλαος

Επιτροπή παρουσίασης : Παπαδάκης Νικόλαος

Ρουσάκης Ιωάννης

Τζαγκαράκης Χαράλαμπος

Έκδοση 1.0

Μάρτιος, 2015

## Ευχαριστίες

Αρχικά, θα θέλαμε να ευχαριστήσουμε τον επόπτη καθηγητή κ. Παπαδάκη Νίκο γιατί χάρη στο θέμα που μας ανέθεσε, μας βοήθησε να διευρύνουμε τους γνωστικούς μας ορίζοντες.

Ένα μεγάλο ευχαριστώ στη Μαρία για την πολύτιμη βοήθεια και καθοδήγησή της.

Τέλος, ευχαριστούμε θερμά όσους ήταν δίπλα μας όλο αυτό το διάστημα, για την κατανόηση και τη στήριξή τους.

## Abstract

In this thesis we developed an information system for a Super Market chain. We started collecting the requirements which should be fulfilled by the system and the information which must be stored in the database. We developed the relational schema based on these requirements and at the next step we used a set of specific mapping rules to convert the relational schema to an XML Schema. We continued loading our XML Schema into exist-db which is an open source database dedicated only to XML data. For data manipulation like delete, insert or update we used the querying language XQuery. Beyond simple querying structures, XQuery supports also the design of functions which we used to build the whole functionality for the database.

## Σύνοψη

Στην παρούσα πτυχιακή εργασία αναπτύξαμε ένα πληροφοριακό σύστημα με σκοπό να χρησιμοποιηθεί σε αλυσίδα Super Market. Αρχικά, καταγράψαμε τις απαιτήσεις που πρέπει να ικανοποιεί ένα τέτοιο σύστημα και τις πληροφορίες που πρέπει να αποθηκεύονται στη βάση δεδομένων. Αναπτύξαμε το σχεσιακό σχήμα ώστε να ικανοποιεί την ανάλυση απαιτήσεων και στη συνέχεια το απεικονίσαμε σε XML Schema ακολουθώντας συγκεκριμένους κανόνες μετατροπής (mapping rules). Στη συνέχεια, συνδέσαμε το XML Schema με την exist-db η οποία είναι μία ανοιχτού λογισμικού βάση δεδομένων για XML δεδομένα. Για την τροποποίηση των δεδομένων που είναι αποθηκευμένα σε XML, είτε αυτό σημαίνει διαγραφή είτε προσθήκη εγγραφής είτε τροποποίηση σε ήδη υπάρχουσες εγγραφές, χρησιμοποιήσαμε τη γλώσσα επερώτησης XQuery κατάλληλη για XML δεδομένα. Πέρα από απλές δομές επερωτήσεων, η XQuery υποστηρίζει και την σχεδίαση συναρτήσεων τις οποίες χρησιμοποιήσαμε για να χτίσουμε την λειτουργικότητα της βάσης δεδομένων.

## Πίνακας Περιεχομένων

1	Εισαγωγή.....	9
2	Ιστορική αναδρομή.....	10
3	Εισαγωγή στην XML.....	12
4	Περιγραφή των δεδομένων και των απαιτήσεων.....	14
4.1	Σχεδίαση E-R diagram.....	15
5	XML Schema.....	18
5.1	Βασικοί κανόνες σύνταξης για XML Schema.....	18
5.2	Κανόνες μετατροπής E-R Diagram σε XML Schema.....	19
5.3	Ανάλυση XML Schema του Super Market.....	22
5.4	eXist-db.....	28
6	XQuery.....	29
6.1	XQuery προσθήκης στοιχείου (element).....	30
6.2	XQuery διαγραφής στοιχείου (element).....	56
6.3	XQuery ανανέωσης στοιχείου (element).....	60
7	Αυτόματες παραγγελίες μέσω XQuery.....	62
8	Επεκτάσεις.....	68
8.1	Εκκίνηση δημιουργίας μια νέας εφαρμογής.....	68
8.2	Περιγραφή της διαδικτυακής εφαρμογής.....	70
9	Βιβλιογραφία.....	73
10	Παράρτημα Ν° 1.....	74
11	Παράρτημα Ν° 2.....	77
11.1	Βήματα εγκατάστασης της eXist-db.....	77
11.2	Σύνδεση της eXist-db με το περιβάλλον ανάπτυξης oxygen developer.....	87
12	Παράρτημα Νο 3.....	91

**Πίνακας Εικόνων**

Εικόνα 3.1: Δενδροειδή αναπαράσταση εγγράφων.....	13
Εικόνα 4.1: Οντότητες & χαρακτηριστικά.....	15
Εικόνα 4.2: Diagram of Super Market .....	16
Εικόνα 5.1: Περιγραφή δομής element τηλεφώνου & διεύθυνσης.....	23
Εικόνα 5.2: Περιγραφή τύπου 'proionLista' και 'proionPerigraph' .....	24
Εικόνα 5.3: Περιγραφή τύπου 'promhtheytesLista' και 'promhtheythsPerigraph' .....	25
Εικόνα 5.4: Στοιχεία(elements) του Super Market.....	25
Εικόνα 5.5: Element αποθήκη.....	26
Εικόνα 5.6: Στοιχείο(element) 'ypokatasthma' .....	26
Εικόνα 5.7: Στοιχείο(element) 'ypallhlos' .....	27
Εικόνα 5.8: Στοιχείο(element) 'paraggelia' .....	27
Εικόνα 5.9: Στοιχείο(element) 'paraggeliaApothikis' .....	28
Εικόνα 6.1: XQuery 'createSuperMarket', συνάρτηση 'createApothiki' .....	30
Εικόνα 6.2: XQuery createSuperMarket, συνάρτηση insertApothiki .....	34
Εικόνα 6.3: Return of XQuery createSuperMarket .....	38
Εικόνα 6.4: Xquery 'addApothiki' .....	39
Εικόνα 6.5: XQuery addProionApothikis .....	52
Εικόνα 6.6: XQuery ddProionYpokatasthmatos .....	53
Εικόνα 6.7: XQuery 'addPromhtheythApothikis' .....	55
Εικόνα 6.8: XQuery 'deleteApothiki' .....	57
Εικόνα 6.9: XQuery'deleteParaggeliaApothikis' .....	58
Εικόνα 6.10: XQuery 'deleteParaggeliasYpokatasthmatos'.....	58
Εικόνα 6.11: XQuery 'deleteProion' .....	58
Εικόνα 6.12: XQuery 'deleteProionApothikis' .....	59
Εικόνα 6.13: XQuery 'deleteProionYpokatasthmatos' .....	59
Εικόνα 6.14: XQuery 'deletePromhtheyth' .....	59
Εικόνα 6.15: XQuery 'deletePromhtheythApothikis' .....	60
Εικόνα 6.16: XQuery 'deleteYpallhloYpokatasthmatos' .....	60
Εικόνα 6.17: XQuery 'deleteYpokatasthmaApothikis' .....	60
Εικόνα 6.18: XQuery 'updateApothemaApothikis' .....	61
Εικόνα 6.19: XQuery 'updateApothemaYpokatasthmatos' .....	61
Εικόνα 7.1: Συνάρτηση 'checkApothemaYpokatasthmatos' .....	62
Εικόνα 7.2: Συνάρτηση 'between-inclusive' .....	63
Εικόνα 7.3: Συνάρτηση 'makeUpdatesYpokatasthma' .....	63
Εικόνα 7.4: Συνάρτηση 'updateApothemaYpokatasthmatos' .....	64
Εικόνα 7.5: Συνάρτηση 'insertParaggeliaYpokatasthmatos' .....	64
Εικόνα 7.6: Συνάρτηση 'createParaggelia' .....	64
Εικόνα 7.7: Συνάρτηση 'updateApothemaApothikis' .....	65
Εικόνα 7.8: Συνάρτηση 'checkApothemaApothikis' .....	65
Εικόνα 7.9: Συνάρτηση 'makeUpdatesApothiki' .....	66
Εικόνα 7.10: Συνάρτηση 'prosfora' .....	66
Εικόνα 7.11: Συνάρτηση 'insertParaggeliaApothikis' .....	67
Εικόνα 7.12: Συνάρτηση 'createParaggeliaApothikis' .....	67
Εικόνα 7.13: Συνάρτηση 'fullApothiki' .....	67
Εικόνα 8.1: Τοπικό URL.....	68
Εικόνα 8.2: eXide, Deployment editor.....	69

Εικόνα 8.3: Application Properties .....	70
Εικόνα 8.4: Δομή φακέλων στο oXygen μετά από τα βήματα για δημιουργία της αρχικής εφαρμογής .....	71
Εικόνα 8.5: Αρχική σελίδα της εφαρμογής.....	71
Εικόνα 11.1: Εγκατάσταση της eXist-db, βήμα 1 .....	77
Εικόνα 11.2: Εγκατάσταση της java, βήμα 1 .....	78
Εικόνα 11.3: Εγκατάσταση της java, βήμα 2 .....	79
Εικόνα 11.4: Εγκατάσταση της java, βήμα 3 .....	79
Εικόνα 11.5: Εγκατάσταση της java, βήμα 4 .....	80
Εικόνα 11.6: Εγκατάσταση της java, βήμα 5 .....	80
Εικόνα 11.7: Εγκατάσταση της eXist-db, βήμα 2 .....	81
Εικόνα 11.8: Εγκατάσταση της eXist-db, βήμα 3 .....	81
Εικόνα 11.9: Εγκατάσταση της eXist-db, βήμα 4 .....	82
Εικόνα 11.10: Εγκατάσταση της eXist-db, βήμα 5 .....	82
Εικόνα 11.11: Εγκατάσταση της eXist-db, βήμα 6 .....	83
Εικόνα 11.12: Εγκατάσταση της eXist-db, βήμα 7 .....	83
Εικόνα 11.13: Εγκατάσταση της eXist-db, βήμα 8 .....	84
Εικόνα 11.14: Εγκατάσταση της eXist-db, βήμα 9 .....	84
Εικόνα 11.15: Εγκατάσταση της eXist-db, βήμα 10 .....	85
Εικόνα 11.16: Εγκατάσταση της eXist-db, βήμα 11 .....	85
Εικόνα 11.17: Εγκατάσταση της eXist-db, βήμα 12 .....	86
Εικόνα 11.18: Εγκατάσταση της eXist-db, βήμα 13 .....	86
Εικόνα 11.19: Εγκατάσταση της eXist-db, βήμα 14 .....	87
Εικόνα 11.20: Ένωση eXist-db με oXygen, βήμα 1 .....	87
Εικόνα 11.21: Ένωση eXist-db με oXygen, βήμα 2 .....	88
Εικόνα 11.22: Ένωση eXist-db με oXygen, βήμα 3 .....	88
Εικόνα 11.23: Ένωση eXist-db και oXygen, βήμα 4 .....	89
Εικόνα 11.24: Ένωση eXist-db και oXygen, βήμα 5 .....	89
Εικόνα 11.25: Ένωση eXist-db και oXygen, βήμα 6 .....	90
Εικόνα 12.1: Παρουσίαση, slide 1 .....	91
Εικόνα 12.2: Παρουσίαση, slide 2 .....	91
Εικόνα 12.3: Παρουσίαση, slide 3 .....	92
Εικόνα 12.4: Παρουσίαση, slide 4 .....	92
Εικόνα 12.5: Παρουσίαση, slide 5 .....	93
Εικόνα 12.6: Παρουσίαση, slide 6 .....	93
Εικόνα 12.7: Παρουσίαση, slide 7 .....	94
Εικόνα 12.8: Παρουσίαση, slide 8 .....	94
Εικόνα 12.9: Παρουσίαση, slide 9 .....	95
Εικόνα 12.10: Παρουσίαση, slide 10 .....	95
Εικόνα 12.11: Παρουσίαση, slide 11 .....	96
Εικόνα 12.12: Παρουσίαση, slide 12 .....	96
Εικόνα 12.13: Παρουσίαση, slide 13 .....	97
Εικόνα 12.14: Παρουσίαση, slide 14 .....	97
Εικόνα 12.15: Παρουσίαση, slide 15 .....	98
Εικόνα 12.16: Παρουσίαση, slide 16 .....	98
Εικόνα 12.17: Παρουσίαση, slide 17 .....	99
Εικόνα 12.18: Παρουσίαση, slide 18 .....	99
Εικόνα 12.19: Παρουσίαση, slide 19 .....	100

**Πίνακας Πινάκων**

Πίνακας 4.1: Super Market's Elements & Attributes .....	14
Πίνακας 4.2: Σχέσεις μεταξύ οντοτήτων Super Market .....	14
Πίνακας 4.3: Διαδικασίες για υλοποίηση .....	15
Πίνακας 4.4: Οντότητες του Super Market και τα αντίστοιχα κλειδιά .....	16
Πίνακας 5.1: Ετικέτες(tags) αλληλουχίας στοιχείων(elements) .....	18
Πίνακας 6.1: XQuery 'createuperMarket', συναρτήσεις δημιουργίας στοιχείων(elements) .....	31
Πίνακας 6.2: XQuery 'createSuperMarket', συναρτήσεις εισαγωγής.....	36
Πίνακας 6.3: XQuery 'addApothiki', συναρτήσεις.....	38
Πίνακας 6.4: XQuery 'addCompleteApothiki', συναρτήσεις .....	41
Πίνακας 6.5: XQuery 'addYpokatasthma', συναρτήσεις .....	45
Πίνακας 6.6: XQuery 'addCompleteYpokatasthma', συναρτήσεις.....	47
Πίνακας 6.7: XQuery 'addParaggeliaYpokatasthmatos', συναρτήσεις.....	49
Πίνακας 6.8: XQuery 'addParaggeliaApothikis', συναρτήσεις .....	50
Πίνακας 6.9: XQuery 'addProion', συναρτήσεις .....	51
Πίνακας 6.10: XQuery 'addProionApothiki', συναρτήσεις .....	52
Πίνακας 6.11: XQuery 'addProionYpokatasthmatos', συναρτήσεις.....	52
Πίνακας 6.12: XQuery 'addPromhtheyth', συναρτήσεις .....	53
Πίνακας 6.13: XQuery 'addPromhtheythApothikis', συναρτήσεις.....	54
Πίνακας 6.14: XQuery addYpallhos, συναρτήσεις.....	55
Πίνακας 6.15: XQuery διαγραφών, συναρτήσεις.....	57



## 1 Εισαγωγή

Ο τομέας της Πληροφορικής έχει πολλά παρακλάδια με τα οποία μπορεί να ασχοληθεί κανείς κι ένα από τα πιο ενδιαφέροντα κατά τη γνώμη μας είναι οι βάσεις δεδομένων(databases). Αναφερόμενοι στον όρο βάσεις δεδομένων μιλάμε για ένα σύνολο συσχετιζόμενων δεδομένων τα οποία αποθηκεύονται με έναν πολύ οργανωμένο τρόπο. Τα δεδομένα που συναντάμε σε μία βάση δεδομένων πρέπει να είναι ολοκληρωμένα(integrated) και καταμεριζόμενα(shared). Η αποθήκευσή τους δηλαδή πρέπει να γίνεται σε σύνολα αρχείων που χαρακτηρίζονται από οργάνωση και ομοιομορφία, στα οποία δεν θα συναντάμε επανάληψη των ίδιων στοιχείων, χαρακτηριστικό των ολοκληρωμένων δεδομένων. Επίσης, στα δεδομένα θα πρέπει να υπάρχει δυνατότητα προσπέλασής τους σε πάνω από έναν χρήστες την ίδια χρονική στιγμή, χαρακτηριστικό των καταμεριζόμενων δεδομένων.

Όταν λοιπόν φτάσουμε στο σημείο να έχουμε μια βάση δεδομένων(database) με τα παραπάνω χαρακτηριστικά, τότε αυτό που χρειαζόμαστε είναι ένας τρόπος για να τη διαχειριστούμε. Λέγοντας διαχείριση εννοούμε τον ορισμό των δομών για την αποθήκευση των πληροφοριών αλλά και την παροχή μηχανισμών για το χειρισμό τους. Το σύνολο των προγραμμάτων που μας προσφέρουν τη δυνατότητα να διαχειριστούμε μία, ή παραπάνω βάσεις δεδομένων που ανήκουν στο ίδιο σύστημα, ονομάζεται σύστημα διαχείρισης βάσεων δεδομένων(databases management system, DBMS). Ένα σύστημα διαχείρισης βάσεων δεδομένων στοχεύει στο να μας παρέχει έναν εύκολο και αποτελεσματικό τρόπο αποθήκευσης και ανάκτησης πληροφοριών από τη Βάση δεδομένων, πράγμα που μας βοηθάει ιδιαίτερα όταν ο όγκος των δεδομένων που έχουμε να διαχειριστούμε είναι μεγάλος. Ένα σημαντικό κομμάτι επίσης που πρέπει να μας παρέχουν τα συστήματα διαχείρισης βάσεων δεδομένων είναι η ασφάλεια των πληροφοριών που αποθηκεύουμε σε αυτά.

Οι βάσεις δεδομένων χρησιμοποιούνται στην καθημερινότητά μας για πολλές εφαρμογές. Παραδείγματα τέτοιων εφαρμογών είναι αρχικά οι τράπεζες που διαχειρίζονται τραπεζικές συναλλαγές, πελατολόγιο κ.λπ. Επίσης, βάσεις δεδομένων χρησιμοποιούν οι αεροπορικές εταιρίες οι οποίες αποθηκεύουν όλες τις κρατήσεις των πελατών και τις πτήσεις. Άλλη εφαρμογή είναι τα πανεπιστήμια που πρέπει να κρατάνε μεγάλο όγκο δεδομένων όπως τα στοιχεία των φοιτητών, τα μαθήματα και τους βαθμούς τους. Αποδεικνύεται λοιπόν ότι οι βάσεις δεδομένων αποτελούν πλέον αναπόσπαστο κομμάτι των επιχειρήσεων. Αντίστοιχα, μια τέτοια επιχείρηση είναι και το Super Market, η οποία χρειάζεται οπωσδήποτε μια βάση δεδομένων για την αποθήκευση και την ανάκληση όλων των στοιχείων από τα οποία αποτελείται και με αυτό το σκοπό έγινε η συγκεκριμένη πτυχιακή εργασία.

Θα ξεκινήσουμε με μία ιστορική αναδρομή στο κεφάλαιο 2 για να περιγράψουμε την εξέλιξη των βάσεων δεδομένων. Θα συνεχίσουμε στο κεφάλαιο 3 παραθέτοντας κάποια βασικά στοιχεία για την XML. Μετά από αυτό, περνάμε στο κεφάλαιο 4 με μία εκτενέστερη περιγραφή των απαιτήσεων της πτυχιακής μας εργασίας καθώς και με την παρουσίαση του διαγράμματος οντοτήτων-σχέσεων(e-r diagram) που προκύπτει από αυτήν. Στο κεφάλαιο 5, αφού πούμε κάποιες βασικές έννοιες για το xml schema, θα περάσουμε στην μετατροπή του διαγράμματος οντοτήτων-σχέσεων σε xml schema. Στο κεφάλαιο 6 κάνουμε μία αναφορά στην βάση δεδομένων που χρησιμοποιήσαμε, την eXist-db. Ακολουθεί η εισαγωγή στα XQuery στο κεφάλαιο 7 και η ανάλυση των συγκεκριμένων XQuery που χρησιμοποιήσαμε προκειμένου να υλοποιήσουμε τις διαδικασίες που μας ζητήθηκαν. Χωριστό κεφάλαιο αποτελεί ο μηχανισμός αυτόματων παραγγελιών με μηχανισμό επερωτήσεων για τον οποίο μιλάμε στο κεφάλαιο 8. Τις πιθανές μελλοντικές επεκτάσεις της εργασίας μας θα αναφέρουμε στο κεφάλαιο 9 και στα κεφάλαια 10 και 11 θα παραθέσουμε την βιβλιογραφία και τα παραρτήματα αντίστοιχα.

## 2 Ιστορική αναδρομή

Ας γυρίσουμε λίγο το χρόνο πίσω για να δούμε πώς η εξέλιξη της χρήσης των βάσεων δεδομένων(databases) έχει φτάσει στο σημείο που τη συναντάμε σήμερα. Αν τοποθετηθούμε γύρω στο 1950 με αρχές του 1960, θα δούμε ότι τα δεδομένα αποθηκεύονταν σε μαγνητικές ταινίες οι οποίες επέτρεπαν μόνο τη σειριακή προσπέλασή τους. Ο αποθηκευτικός τους χώρος όμως υπολογίζεται 10.000 φορές μεγαλύτερος από το μέχρι τότε μέσο αποθήκευσης, τη διάτρητη κάρτα. Το γεγονός αυτό ήταν ικανό για να καθιερώσει τις μαγνητικές ταινίες ως το πιο δημοφιλή τρόπο αποθήκευσης τη δεκαετία του 1980. Παρόλα αυτά, στα τέλη του 1960 με 1970 ήρθαν στο προσκήνιο και οι σκληροί δίσκοι που μας προσέφεραν πλέον τη δυνατότητα άμεσης προσπέλασης των δεδομένων. Εμφανίζονται επιπλέον τα δικτυωτά μοντέλα δεδομένων(network data models) και τα ιεραρχικά μοντέλα δεδομένων (hierarchical data models) με τον Ted Codd να ορίζει το σχεσιακό μοντέλο δεδομένων(relational data model).

Το ιεραρχικό μοντέλο δεδομένων ήταν το πρώτο που εμφανίστηκε, το οποίο είχε μια ιεραρχική δομή και θύμιζε δέντρο. Οι οντότητες(entities) τοποθετούνται σε επίπεδα ιεραρχίας και σχεδιάζονται σαν απολήξεις από τα κλαδιά τα οποία αναπαριστούν τις σχέσεις(relationships) μεταξύ των οντοτήτων. Από μία οντότητα(entity) υψηλού επιπέδου ξεκινάνε πολλά κλαδιά τα οποία καταλήγουν σε άλλες οντότητες κατώτερου επιπέδου. Όμως, μόνο μία οντότητα υψηλού επιπέδου αντιστοιχεί σε μία οντότητα κατώτερου επιπέδου. Πλέον αυτό το μοντέλο δε χρησιμοποιείται γιατί κρίθηκε δύσχρηστο. Από την άλλη μεριά, το δικτυωτό μοντέλο δεδομένων είναι όπως το ιεραρχικό μοντέλο δεδομένων με τη διαφορά ότι σε μία οντότητα κατώτερου επιπέδου μπορούν να αντιστοιχούν πάνω από μία οντότητες ανώτερου επιπέδου.

Περνώντας στο σχεσιακό μοντέλο δεδομένων, να πούμε ότι οι βάσεις δεδομένων που σχεδιάζονται βασισμένες σε αυτό, αποκαλούνται σχεσιακές βάσεις δεδομένων(relational databases). Εδώ, η περιγραφή των βάσεων δεδομένων γίνεται με μαθηματικές έννοιες και με χρήση πίνακα. Έχουμε τόσους πίνακες όσες και οντότητες και κάθε πίνακας αποθηκεύει δεδομένα στις στήλες και στις γραμμές του για μία συγκεκριμένη οντότητα. Στις στήλες συναντάμε τα χαρακτηριστικά της εκάστοτε οντότητας και κάθε γραμμή του πίνακα αποτελεί ένα στιγμιότυπο(instance) της οντότητας. Αυτός ο τρόπος δυστυχώς παρουσιάζει μειονεκτήματα που είναι η ταχύτητα υπολογισμών και ο χώρος αποθήκευσης, αλλά αυτό αφορά μόνο την περίπτωση μεγάλου όγκου δεδομένων. Το πλεονέκτημα όμως του σχεσιακού μοντέλου δεδομένων, είναι η ευχρηστία του καθώς και η σαφήνιά του και γι' αυτό έχει επικρατήσει μέχρι σήμερα σε σχέση με τα άλλα δύο.

Προχωρώντας στο 1980, παρατηρούμε μεγάλη ανάπτυξη των εμπορικών εφαρμογών με χρήση του σχεσιακού μοντέλου δεδομένων, αλλά και την SQL(Structured Query Language), γλώσσα υπολογιστών σχεδιασμένη για διαχείριση δεδομένων στις σχεσιακές βάσεις δεδομένων, να γίνεται πλέον εργοστασιακό πρότυπο(industrial standard). Εμφανίζονται επίσης εκείνη την εποχή τα καταναμημένα συστήματα διαχείρισης βάσεων δεδομένων(distributed database Management System, DDBMS). Μια καταναμημένη βάση δεδομένων(distributed database) αποθηκεύει τα μέρη της σε διαφορετικούς υπολογιστές μέσα στο δίκτυο. Έτσι, στους χρήστες παρέχεται πρόσβαση μόνο στα δεδομένα που τους αφορούν χωρίς να επηρεάζουν την εργασία των υπολοίπων. Η δουλειά του καταναμημένου συστήματος διαχείρισης βάσεων δεδομένων λοιπόν είναι να χειρίζεται αυτή τη βάση δεδομένων σα να ήταν σε έναν υπολογιστή αποθηκευμένη. Οπότε συγχρονίζει τα δεδομένα και στην περίπτωση που πάνω από δύο χρήστες βλέπουν τη βάση δεδομένων, εγγυάται ότι από τις αλλαγές που θα κάνει ο ένας θα ενημερωθεί η βάση δεδομένων, όπου αλλού κι αν βρίσκεται αποθηκευμένη.

Την ίδια περίοδο εμφανίστηκαν και τα αντικειμενοστραφή συστήματα διαχείρισης βάσεων δεδομένων(object-oriented database management system) που ήρθαν για να καλύψουν τα κενά που άφηνε το σχεσιακό μοντέλο δεδομένων, όπως οι μεγάλες απαιτήσεις που έχουν οι σύγχρονες εφαρμογές, π.χ. εφαρμογές πολυμέσων. Οι εφαρμογές αυτές πέρα από τον μεγάλο όγκο δεδομένων και την πολυπλοκότητα που έχουν, χρειάζονται και πολύπλοκους αλγόριθμους προκειμένου να υλοποιήσουν μια διαχείριση των δεδομένων, πράγμα που δεν μπορεί να το προσφέρει το σχεσιακό μοντέλο δεδομένων. Επίσης, άλλο ένα σημείο που μας καθιστά μη εύχρηστη σε τέτοιες περιπτώσεις τη χρήση του σχεσιακού μοντέλου δεδομένων, είναι ο περιορισμός που μας δίνει η δομή του πίνακα σε σχέση με την πολυπλοκότητα των πραγματικών αντικειμένων. Αυτά κι άλλα πολλά ήρθαν να βελτιώσουν τα αντικειμενοστραφή συστήματα διαχείρισης βάσεων δεδομένων που αντιλαμβάνονται τον κόσμο ως ένα σύνολο αντικειμένων, όπου το καθένα από αυτά έχει τα δικά του χαρακτηριστικά.

Κοιτάζοντας λοιπόν το παρελθόν, παρατηρούμε πως η χρήση των βάσεων δεδομένων σημείωσε μια σημαντική άνοδο τις τελευταίες 4 δεκαετίες του 20<sup>ου</sup> αιώνα. Στην αρχή, η χρήση τους από απλούς πολίτες ήταν έμμεση, για παράδειγμα μέσω ενός ταξιδιωτικού πράκτορα που ήταν υπεύθυνος να κλείσει τα αεροπορικά τους εισιτήρια, ο οποίος φυσικά κοιτούσε μία βάση δεδομένων προκειμένου να κλείσει τα εισιτήρια. Μετά την εμφάνιση όμως των αυτοματοποιημένων μηχανημάτων, οι χρήστες έρχονταν πλέον σε άμεση επαφή με τις βάσεις δεδομένων. Παράδειγμα αποτελούν οι διασυνδέσεις τηλεφώνων με υπολογιστές στις οποίες ο χρήστης έχει τη δυνατότητα να εισάγει πληροφορίες ή να διαλέξει μεταξύ κάποιων επιλογών, πατώντας κάποιο πλήκτρο ή κάποιο συνδυασμό τους.

Σημαντικό κεφάλαιο στην άμεση πρόσβαση των βάσεων δεδομένων ήταν το τέλος της δεκαετίας του 1990 με την επανάσταση του διαδικτύου. Είχε έρθει πια η εποχή όπου οι εταιρίες είχαν υιοθετήσει πλέον τη χρήση των βάσεων δεδομένων και ένα μεγάλο μέρος των υπηρεσιών και των πληροφοριών τους ήταν διαθέσιμα στο διαδίκτυο(online). Καθημερινό και επαναλαμβανόμενο παράδειγμα αποτελεί η παραγγελία online κατά την οποία ο χρήστης ψάχνει σε μία βάση δεδομένων το προϊόν που τον ενδιαφέρει και στο τέλος η παραγγελία του καταχωρείται και αυτή σε μία βάση δεδομένων. Έτσι, αν και πολλοί δεν αντιλαμβάνονται ότι συνδιαλέγονται με μία βάση δεδομένων, αυτές έχουν κατακτήσει ένα μεγάλο μέρος της καθημερινότητάς μας. Πρόσφατη αλλά όχι τελευταία εξέλιξη στο χώρο αυτό, καταγράφηκε το 2000 με την παρουσία της XML(Extensible Markup Language) και των XQuery με τα οποία θα ασχοληθούμε αναλυτικότερα στα επόμενα κεφάλαια.

### 3 Εισαγωγή στην XML

Ζούμε σε μία εποχή όπου το μεγαλύτερο όγκο πληροφοριών που χρειαζόμαστε καθημερινά, τον αντλούμε από πηγές που βρίσκουμε ελεύθερα στο διαδίκτυο. Οπότε καταλαβαίνουμε ότι είναι αναγκαία η εύκολη πρόσβαση σε αυτό, η δυνατότητα μεταφοράς πληροφοριών και η ευελιξία τους. Όλα αυτά όμως δεν ήταν δυνατό να μας τα προσφέρει η HTML<sup>1</sup> κι έτσι δημιουργήθηκε η XML<sup>2</sup> με σκοπό να την συμπληρώσει και όχι να την αντικαταστήσει. Να προσθέσουμε επίσης πως με βάση την XML δημιουργήθηκαν κι άλλες γλώσσες όπως η XHTML<sup>3</sup>, η οποία είναι μία ανασχεδιασμένη HTML.

Πιο συγκεκριμένα, η XML είναι μια γλώσσα σήμανσης, όπως μαρτυράει και το όνομά της, η οποία περιέχει κανόνες με τους οποίους γίνεται η ηλεκτρονική κωδικοποίηση κειμένων. Προορίζεται για έγγραφα με δομημένες πληροφορίες οι οποίες απαρτίζονται από το περιεχόμενο και το ρόλο που παίζει το περιεχόμενο αυτό, με την ίδια δομή για όλα σχεδόν τα έγγραφα. Η XML προσφέρει τη δυνατότητα προσαρμογής στο στυλ του εγγράφου και στην δομή του, με μεγαλύτερη ποικιλία από αυτή που παρέχει η HTML. Επίσης, τα tags που χρησιμοποιούμε στην XML είναι ορισμένα από τον χρήστη, όπως και οι μεταξύ τους σχέσεις και όχι προκαθορισμένα.. Υποστηρίζει Unicode για όλες τις γλώσσες και παρά το γεγονός ότι σχεδιάστηκε για την μορφοποίηση δεδομένων κειμένου, χρησιμοποιείται και για αναπαράσταση δομών δεδομένων.

Αναλυτικότερα τώρα, η XML αποτελείται από δύο βασικά συστατικά, τα στοιχεία(elements) και τα γνωρίσματα(attributes). Κάθε στοιχείο(element) περικλείεται από μία ετικέτα(tag) έναρξης κι μία ετικέτα(tag) λήξης και η κάθε ετικέτα(tag) έχει μία πληροφορία, όπως για παράδειγμα :

```
<firstname> Γιώργος </firstname >
```

Τα γνωρίσματα(attributes) που έχουν τα στοιχεία(elements) ορίζονται ως εξής:

```
<firstname language="GR"> Γιώργος </firstname >
```

Τα δύο αυτά συστατικά είναι ικανά να μας αφήσουν να περιγράψουμε όποια πληροφορία θέλουμε αρκεί να ακολουθήσουμε κάποιους κανόνες:

- ✓ μέσα σε ένα στοιχείο(element) μπορούμε να έχουμε κανένα ή και περισσότερα άλλα στοιχεία(elements)
- ✓ κανένα ή περισσότερα γνωρίσματα(attributes)
- ✓ κείμενο

Ας δούμε ένα απλό παράδειγμα που αναφέρεται στην παραγγελία ενός προϊόντος:

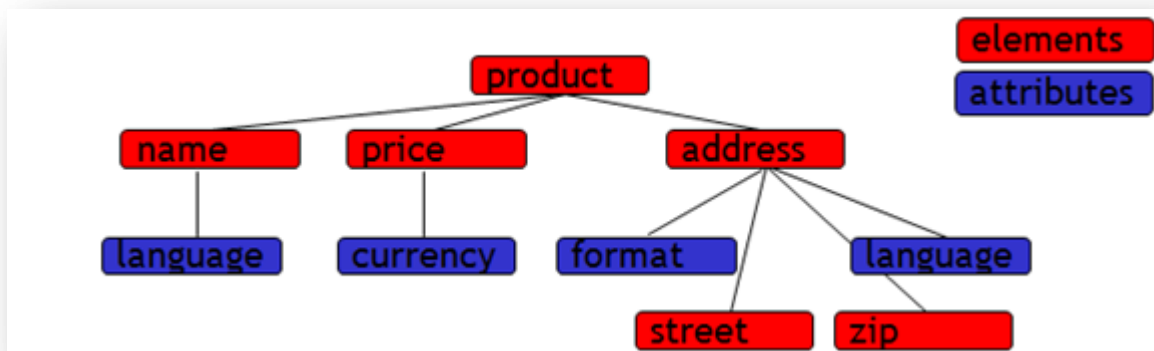
```
<product>
  <name language="Greek">Βιβλίο XML-XQuery</name>
  <price currency="Euro"> 50 </price>
  <address format="XL1245" language="Greek">
    <street>18 Δικαιοσύνης</street>
    <zip>15845</zip>
  </address>
</product>
```

Τέλος, να αναφέρουμε άλλη μία ιδιότητα της XML που αφορά την αναπαράσταση των εγγράφων σε μορφή δέντρου. Για μία τέτοια αναπαράσταση, τα στοιχεία(elements) έχουν τη μορφή κόμβου και τα γνωρίσματα(attributes) μαζί με τα στοιχεία(elements) που περιέχονται στο αρχικό στοιχείο(element), είναι υποκόμβοι του. Να σημειώσουμε ότι τα γνωρίσματα(attributes) δεν μπορούν να περιέχουν υποκόμβους και πως σε κάθε έγγραφο XML έχουμε μόνο έναν αρχικό κόμβο, κάτω από τον οποίο κρεμιούνται όλα τα υπόλοιπα. Παράδειγμα τέτοιας αναπαράστασης είναι το παρακάτω :

<sup>1</sup> Hyper Text Markup Language

<sup>2</sup> Extensible Markup Language

<sup>3</sup> EXtensible HyperText Markup Language



Εικόνα 3.1: Δενδροειδή αναπαράσταση εγγράφων

#### 4 Περιγραφή των δεδομένων και των απαιτήσεων

Σκοπός της πτυχιακής εργασίας που μας ανατέθηκε είναι η ανάπτυξη ενός πληροφοριακού συστήματος για αλυσίδα Super Market. Τα στοιχεία που μας ζητήθηκαν να κρατάμε στη βάση δεδομένων είναι τα εξής :

Στοιχεία(element)	Χαρακτηριστικά
Προϊόν	1.Όνομα 2. Είδος 3. Παραγωγός/Κατασκευαστής 4. Τιμή
Αποθήκη	1.Όνομα 2.Διεύθυνση
Υποκατάστημα	1.Όνομα 2.Διεύθυνση
Υπάλληλος	1.Ονοματεπώνυμο 2.Διεύθυνση 3.Τηλέφωνο 4.Μισθός 5.Αριθμός ταυτότητας
Προμηθευτής	1.Ονοματεπώνυμο 2.Διεύθυνση 3.Τηλέφωνο
Παραγγελία	1.Κωδικός 2.Ποσότητα

Πίνακας 4.1: Super Market's Elements & Attributes

Ενώ τα στοιχεία για τις μεταξύ τους σχέσεις που μας δίνονται είναι τα παρακάτω :

	Αποθήκη	Προϊόν	Υποκατάστημα	Υπάλληλος	Προμηθευτής
Αποθήκη		N	1		N
Προϊόν	N				(1,N)
Υποκατάστημα	N			1	
Υπάλληλος			N		
Προμηθευτής	N	(1,N)			

Πίνακας 4.2: Σχέσεις μεταξύ οντοτήτων Super Market

Στην πρώτη γραμμή και στην πρώτη στήλη του πίνακα βρίσκονται οι οντότητες του προβλήματός μας. Στην δεύτερη στήλη συναντάμε τις σχέσεις που συνδέουν την οντότητα «Αποθήκη» με κάθε μία από τις οντότητες της πρώτης στήλης κ.ο.κ. Να συμπληρώσουμε ότι θέλουμε να αποθηκεύουμε :

- ✓ το απόθεμα από τα προϊόντα που έχει η αποθήκη
- ✓ το απόθεμα από τα προϊόντα που έχει το υποκατάστημα
- ✓ το ελάχιστο απόθεμα που πρέπει να έχει η αποθήκη
- ✓ το ελάχιστο απόθεμα που πρέπει να έχει το υποκατάστημα
- ✓ τη τιμή πώλησης που έχει ο κάθε προμηθευτής για το προϊόν που μας προμηθεύει.

Αφού τελειώσουμε με το σχεδιασμό της βάσης δεδομένων, πρέπει να υλοποιήσουμε με μορφή επερωτήματος τις ακόλουθες διαδικασίες :

Διαδικασίες	Περιγραφή
Εισαγωγή κάθε οντότητας	Προσθήκη κάθε οντότητας στη βάση
Διαγραφή κάθε οντότητας	Διαγραφή κάθε οντότητας από την βάση
Εφοδιασμός	Μεταφορά προϊόντων από την αποθήκη προς το υποκατάστημα με αντίστοιχη μείωση των αποθεμάτων. Η διαδικασία αυτή ενεργοποιείται κάθε φορά που το απόθεμα σε ένα προϊόν πέσει κάτω από το ελάχιστο. Η ποσότητα που

**Διαδικασία αυτόματης παραγγελίας**

παραγγέλνει το μαγαζί είναι τόση ώστε το απόθεμα που τελικά θα έχει, να είναι το διπλάσιο του ελαχίστου.

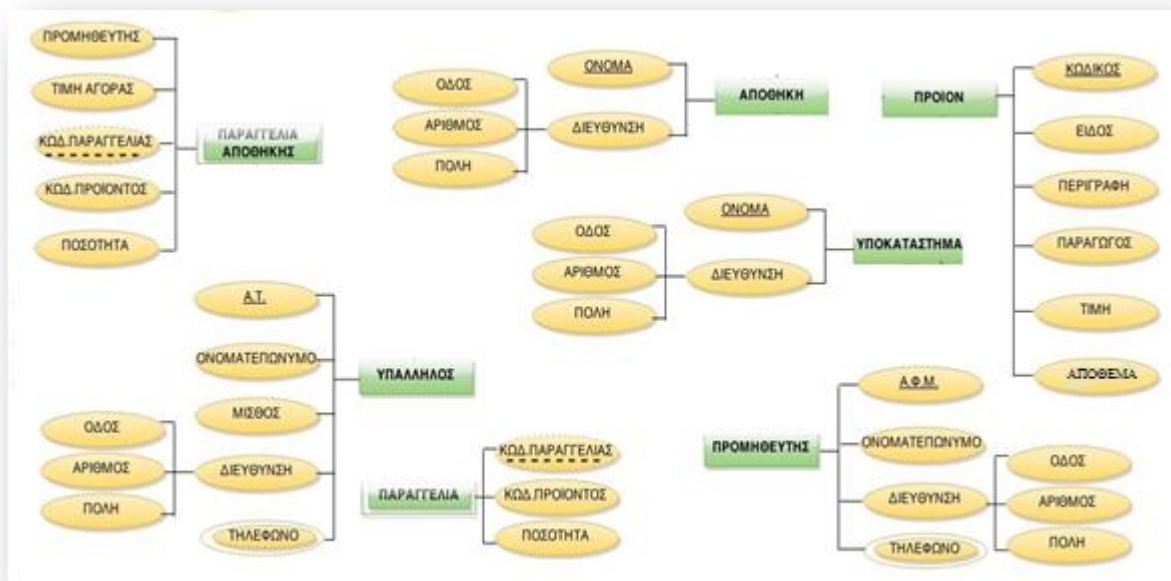
Όταν το απόθεμα σε μία αποθήκη πέσει κάτω από το ελάχιστο για ένα προϊόν, τότε γίνεται παραγγελία στον προμηθευτή με τη μικρότερη τιμή πώλησης. Η ποσότητα θα είναι τόση ώστε η αποθήκη να έχει τελικά το διπλάσιο του ελαχίστου αποθέματός της.

**Πίνακας 4.3: Διαδικασίες για υλοποίηση**

**4.1 Σχεδίαση E-R diagram**

Ξεκινάμε την εργασία λοιπόν με το διάγραμμα οντοτήτων-σχέσεων(Entities-Relationships diagram) που αντιστοιχεί στα δεδομένα που μας δόθηκαν. Ένα διάγραμμα οντοτήτων-σχέσεων είναι στην ουσία ένα διάγραμμα που παρουσιάζει τα δεδομένα με τρόπο που ενδείκνυται αν θέλουμε να τα περάσουμε σε μία βάση δεδομένων. Όπως δηλώνει και το όνομά του, τα βασικά στοιχεία που αποτελούν το διάγραμμα οντοτήτων-σχέσεων είναι οι οντότητες και οι μεταξύ τους σχέσεις. Να σημειώσουμε, για ιστορικούς λόγους, πως η εν λόγω μοντελοποίηση δημοσιεύθηκε το 1976 από τον Peter Chen.

Στηριζόμενοι λοιπόν στον «Πίνακα 4.1: Super Market’s Elements & Attributes», δημιουργήσαμε τις οντότητες που απαρτίζουν το διάγραμμα οντοτήτων-σχέσεων του Super Market.



**Εικόνα 4.1: Οντότητες & χαρακτηριστικά**

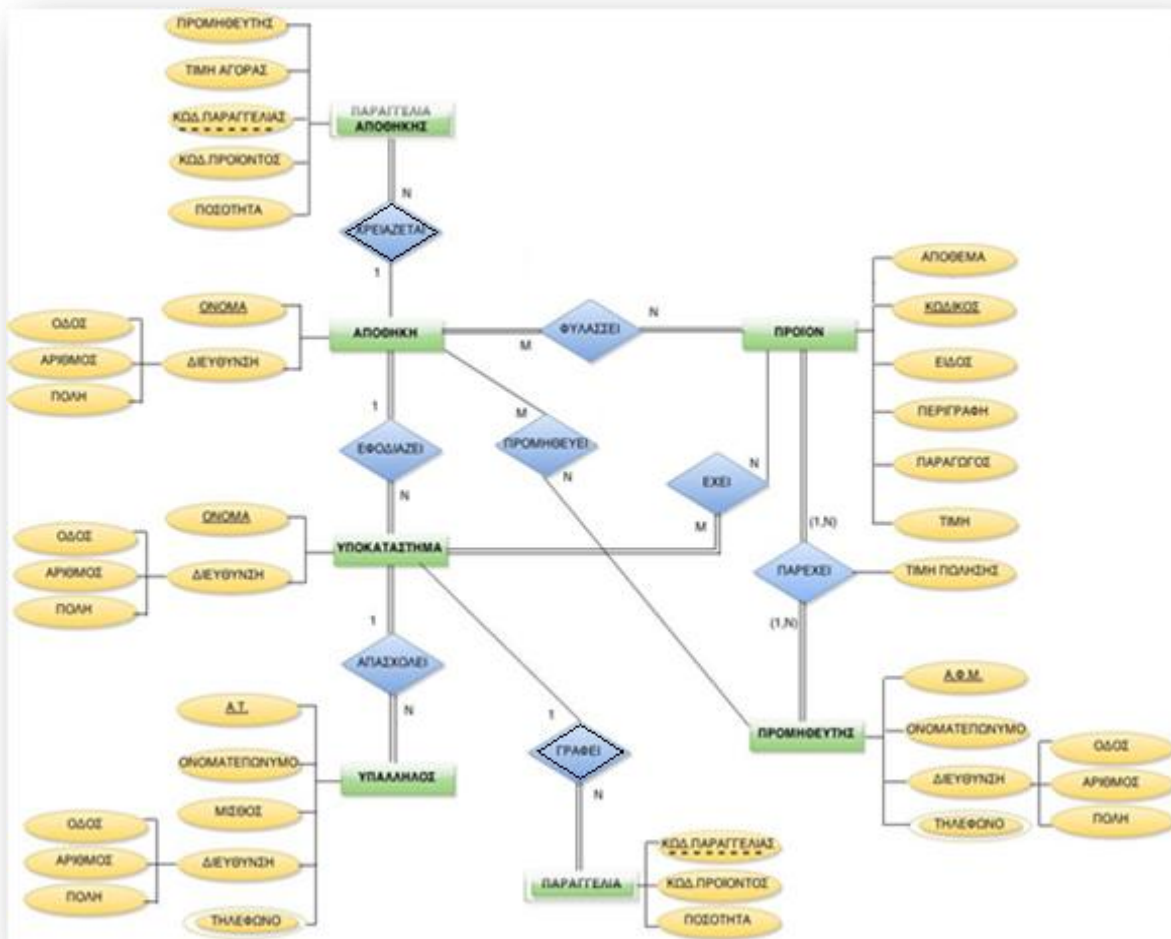
Σε κάποιες από τις οντότητες προσθέσαμε επιπλέον χαρακτηριστικά που θεωρήσαμε απαραίτητα. Στα προϊόντα για παράδειγμα προσθέσαμε τον κωδικό που αποτελεί τον φορολογικό κωδικό που χρησιμοποιεί η εταιρεία για το προϊόν και το χαρακτηρίζει μονοσήμαντα αλλά και το απόθεμα προκειμένου να μπορούμε να υλοποιήσουμε τις αυτόματες παραγγελίες. Επίσης, στον προμηθευτή προσθέσαμε το Α.Φ.Μ του γιατί είναι το καταλληλότερο για να χρησιμοποιηθεί ως πρωτεύον κλειδί (primary key) της οντότητας. Στην παραγγελία πέρα από τον κωδικό του είδους και την ποσότητα, χρειαζόμαστε και έναν κωδικό παραγγελίας που θα είναι μοναδικός για κάθε παραγγελία κάθε υποκαταστήματος. Τέλος, προσθέσαμε μία οντότητα, την παραγγελία αποθήκης, προκειμένου να μας βοηθήσει στη διαδικασία αυτόματης παραγγελίας που πρέπει να υλοποιήσουμε παρακάτω.

Τα χαρακτηριστικά των οντοτήτων που είναι υπογραμμισμένα αποτελούν τα πρωτεύον κλειδιά τους, δηλαδή το χαρακτηριστικό αυτό που είναι μοναδικό για κάθε στιγμιότυπο της οντότητας. Σε κάποιες από αυτές, τα κλειδιά έχουν διακεκομμένη υπογράμμιση και αυτό συμβαίνει γιατί οι αντίστοιχες οντότητες είναι ασθενείς, όρος που θα αναλύσουμε παρακάτω. Συγκεντρωτικά τα πρωτεύον κλειδιά των οντοτήτων που υπάρχουν στο Super Market είναι τα εξής:

Οντότητα	Πρωτεύον κλειδί
Αποθήκη	Όνομα
Υποκατάστημα	Όνομα
Υπάλληλος	Αριθμός Ταυτότητας
Προϊόν	Κωδικός
Προμηθευτής	Α.Φ.Μ
Παραγγελία	Κωδικός παραγγελίας
Παραγγελία αποθήκης	Κωδικός παραγγελίας

Πίνακας 4.4: Οντότητες του Super Market και τα αντίστοιχα κλειδιά

Συνεχίζοντας, για να χτίσουμε το διάγραμμα οντοτήτων-σχέσεων βασίζομαστε στον «Πίνακα 4.2: Σχέσεις μεταξύ οντοτήτων Super Market» και καταλήγουμε στο παρακάτω διάγραμμα:



Εικόνα 4.2: Diagram of Super Market

Οι σχέσεις μεταξύ των οντοτήτων απεικονίζονται σε ρόμβο και έχουμε την περίπτωση της ολικής συμμετοχής (total participation) και της μερικής συμμετοχής (partial participation) στη σχέση. Στην περίπτωση της ολικής συμμετοχής, κάθε στιγμιότυπο της οντότητας θα συμμετέχει στην σχέση κι αυτό απεικονίζεται ενώνοντας την οντότητα αυτή με την σχέση, με διπλή γραμμή. Αντίθετα, στην μερική συμμετοχή δεν είναι υποχρεωτικό κάθε στιγμιότυπο της οντότητας να εμπλέκεται στη σχέση



κι έτσι ενώνουμε την οντότητα και τη σχέση με μια μόνο γραμμή. Παράδειγμα ολικής συμμετοχής στο Super Market αποτελεί η οντότητα του υποκαταστήματος στην σχέση που την συνδέει με τα προϊόντα. Κάθε υποκατάστημα θα έχει υποχρεωτικά προϊόντα(ολική συμμετοχή), ενώ δεν είναι ανάγκη όλα τα προϊόντα να υπάρχουν στα υποκαταστήματα(μερική συμμετοχή).

Στο σημείο αυτό πρέπει να αναφερθούμε σε κάποιες ειδικές απεικονίσεις που χρησιμοποιούμε στο διάγραμμα οντοτήτων-σχέσεων προκειμένου να γίνει απόλυτα κατανοητό. Παραπάνω αναφερθήκαμε σε ασθενείς οντότητες(weak entities). Οι οντότητες αυτές λοιπόν ονομάζονται ασθενείς γιατί εξαρτώνται πλήρως από μία οντότητα που την χαρακτηρίζουμε ισχυρή και με την οποία τις συνδέει μία σχέση. Το πρωτεύον κλειδί μίας ασθενούς οντότητας αποτελείται από το πρωτεύον κλειδί της ισχυρής οντότητας μαζί με ένα χαρακτηριστικό της ασθενούς οντότητας, που το ονομάζουμε μερικό κλειδί. Οι ασθενείς οντότητες απεικονίζονται με διπλό παραλληλόγραμμο, το μερικό κλειδί τους απεικονίζεται με διακεκομμένο οβάλ σχήμα και υπογραμμίζεται με διακεκομμένη γραμμή, ενώ η σχέση που τις συνδέει με την ισχυρή οντότητα απεικονίζεται με διπλό ρόμβο και ονομάζεται ταυτοποιητική συσχέτιση. Ένα παράδειγμα ασθενούς οντότητας στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων είναι η παραγγελία που δεν μπορεί να υπάρξει αν δεν υπάρχει και το υποκατάστημα. Μία ακόμα διαφορετική απεικόνιση στο σχήμα έχουν τα χαρακτηριστικά που μπορούν να έχουν πάνω από μία τιμές, τα πλειότιμα. Αυτά τα απεικονίζουμε με διπλό οβάλ και στο παράδειγμά μας πλειότιμο χαρακτηριστικό αποτελεί το τηλέφωνο.

Να σημειώσουμε τελειώνοντας τη περιγραφή του διαγράμματος οντοτήτων-σχέσεων, πως τα βάρη στις σχέσεις μεταξύ των οντοτήτων τα συμπληρώσαμε με βάση τον «Πίνακα 4.2: Σχέσεις μεταξύ οντοτήτων Super Market». Προσθέσαμε κι εμείς κάποιες σχέσεις που δε δίνονταν, όπως ‘υποκατάστημα – προϊόν’ που είναι M:N, προκειμένου να αποθηκεύουμε ότι μας ζητήθηκε. Ακόμα δύο σχέσεις προστιθέμενες από εμάς είναι αυτή της ‘αποθήκη-παραγγελία αποθήκης’ και ‘υποκατάστημα-παραγγελία’ που είναι αντίστοιχης, σχέση 1:N. Επίσης, πέρα από τα χαρακτηριστικά των οντοτήτων, υπάρχουν και τα χαρακτηριστικά που κρατάμε για μια σχέση και απεικονίζεται με οβάλ και συνδέεται πάνω στην εν λόγω σχέση αλλά εμείς δεν έχουμε τέτοιο χαρακτηριστικό.

## 5 XML Schema

Ένα πληροφοριακό σύστημα, προκειμένου να μπορεί να καταλάβει και να επεξεργαστεί ένα ηλεκτρονικό έγγραφο, θα πρέπει να υπακούει αυτό το έγγραφο σε κανόνες που αναφέρονται στην ιεραρχία των δεδομένων αλλά και στην δομή της πληροφορίας. Επομένως, αυτό που χρειαζόμαστε είναι μία προτυποποίηση των εγγράφων και πραγματοποιείται με την χρήση των γλωσσών σχημάτων(schema languages). Υπάρχουν αρκετές γλώσσες σχημάτων(schema languages), όπως η Document Type Definitions (DTDs) και η Relax-NG, αλλά η πιο διαδεδομένη είναι η XML Schema Description Language, την οποία και θα χρησιμοποιήσουμε. Μπορούμε να παρομοιάσουμε το XSD<sup>4</sup> με ένα συντακτικό, πάνω στο οποίο θα στηριχτούμε για να παράγουμε ένα σωστά δομημένο XML έγγραφο. Άρα, αυτό που περιέχει την πληροφορία που θέλουμε να μεταφερθεί σε μία εφαρμογή, είναι το XML έγγραφο στηριζόμενο απλά στους κανόνες του XSD.

### 5.1 Βασικοί κανόνες σύνταξης για XML Schema

Το XSD ξεκινάει πάντα με το κεντρικό στοιχείο(element) του σχήματος, που αποκαλείται ρίζα(root element), το οποίο μπορεί να περιλαμβάνει τόσο στοιχεία(elements) όσο και γνωρίσματα(attributes) και περικλείεται μέσα στην ετικέτα(tag) :

```
<xs:schema> ..... </xs:schema>
```

Το κάθε στοιχείο(element) ορίζει τι δομή θα έχει ένα δεδομένο κι ανάλογα την πολυπλοκότητα των δεδομένων έχουμε τους απλούς τύπους(simple type) και τους σύνθετους τύπους(complex type) στοιχείου(element). Ένας απλός τύπος στοιχείου(simple type element) περιγράφει δεδομένα που συνήθως είναι είτε απλά string, integer κλπ είτε τύπου Boolean, όπως για παράδειγμα :

```
<xs:element name="FirstName" type="xs:string"/>
```

Αντίθετα, ένας σύνθετος τύπος στοιχείου complex type (element) περιγράφεται από μία ομάδα δεδομένων και μπορεί να περιέχει και άλλα στοιχεία(elements), απλά ή σύνθετα και γνωρίσματα(attributes). Για το λόγο αυτό όμως πρέπει να ορίζουμε τη σειρά με την οποία θα συναντάμε τα στοιχεία(elements) μέσα σε σύνθετα στοιχεία(complex elements) αλλά και το πόσο συχνά αυτά θα εμφανίζονται. Για να καθορίσουμε την σειρά που θα εμφανίζονται τα δεδομένα, έχουμε τις εξής επιλογές στις ετικέτες(tag):

Ετικέτες(tags)	Αποτέλεσμα
<code>&lt;all&gt; ... &lt;/all&gt;</code>	Θα πρέπει να υπάρχουν όλα τα στοιχεία(elements) χωρίς όμως συγκεκριμένη σειρά.
<code>&lt;choice&gt; ... &lt;/choice&gt;</code>	Θα πρέπει να υπάρχει ένα από όλα τα στοιχεία(elements).
<code>&lt;Sequence&gt; ... &lt;/Sequence&gt;</code>	Θα πρέπει να υπάρχουν όλα τα στοιχεία(elements) με τη σειρά που έχουν δηλωθεί μέσα στα tags.

Πίνακας 5.1: Ετικέτες(tags) αλληλουχίας στοιχείων(elements)

Ενώ για την συχνότητα εμφάνισής τους έχουμε τα χαρακτηριστικά 'minOccurs' και 'maxOccurs' που ορίζουν τον ελάχιστο και το μέγιστο αριθμό εμφανίσεων του στοιχείου(element) αντίστοιχα.

```
<xs:element name="Phone" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
```

Ένα ολοκληρωμένο παράδειγμα ενός σύνθετου στοιχείου(complex element) είναι αυτό της ταχυδρομικής διεύθυνσης:

<sup>4</sup> Xml Schema Description

```

<xs:element name="address">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Number" type="xs:int"/>
      <xs:element name="Street" type="xs:string"/>
      <xs:element name="ZipCode" type="xs:int"/>
      <xs:element name="City" type="xs:string"/>
      <xs:element name="Country" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

## 5.2 Κανόνες μετατροπής E-R Diagram σε XML Schema

Στην παρούσα ενότητα παρουσιάζουμε τους κανόνες μετατροπής που υπάρχουν για τον μετασχηματισμό του διαγράμματος οντοτήτων-σχέσεων (Entities-Relationships diagram) σε ένα ορθό XML Schema. Για τον σκοπό αυτό ακολουθήσαμε τους κανόνες μετατροπής που παρουσιάζονται στην δημοσίευση [14].

- 1<sup>ος</sup> Κανόνας: Για το διάγραμμα οντοτήτων-σχέσεων δημιουργούμε ένα στοιχείο ρίζα (root element) που ονομάσαμε 'SuperMarket' και το οποίο πρέπει να δομείται ως εξής:

```

<xs:element name="SuperMarket">
  <xs:complexType>
    <xs:sequence>
      <!--Όλα τα στοιχεία(elements) που θα προκύψουν από τον
      μετασχηματισμό τοποθετούνται εδώ -->
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

- 2ος κανόνας: Για κάθε κανονικό σύνολο οντοτήτων (Regular Entity Set) του διαγράμματος οντοτήτων-σχέσεων, δημιουργείται ένα στοιχείο(element) και τοποθετείται κάτω από το στοιχείο ρίζα(root element). Το πρωτεύον κλειδί της κάθε οντότητας καθορίζεται από μια δήλωση κλειδιού που περιλαμβάνει όλα τα γνωρίσματα (attributes) τα οποία το διαμορφώνουν. Ας δούμε όμως συγκεκριμένα το παράδειγμα του στοιχείου(element) 'apothiki', ώστε να γίνει κατανοητός ο κανόνας:

```

<xs:element name="apothiki">
  <xs:complexType>
    <xs:sequence>
      <!--όλα τα στοιχεία(elements) που περιγράφουν την
      συγκεκριμένη οντότητα αναπτύσσονται εδώ -->
    </xs:sequence>
    <!--δήλωση του γνωρίσματος(attribute) 'onoma' ως
    πρωτεύον κλειδί για το στοιχείο apothiki -->
  </xs:complexType>
  <xs:key name="key_apothiki">
    <xs:selector xpath="apothiki"/>
    <xs:field xpath="@onoma"/>
  </xs:key>
</xs:element>

```

Αντίστοιχα με το παράδειγμα που μόλις είδαμε, δημιουργήσαμε ανάλογες δομές για όλα τα στοιχεία(elements) του διαγράμματος οντοτήτων-σχέσεων που είναι κανονικά σύνολα οντοτήτων(Regular Entity Set).

- 3<sup>ος</sup> κανόνας: Αφορά τις ασθενείς οντότητες (weak entities) του διαγράμματος οντοτήτων-σχέσεων για τις οποίες δημιουργούμε ένα στοιχείο(element) με το όνομα της ασθενής οντότητας και την τοποθετούμε κάτω από το στοιχείο(element) της ισχυρής οντότητας. Επίσης, φροντίζουμε ο ορισμός του πρωτεύον κλειδιού της ασθενής οντότητας, να περιλαμβάνει και το πρωτεύον κλειδί της ισχυρής οντότητας. Ασθενής οντότητα στο παράδειγμά μας αποτελεί η παραγγελία του υποκαταστήματος και της αποθήκης. Παραθέτουμε τον κώδικα για την παραγγελία αποθήκης και αντίστοιχα δουλέψαμε και για την παραγγελία υποκαταστήματος:

```
<xs:element name="apothiki">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="paraggelia" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <!-- εδώ μπαίνουν τα στοιχεία(elements)παραγγελίας- ->
            </xs:sequence>
            <xs:attribute name="kwdikosParaggelias" type="xs:int" use="required"/>
          </xs:complexType>
          <xs:key name="key_paraggelia">
            <xs:selector xpath="paraggelia"/>
            <xs:field xpath="apothiki/@onoma"/>
            <xs:field xpath="@kwdikosParaggelias"/>
          </xs:key>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

- 4<sup>ος</sup> κανόνας: Αφορά τις οντότητες υψηλού επιπέδου (High-level entity set) που δεν προβλέπονται όμως στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων. Ο κανόνας μετατροπής ωστόσο αναφέρεται εδώ [14].
- 5<sup>ος</sup> κανόνας: Αφορά one-to-one σχέσεις που δεν υπάρχουν στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων. Ο κανόνας μετατροπής ωστόσο αναφέρεται εδώ [14].
- 6<sup>ος</sup> κανόνας: Αφορά τις 1-to-N σχέσεις (one-to-many relationships). Δημιουργείται ένα στοιχείο(element) με το όνομα της σχέσης και ανάλογα, αν η N πλευρά έχει ολική (total) ή μερική (partial) συμμετοχή στη σχέση, διαχωρίζουμε τις εξής περιπτώσεις:
  - a. Αν η σχέση είναι ολική: δημιουργούμε το στοιχείο(element) που περιγράφει τη σχέση κάτω από το στοιχείο της πλευράς N, τροποποιούμε το στοιχείο της πλευράς N προσθέτοντας maxOccur="unbounded" και το μετακινούμε κάτω από το στοιχείο που βρίσκεται στην πλευρά 1. Έχουμε αρκετές τέτοιου είδους σχέσεις στο παράδειγμα του SuperMarket και μία από αυτές είναι η σχέση 'Αποθήκη-Υποκατάστημα', αλλά εφόσον δεν υπάρχει κάποιο γνώρισμα που να περιγράφει την σχέση, δεν δημιουργήσαμε στοιχείο(element) γι' αυτήν:

```
<xs:element name="apothiki">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ypokatasthma" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <!-- εδώ τοποθετούνται τα στοιχεία(elements)
            του υποκαταστήματος -->
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

b. Αν η σχέση είναι μερική: δημιουργούμε το στοιχείο που περιγράφει τη σχέση κάτω από το στοιχείο της πλευράς N. Δημιουργούμε το ξένο κλειδί με τα χαρακτηριστικά που χρειάζεται μέσα στο element της σχέσης και επίσης δημιουργούμε αναφορές στο κλειδί της οντότητας που βρίσκεται στην πλευρά 1. Τέτοιες σχέσεις όμως δεν προβλέπονται στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων.

- 7<sup>ος</sup> κανόνας: Αφορά τις N-to-M σχέσεις (Many-to-many relationships). Για τη σχέση δημιουργείται ένα στοιχείο(element) με το όνομα της σχέσης και με το χαρακτηριστικό maxOccurs="unbounded". Στη συνέχεια τοποθετούμε το στοιχείο(element) που περιγράφει την σχέση κάτω από το στοιχείο(element) της επικρατέστερης οντότητας (dominant entity). Προσθέτουμε τα χαρακτηριστικά των ξένων κλειδίων μέσα στο στοιχείο(element) της σχέσης με ξεχωριστές keyref δηλώσεις που αναφέρονται στην οντότητα που βρίσκεται στην άλλη μεριά της σχέσης. Στο δικό μας διάγραμμα οντοτήτων-σχέσεων συναντάμε τέτοιου είδους σχέσεις και μία από αυτές είναι η σχέση 'Αποθήκη-Προϊόν'. Το γεγονός όμως ότι δεν έχουμε κάποιο χαρακτηριστικό στην σχέση που ενώνει τις οντότητες, κάνει μη απαραίτητη τη χρήση στοιχείου(element) για την σχέση. Οπότε βάζουμε απλά κάτω από την επικρατέστερη οντότητα, που είναι η Αποθήκη, την οντότητα της άλλης μεριάς, το Προϊόν, με το χαρακτηριστικό maxOccurs="unbounded". Ανάλογα δουλέψαμε και για τις υπόλοιπες σχέσεις N-to-M.
- 8<sup>ος</sup> κανόνας: Αφορά τις αναδρομικές σχέσεις (recursive relationships) που δεν προβλέπονται στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων. Ο κανόνας μετατροπής ωστόσο αναφέρεται εδώ [14].
- 9ος κανόνας: Αφορά τις ISA σχέσεις που δεν προβλέπονται στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων. Ο κανόνας μετατροπής ωστόσο αναφέρεται εδώ [14].
- 10<sup>ος</sup> κανόνας: Αφορά τις N-ary σχέσεις που δεν προβλέπονται στο συγκεκριμένο διάγραμμα οντοτήτων-σχέσεων. Ο κανόνας μετατροπής ωστόσο αναφέρεται εδώ [14].
- 11<sup>ος</sup> κανόνας: Αφορά τα σύνθετα γνώρισμα (composite attributes) τα οποία έχουν δικά τους γνώρισμα που τα περιγράφουν. Για κάθε γνώρισμα που περιγράφεται είτε σε μια οντότητα, είτε σε μια σχέση, είτε σε ένα σύνθετο γνώρισμα, δημιουργούμε ένα στοιχείο(element) με το όνομα του χαρακτηριστικού και το τοποθετούμε κάτω από το στοιχείο(element) που περιγράφει, είτε είναι οντότητα, είτε σχέση είτε ένα σύνθετο χαρακτηριστικό. Παράδειγμα σύνθετου γνωρίσματος(composite attributes) στο παράδειγμά μας, αποτελεί η διεύθυνση:

```
<xs:element name="dieythinsi">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Odos" type="xs:string"/>
      <xs:element name="Arithmos" type="xs:int"/>
      <xs:element name="Poli" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

- 12<sup>ος</sup> κανόνας: Αφορά απλά γνώρισμα (atomic attribute). Οι κανόνες που εφαρμόζονται εδώ έχουν άμεση εξάρτηση με το αν είναι πολλαπλών τιμών το γνώρισμα ή παίρνει μια τιμή:
  - a. Αν είναι πλειότιμο, δημιουργούμε ένα στοιχείο(element) με το όνομά του και το τοποθετούμε κάτω από το στοιχείο(element) που περιγράφει και επίσης προσθέτουμε το maxOccurs="unbounded". Στο παράδειγμα μας, πλειότιμο γνώρισμα αποτελεί το τηλέφωνο που το συναντάμε σαν γνώρισμα σε πολλές οντότητες. Μία από αυτές είναι και ο υπάλληλος και ο κώδικας που χρησιμοποιούμε είναι ο εξής:

```
<xs:element name="ypallhlos" >
  <xs:complexType>
    <xs:sequence>
      <!-- - τα υπόλοιπα element του υπαλλήλου -- >
      <xs:element ref="thlefwno" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

- b. Αν παίρνει μια τιμή μόνο τότε έχουμε δύο επιλογές. Μπορούμε να δημιουργήσουμε ένα γνώρισμα(attribute) με το όνομά του και να το προσθέσουμε στο στοιχείο(element) που περιγράφει. Εναλλακτικά, μπορούμε να δημιουργήσουμε ένα στοιχείο(element) με το όνομά του και να το τοποθετήσουμε κάτω από το στοιχείο(element) της οντότητας ή της σχέσης ή του σύνθετου χαρακτηριστικού που περιγράφει. Στην οντότητα υπάλληλος συναντάμε και τους δύο τρόπους. Να σημειώσουμε ότι το πρωτεύον κλειδί της οντότητας το ορίζουμε ως γνώρισμα(attribute) γιατί στα στοιχεία(elements) δεν έχουμε την δυνατότητα να βάλουμε το χαρακτηριστικό της απαραίτητης χρήσης, use="required".

```
<xs:element name="ypallhlos" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="onomatepwynomo" type="xs:string"/>
      <xs:element name="misthos" type="xs:float"/>
      <xs:element ref="dieythinsi"/>
      <xs:element ref="thlefwno" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="at" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:key name="key_ypallhlos">
    <xs:selector xpath="ypallhlos"/>
    <xs:field xpath="@at"/>
  </xs:key>
</xs:element>
```

### 5.3 Ανάλυση XML Schema του Super Market

Ας ξεκινήσουμε αναφέροντας πως το στοιχείο ρίζα(root element) του XSD είναι το σύνθετο element(complex element) με όνομα 'Super Market', κάτω από το οποίο θα 'κρέμονται' όλα τα υπόλοιπα στοιχεία(elements) που προκύπτουν από το διάγραμμα οντοτήτων- σχέσεων, σύνθετα και απλά. Τα στοιχεία(elements) 'thlefwno' και 'dieythinsi', αφού τα έχουν σαν στοιχεία(elements) πάνω από μία οντότητες, θα τα ορίσουμε χωριστά, έξω από το 'Super Market' και θα κάνουμε σε αυτά αναφορά όταν θέλουμε να τα χρησιμοποιήσουμε. Πιο συγκεκριμένα, το 'thlefwno' επειδή είναι πλειότιμο, σκεφτήκαμε πως μαζί με τον αριθμό τηλεφώνου είναι χρήσιμο να κρατάμε και το είδος του τηλεφώνου, όπως π.χ. αν είναι κινητό ή σταθερό. Οι τιμές που μπορεί να πάρει το είδος του τηλεφώνου είναι συγκεκριμένες και τη λίστα την ορίζουμε εμείς. Δηλώνουμε αρχικά τί τύπου θα είναι οι τιμές που θα έχει η λίστα, στην περίπτωση μας είναι τύπου 'normalizedString':

```
<xs:restriction base="xs:normalizedString"> .....
```

κι έπειτα δηλώνουμε τις τιμές που θα έχει η λίστα:

```
<xs:enumeration value="..."/>
```

Όσο για τη 'dieythinsi', είναι κι αυτή σύνθετο στοιχείο(complex element) αλλά αποτελείται μόνο από απλά στοιχεία(simple elements), την οδό, τον αριθμό και την πόλη, κανόνας N<sup>ο</sup> 11.

```

<xs:element name="thlefwno">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="eidosthlefwnou" type="phoneTypes"/>
      <xs:element name="number" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:simpleType name="phoneTypes">
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="κινητό"/>
    <xs:enumeration value="σταθερό"/>
    <xs:enumeration value="FAX"/>
  </xs:restriction>
</xs:simpleType>
<xs:element name="dieythinsi">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Odos" type="xs:string"/>
      <xs:element name="Arithmos" type="xs:int"/>
      <xs:element name="Poli" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

Εικόνα 5.1: Περιγραφή δομής element τηλεφώνου &amp; διεύθυνσης

Έξω από το στοιχείο ρίζα(root element) δηλώνουμε επίσης κάποιους σύνθετους τύπους στοιχείων(complex elements) που θα χρησιμοποιήσουμε για να περιγράψουμε κάποια στοιχεία(elements) :

- 'promhtheytesLista'
- 'promhtheythsPerigraph'
- 'proiontaLista'
- 'proionPerigraph'

Σκοπός μας είναι να δηλώσουμε δύο στοιχεία(elements) στο 'SuperMarket' από τα οποία το ένα θα είναι τύπου 'promhtheythsPerigraph' και το άλλο τύπου 'proionPerigraph'. Εκεί θα κρατάμε την περιγραφή για όλα τα προϊόντα που δουλεύει η εταιρεία καθώς και τα στοιχεία όλων των προμηθευτών με τους οποίους συνεργάζεται η εταιρεία. Ενώ κάτω από τις οντότητες που συνδέονται με προϊόντα ή προμηθευτές, θα δημιουργήσουμε στοιχεία(elements) τύπου 'proiontaLista' και τύπου 'promhtheytesLista', αντίστοιχα.

Τα στοιχεία(elements) τύπου 'proiontaLista' θα κάνουν αναφορά στα στοιχεία(elements) τύπου 'proionPerigraph' και αντίστοιχα, τα στοιχεία(elements) τύπου 'promhtheytesLista' θα κάνουν αναφορά στα στοιχεία(elements) τύπου 'promhtheythsPerigraph'. Η αναφορά αυτή θα γίνεται μέσω ID και IDREFS. Πιο συγκεκριμένα, το στοιχείο(element) τύπου 'proiontaLista' θα περιέχει ένα χαρακτηριστικό τύπου 'IDREF' το οποίο κάνει αναφορά σε χαρακτηριστικό τύπου 'ID' που περιέχεται στα στοιχεία(elements) τύπου 'proionPerigraph'. Ακριβώς αντίστοιχα συμβαίνει και η αναφορά των στοιχείων(elements) τύπου 'promhtheytesLista' στα στοιχεία(elements) τύπου 'promhtheythsPerigraph'. Να σημειώσουμε πως το χαρακτηριστικό που θα οριστεί ως τύπου 'ID' θα είναι και το πρωτεύον κλειδί του στοιχείου(element) που περιγράφει, για να επωφεληθούμε την ιδιότητα της μοναδικότητας.

Θεωρήσαμε καλύτερο έναν τέτοιο τρόπο σχεδίασης του XML Schema έτσι ώστε οι οντότητες που συνδέονται με προϊόντα και προμηθευτές να κάνουν αναφορά σε αυτά τα στοιχεία(elements). Το

πλεονέκτημα που κερδίζουμε από αυτή τη σχεδίαση είναι ότι δεν θα μπορεί να γίνει αναφορά σε προμηθευτή ο οποίος δεν υπάρχει στην λίστα των προμηθευτών και κατ' επέκταση δε συνεργάζεται καν με το SuperMarket. Αντίστοιχα, ένα μαγαζί δε θα μπορεί για παράδειγμα στην παραγγελία του να καταχωρήσει κωδικό προϊόντος αν αυτό το προϊόν δεν το δουλεύει καθόλου η εταιρεία. Ακολουθεί το συγκεκριμένο κομμάτι κώδικα από το XML Schema ξεκινώντας με τους τύπους που αφορούν τα προϊόντα και συνεχίζοντας με τους τύπους που αφορούν τους προμηθευτές:

```
<xs:complexType name="proiontaLista">
  <xs:sequence>
    <xs:element name="proionIDREF" type="xs:IDREF"/>
    <xs:element name="apothema" type="xs:float"/>
    <xs:element name="minApothema" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="proionPerigrafh">
  <xs:sequence>
    <xs:element name="proionKwdikos" type="xs:ID"/>
    <xs:element name="perigrafh" type="xs:string"/>
    <xs:element name="eidos" type="xs:string"/>
    <xs:element name="paragwgos" type="xs:string"/>
    <xs:element name="timh" type="xs:float"/>
    <xs:element name="promhtheyths" type="promhtheytesLista"
      minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

Εικόνα 5.2: Περιγραφή τύπου 'proiontaLista' και 'proionPerigrafh'

Όπως βλέπουμε, ο τύπος 'proionPerigrafh' περιέχει σε στοιχεία(elements) τα χαρακτηριστικά της οντότητας προϊόν, σύμφωνα με τον κανόνα 12.b που αναφέραμε στην προηγούμενη ενότητα, αλλά κι ένα στοιχείο(element) τύπου 'promhtheytesLista'. Ο λόγος που προσθέσαμε κι αυτό το στοιχείο(element) είναι γιατί χρειαζόμαστε για κάθε προϊόν, τους προμηθευτές που μας το προμηθεύουν καθώς και την τιμή πώλησης που έχει ο καθένας. Η δομή αυτή μας εξυπηρετεί για την διαδικασία αυτόματης παραγγελίας που θα υλοποιήσουμε, αφού θέλουμε να παραγγέλνουμε το προϊόν που έχουμε έλλειψη, από τον προμηθευτή με την μικρότερη τιμή. Ενώ η δομή του τύπου 'proiontaLista' εξυπηρετεί για το απόθεμα που θέλουμε να κρατάμε για τα προϊόντα στις αποθήκες και στα υποκαταστήματα.



```

<xs:complexType name="promhtheytesLista">
  <xs:sequence>
    <xs:element name="promitheutisIDREF" type="xs:IDREF"/>
    <xs:element name="timhPwlhshs" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="promhtheythsPerigrafh">
  <xs:sequence>
    <xs:element name="afm" type="xs:ID"/>
    <xs:element name="onomatepwnymo" type="xs:string"/>
    <xs:element ref="dieythinsi"/>
    <xs:element ref="thlefwno" minOccurs="1"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

```

Εικόνα 5.3: Περιγραφή τύπου 'promhtheytesLista' και 'promhtheythsPerigrafh'

Επιστρέφοντας στο στοιχείο ρίζα(root element), ας εξετάσουμε ξεχωριστά τα στοιχεία(elements) που το αποτελούν. Συναντάμε στην αρχή την 'arothiki' κι έπειτα τα στοιχεία(elements) τύπου 'promhtheythsPerigrafh' και 'proionPerigrafh' στα οποία αναφερθήκαμε. Οι ετικέτες(tags) που χρησιμοποιήσαμε για την ταξινόμηση των στοιχείων(elements) αυτών είναι ο συνδυασμός της αυστηρής ακολουθίας(sequence) και της επιλογής(choise), στην οποία έχουμε προσθέσει το χαρακτηριστικό 'minOccurs="0"' και 'maxOccurs="unbounded"'. Με τον τρόπο αυτό, δεν είναι υποχρεωτικό να υπάρχουν όλα τα στοιχεία(elements) αλλά αν υπάρχουν, θα μπουν με τη σειρά που τα ορίσαμε.

Στην απόφαση αυτής της σχεδίασης φτάσαμε όταν θελήσαμε με XQuery να προσθέσουμε μόνο ένα από τα στοιχεία του 'SuperMarket'. Αν χρησιμοποιούσαμε μόνο την ετικέτα αυστηρής ακολουθίας(sequence), στην προσθήκη για παράδειγμα μόνο του στοιχείου 'arothiki' θα ζητούσε μετά από κάθε 'arothiki' και στοιχεία(elements) 'proionta' και 'promhtheyths'. Ενώ στην χρήση μόνο της ετικέτας(tags) επιλογής(choise) με το χαρακτηριστικό 'minOccurs="0"' και 'maxOccurs="unbounded"' αν θέλαμε όλα τα στοιχεία(elements) δεν θα είχαμε συγκεκριμένη σειρά εμφάνισής τους. Οπότε καταλήξαμε στον συνδυασμό τους για να πάρουμε το επιθυμητό αποτέλεσμα με τον κώδικα που ακολουθεί:

```

<xs:element name="SuperMarket">
  <xs:complexType>
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="arothiki">
          <xs:element name="proionta" type="proionPerigrafh"
            minOccurs="1" maxOccurs="unbounded"/>
          <xs:element name="promhtheyths" type="promhtheythsPerigrafh"
            minOccurs="1" maxOccurs="unbounded"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

Εικόνα 5.4: Στοιχεία(elements) του Super Market

Το πρώτο στοιχείο(element) που έχει το ‘Super Market’ είναι η ‘apothiki’, το οποίο είναι κι αυτό με τη σειρά του σύνθετο. Περιλαμβάνει αρχικά τα χαρακτηριστικά της, κανόνας N°12.b, αλλά και τις οντότητες με τις οποίες συνδέεται αυτή μέσω των σχέσεων που ορίστηκαν στο διάγραμμα οντοτήτων-σχέσεων, σύμφωνα με τους κανόνες N° 6 και N° 7.

```
<xs:element name="apothiki">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="dieythinsi"/>
      <xs:element name="ypokatasthma" maxOccurs="unbounded">
      <xs:element name="proiontaApothema" type="proiontaLista"
minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="promhtheythsApothikis" type="promhtheytesLista"
minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="paraggeliaApothikis" minOccurs="0" maxOccurs="unbounded">
    </xs:sequence>
    <xs:attribute name="onoma" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:key name="key_apothiki">
    <xs:selector xpath="apothiki"/>
    <xs:field xpath="@onoma"/>
  </xs:key>
</xs:element>
```

Εικόνα 5.5: Element αποθήκη

Δύο από τα στοιχεία(elements) που περιλαμβάνει η ‘apothiki’ είναι σύνθετα. Αναφερόμαστε στα ‘ypokatasthma’ και ‘paraggeliaApothikis’ και ξεκινάμε την ανάλυση από το πρώτο:

```
<xs:element name="ypokatasthma" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="dieythinsi"/>
      <xs:element name="ypallhlos" maxOccurs="unbounded">
      <xs:element name="paraggelia" minOccurs="0" maxOccurs="unbounded">
      <xs:element name="proiontaApothema" type="proiontaLista"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="onoma" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:key name="key_ypokatasthma">
    <xs:selector xpath="ypokatasthma"/>
    <xs:field xpath="@onoma"/>
  </xs:key>
</xs:element>
```

Εικόνα 5.6: Στοιχείο(element) ‘ypokatasthma’

Βλέπουμε λοιπόν ότι μιλάμε για ακόμα ένα σύνθετο στοιχείο(complex element) το οποίο πέρα από τα χαρακτηριστικά του, κανόνας N° 12.b, περιέχει κι άλλα στοιχεία(elements) που προκύπτουν από τις σχέσεις που τα συνδέουν, κανόνας N° 6 και N°7. Περιέχει στοιχείο(element) για την οντότητα υπάλληλος καθώς και στοιχείο(element) για την ασθενής οντότητα παραγγελία. Ακόμη, παρατηρούμε το στοιχείο(element) που είναι τύπου 'proiontaLista', στο οποίο αναφερθήκαμε παραπάνω, που εξασφαλίζει την αποθήκευση του αποθέματος των προϊόντων στο υποκατάστημα. Ο τρόπος απεικόνισής τους σε XML Schema στηρίζεται πάντα στους κανόνες που έχουμε αναφέρει και ακολουθεί ο συγκεκριμένος κώδικάς τους.

```
<xs:element name="ypallhlos" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="onomatepwnymo" type="xs:string"/>
      <xs:element name="misthos" type="xs:float"/>
      <xs:element ref="dieythinsi"/>
      <xs:element ref="thlefwno" minOccurs="1" maxOccurs="unbounded"/>
      <xs:attribute name="at" type="xs:string" use="required"/>
    </xs:sequence>
  </xs:complexType>
  <xs:key name="key_ypallhlos">
    <xs:selector xpath="ypallhlos"/>
    <xs:field xpath="@at"/>
  </xs:key>
</xs:element>
```

Εικόνα 5.7: Στοιχείο(element) 'ypallhlos'

```
<xs:element name="paraggelia" minOccurs="0" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="kwdikos" type="xs:IDREF"/>
      <xs:element name="posothta" type="xs:float"/>
    </xs:sequence>
    <xs:attribute name="kwdikosParaggelias" type="xs:int" use="required"/>
  </xs:complexType>
  <xs:key name="key_paraggelia">
    <xs:selector xpath="paraggelia"/>
    <xs:field xpath="ypokatasthma/@onoma"/>
    <xs:field xpath="@kwdikosParaggelias"/>
  </xs:key>
</xs:element>
```

Εικόνα 5.8: Στοιχείο(element) 'paraggelia'

Περνάμε στο δεύτερο σύνθετο στοιχείο(complex element) του σύνθετου στοιχείου(complex element) 'apothiki', το στοιχείο(element) 'paraggeliaApothikis'. Να σημειώσουμε πως πρόκειται για μία ασθενής οντότητα αφού εξαρτάται πλήρως από την 'apothiki', γεγονός που απεικονίζεται τόσο στο διάγραμμα οντοτήτων-σχέσεων όσο και στον κώδικα που ακολουθεί, στο σημείο ορισμού του πρωτεύοντος κλειδιού του, κανόνας N° 3.

```

<xs:element name="paraggeliaApothikis" minOccurs="0"
maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="kwdikos" type="xs:IDREF"/>
      <xs:element name="posothta" type="xs:float"/>
      <xs:element name="promhtheyths" type="xs:IDREF"/>
      <xs:element name="timhAgoras" type="xs:float"/>
    </xs:sequence>
    <xs:attribute name="kwdikosParaggeliasApothikis"
type="xs:int" use="required"/>
  </xs:complexType>
  <xs:key name="key_paraggeliaApothikis">
    <xs:selector xpath="paraggeliaApothikis"/>
    <xs:field xpath="@kwdikosParaggeliasApothikis"/>
  </xs:key>
</xs:element>

```

Εικόνα 5.9: Στοιχείο(element) 'paraggeliaApothikis'

Ενώνοντας τα κομμάτια που αναλύσαμε και συμβουλευόμενοι πάντα τους κανόνες της υπο-ενότητας 5.2 προκύπτει το XML Schema που βρίσκεται στο Παράρτημα Ν° 1.

#### 5.4 eXist-db

Η eXist-db αποτελεί μια ελεύθερη και ανοιχτού κώδικα βάση δεδομένων που χρησιμοποιείται αποκλειστικά για την διαχείριση XML δεδομένων. Ουσιαστικά, μας επιτρέπει τη χρήση XML εγγράφων ως τη βασική μονάδα αποθήκευσης και την XML να χρησιμοποιείται ως την εσωτερική δομή των δεδομένων. Βασικά χαρακτηριστικά της eXist-db είναι η εύκολη εγκατάστασή της, βλ. Παράρτημα Ν° 2, ενώ παράλληλα παρέχει και εύκολο τρόπο εκκίνησης της χρήσης της. Επίσης η φόρτωση των δεδομένων σε XML μορφή γίνεται με την υποστήριξη απλών ενεργειών όπως drag and drop. Επιπλέον, υποστηρίζει την XQuery για την εκτέλεση επερωτήσεων στα XML δεδομένα που περιέχει, για την οποία θα μιλήσουμε εκτενέστερα στο απόμεινο κεφάλαιο. Παρέχει επίσης, υποστήριξη στην XPath, XSLT, XQuery Update καθώς και πλήρη αναζήτηση κειμένου (Full Text Search).

Το σημαντικότερο χαρακτηριστικό της eXist-db που χρησιμοποιήθηκε στην παρούσα εργασία αποτελεί η επέκταση ενημερώσεων που παρέχεται στην XQuery (XQuery Update Extension). Η συγκεκριμένη επέκταση σχεδιάστηκε με σκοπό τη μόνιμη ενημέρωση εγγράφων που είναι αποθηκευμένα στη βάση δεδομένων. Ωστόσο το μειονέκτημα αυτής της επέκτασης είναι ότι δεν μπορεί να μεταβάλλει τα αποτελέσματα που επιστρέφονται από ένα ερώτημα. Συνεπώς, για κόμβους που φορτώνονται στη μνήμη (in-memory constructed elements) δεν έχει νόημα η εφαρμογή των ενημερώσεων καθώς επιστρέφουν το κενό στοιχείο. Όλες οι δηλώσεις ενημέρωσης ξεκινάνε με την λέξη κλειδί "update" και ακολουθεί η ενέργεια ενημέρωσης: "insert", "delete", "replace", "value" και "rename". Οι συγκεκριμένες δηλώσεις ενημέρωσης μπορούν να ενσωματωθούν οπουδήποτε στον κώδικα της XQuery ή στο σώμα της συνάρτησης. Προσοχή επίσης πρέπει να δοθεί σε ενέργειες διαγραφής που συνδυάζονται με FLWOR εκφράσεις, που θα δούμε παρακάτω, καθώς η διαγραφή ή αντικατάσταση εφαρμόζονται άμεσα και δεν είναι διαθέσιμα για κάποιο επερωτήμα.

## 6 XQuery

Για να ξεκινήσουμε να μιλάμε για την γλώσσα XQuery πρέπει πρώτα να αναφερθούμε στα βασικά στοιχεία της γλώσσας XPath, αφού η πρώτη δημιουργήθηκε με σκοπό την επέκταση της λειτουργικότητας της δεύτερης. Η XPath λοιπόν δημιουργήθηκε για να βρίσκει συγκεκριμένα σημεία μέσα σε ένα XML έγγραφο εντοπίζοντας ένα δομικό στοιχείο του XML εγγράφου, με βάση τη θέση του στο έγγραφο XML, τον τύπο του ή γενικά κάποιο άλλο κριτήριο. Ένα πλεονέκτημα της γλώσσας XPath είναι ότι μας προσφέρει τη δυνατότητα να χρησιμοποιήσουμε απλές μαθηματικές εκφράσεις, αφού έχει τη δυνατότητα να αναπαριστά και να διαχειρίζεται αριθμητικά και αλφαριθμητικές γραμματοσειρές. Να πούμε σε αυτό το σημείο πως η γλώσσα XPath ορίζει ένα έγγραφο XML ως ένα δέντρο με τα δομικά στοιχεία που ακολουθούν:

- Κόμβος-ρίζα(root node)
- Κόμβος-αντικείμενο(element node)
- Κόμβος κειμένου(text node)
- Κόμβος-ιδιότητα(attribute node)
- Κόμβος-σχόλιο(comment node)
- Κόμβος επεξεργασίας(processing – instruction node)
- Κόμβος ονομασίας(namespace node)

Στο δομικό στοιχείο-κόμβο που ψάχνουμε, η XPath καταλήγει ακολουθώντας το μονοπάτι(location path) που θα της δώσουμε, το οποίο είναι ένα δομικό στοιχείο ή ένα σύνολο δομικών στοιχείων του εγγράφου XML. Αν μιλάμε για σύνολο δομικών στοιχείων τότε αυτά στο μονοπάτι θα χωρίζονται με τον χαρακτήρα /(slash) που χρησιμοποιείται και για τον συμβολισμό του κόμβου ρίζα(root node). Επίσης, αν ο κόμβος του μονοπατιού είναι κόμβος-ιδιότητα(attribute node), πρέπει να χρησιμοποιήσουμε το πρόθεμα '@'. Ας δούμε όμως ένα παράδειγμα για καλύτερη κατανόηση:

```
<portal>
  <cameras>
    <digital>
      <cam brand="Canon" model="A60" price="550"></cam>
    </digital>
    <SLR>
      <cam brand="Canon" model="EOS-3" price="980"></cam>
    </SLR>
  </cameras>
</portal>
```

Αν εδώ θέλαμε να προσπελάσουμε όλες τις ψηφιακές μηχανές, το μονοπάτι που θα χρειαζόμασταν είναι:

```
/cameras/digital
```

Ενώ, αν θέλαμε τις τιμές για κάθε SLR μηχανές, θα είχαμε το μονοπάτι :

```
/cameras/SLR/cam/@price
```

Επίσης, θα μπορούσαμε να θέλουμε το αποτέλεσμά μας να είναι τα δομικά στοιχεία του XML εγγράφου τα οποία ικανοποιούν κάποια σχέση :

```
/cameras/SLR/cam[@brand = 'Canon' and @price >= 100]
```

Περνώντας τώρα στην γλώσσα XQuery, να πούμε αρχικά ότι το μοντέλο δεδομένων του στηρίζεται στο αντίστοιχο μοντέλο δεδομένων που χρησιμοποιεί και η γλώσσα XPath. Έχει τον κόμβο, που αντιστοιχεί στο δομικό στοιχείο του εγγράφου XML και την ατομική τιμή που μπορεί να είναι ένας από τους βασικούς τύπους δεδομένων, όπως ακέραιος(integer),συμβολοσειρά(string) κλπ. Αυτά αποτελούν το αντικείμενο(item) ενώ η ακολουθία(sequence) αποτελείται από ένα ταξινομημένο σύνολο ενός ή περισσότερων αντικειμένων. Επίσης, για τη δυνατότητα φιλτραρίσματος τιμών χρησιμοποιούμε το πρόθεμα, που είναι μία έκφραση η οποία περιέχεται σε ' [ ] '. Άρα αν υποθέσουμε πως το αρχείο του προηγούμενου παραδείγματος ονομαζόταν "portal.xml" και θέλουμε με XQuery να πάρουμε τις SLR κάμερες με αξία μεγαλύτερη των 100 ευρώ, θα είχαμε:

```
document("portal.xml")/cameras/SLR/cam[@brand = 'Canon' and @price >= 100]
```

Αναφέραμε όμως πως η γλώσσα XQuery αποτελεί επέκταση της γλώσσας XPath άρα περιμένουμε να μας προσφέρει κάτι παραπάνω σε σχέση με την XPath. Αυτό το παραπάνω είναι η δυνατότητα που μας δίνει να χρησιμοποιούμε εκφράσεις FLWOR, οι οποίες αποτελούνται από :

- For, τον επαναληπτικό βρόγχο
- Let, έκφραση ανάθεσης
- Where, έκφραση συνθήκης
- Order by, έκφραση ταξινόμησης
- Return, επιστροφή αποτελέσματος

Αυτή η βασική λειτουργία της γλώσσας XQuery λοιπόν μας βοήθησε ώστε να υλοποιήσουμε τις διαδικασίες που μας ζητήθηκαν και τις οποίες θα αναλύσουμε παρακάτω.

## 6.1 XQuery προσθήκης στοιχείου (element)

Αρχικά πρέπει να δημιουργήσουμε ένα έγγραφο XML το οποίο θα το συνδέσουμε με το XML Schema που δημιουργήσαμε στο κεφάλαιο 5, προκειμένου να υπακούει στους κανόνες που ορίσαμε, και το οποίο θα ενημερώνεται στις αλλαγές που θα κάνουμε μέσω XQuery. Η ένωση του εγγράφου XML με το XML Schema γίνεται περικλείοντας το XML έγγραφο στην ετικέτα(tag):

```
<SuperMarket xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="oxygen:/SuperMarket_DataSource$SuperMarket_on%20localhos
t8080/db/supermarket/SuperMarket.xsd">
  <!-- -XML έγγραφο -->
</SuperMarket>
```

Ξεκινάμε με την υλοποίηση των διαδικασιών που αφορούν την δυνατότητα προσθήκης των στοιχείων(elements) που συμμετέχουν στο XML Schema. Βασική προϋπόθεση για να έχει νόημα η προσθήκη στοιχείων, είναι να έχουμε δημιουργήσει αρχικά ένα ολοκληρωμένο SuperMarket με τουλάχιστον ένα στιγμιότυπο(instance) από κάθε στοιχείο(element). Ακολουθούν τα μέρη του Xquery που δημιουργήσαμε γι' αυτό το σκοπό.

```
1 declare function local:createApothiki($onoma as xs:string)
2 as element(apothiki) {
3   let $first := xs:QName('first')
4   return (element apothiki {
5     attribute onoma {$onoma}
6   });
```

Εικόνα 6.1: XQuery 'createSuperMarket', συνάρτηση 'createApothiki'

Ορίζουμε αρχικά μία συνάρτηση η οποία θα μας δημιουργήσει το στοιχείο(element) 'apothiki'. Στην γραμμή 1, ορίζουμε το όνομά της(createApothiki) και στις παρενθέσεις που ακολουθούν δηλώνουμε τα ορίσματα που πρέπει να χρησιμοποιήσουμε κατά την κλήση της, που είναι η παράμετρος<sup>5</sup> 'onoma' τύπου string. Στην γραμμή 2 δηλώνουμε τί θα επιστρέψει η συνάρτηση και στην περίπτωση μας επιστρέφει στοιχείο(element) τύπου 'apothiki'. Συνεχίζοντας στη γραμμή 3, έχουμε τη δήλωση μίας μεταβλητής τύπου QName που χρησιμοποιείται για τη δημιουργία έγκυρων αναγνωριστικών για στοιχεία και χαρακτηριστικά. Τέλος, στην γραμμή 4 έχουμε την επιστροφή του αποτελέσματος με την γραμμή 5 να βάζει το όρισμα της συνάρτησης ως το χαρακτηριστικό του στοιχείου(element) που επιστρέφει.

Συνεχίζουμε με τον ορισμό της συνάρτησης που θα δημιουργεί το σύνθετο στοιχείο(complex element) 'dieythinsi'. Η δήλωση της συνάρτησης αυτής αλλά και όλων των συναρτήσεων που θα ακολουθήσουν, με σκοπό τη δημιουργία κάποιου στοιχείου(element), έχουν την ίδια δομή με αυτή που αναλύσαμε για την 'createApothiki'. Ακολουθεί ένας συγκεντρωτικός πίνακας με τις συναρτήσεις

<sup>5</sup> Στις παραμέτρους αναφερόμαστε πάντα με το πρόθεμα '\$'.

δημιουργίας στοιχείων(elements) που χρησιμοποιήθηκαν στο συγκεκριμένο XQuery καθώς και ο κώδικάς τους.

Όνομα συνάρτησης	Ορίσματα	Στοιχείο(element) επιστροφής
<b>createDieythinsi</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string	dieythinsi
<b>createYpokatasthma</b>	1. \$sonoma as xs:string	ypokatasthma
<b>createYpallhlos</b>	2. \$a.t. as xs:string 3. \$sonomaterwnymo as xs:string 4. \$misthos as xs:float 5. \$phoneNumber as xs:string 6. \$seidosThlefwnou as xs:string	ypallhlos
<b>createParaggelia</b>	1. \$kwdikos as xs:string 2. \$posothta as xs:float	paraggelia
<b>CreateParaggeliaApothikis</b>	1. \$kwdikos as xs:string 2. \$posothta as xs:float 3. \$promhthyths as xs:string 4. \$timhAgoras as xs:float	paraggeliaApothikis
<b>createPromhtheyths</b>	1. \$a.f.m. as xs:string 2. \$sonomaterwnymo as xs:string 3. \$phoneNumber as xs:string 4. \$seidosThlefwnou as xs:string	promhtheyths
<b>createProion</b>	1. \$kwdikos as xs:string 2. \$perigraph as xs:string 3. \$seidos as xs:string 4. \$paragwgos as xs:string 5. \$promitheutisIDREF as xs:string 6. \$timh as xs:float 7. \$timhPwlshs as xs:float	proionta
<b>createApothema</b>	1. \$proionIDREF as xs:string 2. \$apothema as xs:float 3. \$minApothema as xs:float	proiontaApothema
<b>createpromhtheythsApothikis</b>	1. \$promitheutisIDREF as xs:string 2. \$timhPwlshs as xs:float	promhtheythsApothikis

Πίνακας 6.1: XQuery 'createuperMarket', συναρτήσεις δημιουργίας στοιχείων(elements)

Να επισημάνουμε πριν δούμε τον κώδικα, ένα σημείο για την 'paraggelia' και την 'paraggeliaApothikis'. Σαν ορίσματα δεν παίρνουν τον κωδικό της παραγγελίας και ο λόγος είναι γιατί οι παραγγελίες θέλουμε να παίρνουν έναν αύξοντα αριθμό για κωδικό τους. Οπότε στη συγκεκριμένη δημιουργία παραγγελίας, που είναι η πρώτη, θέλουμε να έχει τον αριθμό 1, ο οποίος στις επόμενες παραγγελίες θα αυξάνεται με βήμα 1. Έτσι, μέσα στο return της 'createparaggelia' και 'createparaggeliaApothikis', το γνώρισμα(attribute) 'kwdikosParaggelias' και 'kwdikosParaggeliasApothikis' αντίστοιχα, παίρνουν τη σταθερή τιμή 1.

```
declare function local:createDieythinsi ($odos as xs:string, $arithmos as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
```

```
    element Odos {$odos},
    element Arithmos{$arithmos},
    element Poli{$poli}
  });

declare function local:createYpokatasthma ($sonoma as xs:string)
as element (ypokatasthma) {
  let $first := xs:QName('first')
  return (element ypokatasthma {
    attribute onoma {$sonoma}
  });
});

declare function local:createYpallhlos ($a.t. as xs:string, $onomatepwnymo as xs:string, $misthos as
xs:float,$phoneNumber as xs:string, $eidosthlefwnou as xs:string)
as element (ypallhlos) {
  let $first := xs:QName('first')
  return (element ypallhlos {
    attribute at {$a.t.},
    element onomatepwnymo {$onomatepwnymo},
    element misthos {$misthos},
    element thlefwno {
      element eidosthlefwnou {$eidosthlefwnou},
      element number {$phoneNumber}
    }
  });
});

declare function local:createParaggelia ( $kwdikos as xs:string, $posothta as xs:float)
as element (paraggelia) {
  let $first := xs:QName('first')
  return (element paraggelia {
    attribute kwdikosParaggelias {1},
    element kwdikos {$kwdikos},
    element posothta {$posothta}
  });
});

declare function local:createParaggeliaApothikis ( $kwdikos as xs:string, $posothta as xs:float,
$promhthyths as xs:string,$timhAgoras as xs:float)
as element (paraggeliaApothikis) {
  let $first := xs:QName('first')
  return (element paraggeliaApothikis {
    attribute kwdikosParaggeliasApothikis {1},
    element kwdikos {$kwdikos},
    element posothta {$posothta},
    element promhtheyths {$promhthyths},
    element timhAgoras {$timhAgoras}
  });
});

declare function local:createPromhtheyths ($a.f.m. as xs:string,$onomatepwnymo as xs:string,
$phoneNumber as xs:string, $eidosthlefwnou as xs:string)
as element (promhtheyths) {
  let $first := xs:QName('first')
  return (element promhtheyths {
    element afm {$a.f.m.},
    element onomatepwnymo {$onomatepwnymo},
    element thlefwno {
```



```

        element eidosthlefwnou {$eidosthlefwnou},
        element number {$phoneNumber}
    }
});

declare function local:createProion ($kwdikos as xs:string, $perigraph as xs:string, $eidos as
xs:string,$paragwgos as xs:string,$promitheutisIDREF as xs:string, $timh as xs:float,$timhPwlshs as
xs:float )
as element (proionta) {
let $first := xs:QName('first')
return (element proionta {
    element proionKwdikos {$kwdikos},
    element perigraph {$perigraph},
    element eidos {$eidos},
    element paragwgos {$paragwgos},
    element timh {$timh},
    element promtheyths {
    element promitheutisIDREF {$promitheutisIDREF},
    element timhPwlshs {$timhPwlshs}
    }
});

declare function local:createApothema ($proionIDREF as xs:string, $apothema as xs:float,
$minApothema as xs:float)
as element (proiontaApothema) {
let $first := xs:QName('first')
return (element proiontaApothema {
    element proionIDREF {$proionIDREF},
    element apothema {$apothema},
    element minApothema {$minApothema}
});

declare function local:createpromtheythsApothikis ($promitheutisIDREF as xs:string, $timhPwlshs
as xs:float)
as element (promtheythsApothikis) {
let $first := xs:QName('first')
return (element promtheythsApothikis {
    element promitheutisIDREF {$promitheutisIDREF},
    element timhPwlshs {$timhPwlshs}
});

```

Για να υπακούει όμως το αποτέλεσμα του XQuery στο XML Schema θα πρέπει όλα αυτά τα στοιχεία(elements) που δημιουργούμε, να εισάγονται σε συγκεκριμένη θέση. Για το λόγο αυτό δημιουργήσαμε συναρτήσεις οι οποίες καλούν τις συναρτήσεις δημιουργίας που αναφέραμε και τοποθετούν το αποτέλεσμα, δηλαδή τα στοιχεία(element), στη θέση που ορίζουμε. Υπάρχουν τρεις εκφράσεις που χρησιμοποιούμε προκειμένου να δηλώσουμε τη θέση του στοιχείου(element) σε σχέση με τα υπόλοιπα στοιχεία(element):

- Into 'στοιχείο(element)', εισαγωγή ως τελευταίο παιδί του 'στοιχείου(element)'
- Following 'στοιχείο(element)', εισαγωγή αμέσως μετά το 'στοιχείο(element)'
- Preceding 'στοιχείο(element)', εισαγωγή πριν το 'στοιχείο(element)'

Ακολουθεί η συνάρτηση εισαγωγής για το στοιχείο(element) 'apothiki' και η περιγραφή της :

```

107 declare function local:insertApothiki ($onoma as xs:string) {
108 update insert local:createApothiki($onoma) into
109 doc('SuperMarket.xml')/SuperMarket
110 };

```

Εικόνα 6.2: XQuery createSuperMarket, συνάρτηση insertApothiki

Στη γραμμή 107 του Xquery συναντάμε την δήλωση της συνάρτησης 'insertApothiki' και το όρισμα που παίρνει από την κλήση της, που είναι η μεταβλητή 'onoma' τύπου string. Στην γραμμή 108 καλούμε την συνάρτηση 'createApothiki' και αυτό που θα μας επιστρέψει, δηλαδή το στοιχείο(element) 'apothiki', το εισάγουμε(insert) στο αρχείο 'SuperMarket.xml' ως τελευταίο παιδί(into) του στοιχείου(element) 'SuperMarket'(/SuperMarket). Με την λέξη update πριν από όλη αυτή την έκφραση δηλώνουμε πως θα μας επιστρέψει το 'SuperMarket' που θα προκύψει μετά την εισαγωγή του στοιχείου(element) 'apothiki'. Να σημειώσουμε πως στα ορίσματα της συνάρτησης εισαγωγής συμπεριλαμβάνουμε και τα ορίσματα που θα χρησιμοποιηθούν στην κλήση συνάρτησης δημιουργίας του στοιχείου(element). Στην περίπτωση εισαγωγής της αποθήκης έτυχε να παίρνουν και οι δύο συναρτήσεις για όρισμα το 'onoma' για την 'apothiki' αλλά αυτό δε συμβαίνει πάντα. Ακολουθώντας την ίδια μέθοδο και για τα υπόλοιπα στοιχεία, δημιουργήσαμε και γι' αυτά συναρτήσεις εισαγωγής, ακολουθεί συγκεντρωτικός πίνακας καθώς και ο κώδικας:

Όνομα συνάρτησης	Ορίσματα	Θέση στοιχείου(element) που επιστρέφει η αντίστοιχη συνάρτηση δημιουργίας
<b>insertDieythinsiApothikis</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$onoma as xs:string</li> </ol>	Into doc('SuperMarket.xml') /SuperMarket/apothiki[@onoma=\$onoma]
<b>insertYpokatasthma</b>	<ol style="list-style-type: none"> <li>1. \$onomaYpokata as xs:string</li> <li>2. \$onomaApothikis as xs:string</li> </ol>	Into doc('SuperMarket.xml') /SuperMarket/apothiki[@onoma=\$onomaApothikis]
<b>insertDieythinsiYpokatasthma</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$onomaYpokatasthmatos as xs:string</li> </ol>	Into doc('SuperMarket.xml') //ypokatasthma[@onoma=\$onomaYpokatasthmatos]
<b>insertYpallhlo</b>	<ol style="list-style-type: none"> <li>1. \$a.t. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$misthos as xs:float</li> <li>4. \$phoneNumber as xs:string</li> <li>5. \$seidosThlefwnou as xs:string</li> <li>6. \$ypokatasthma as xs:string</li> </ol>	Into doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma]
<b>insertDieythinsiYpallhlou</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> </ol>	Following doc('SuperMarket.xml') //ypallhlos[@at=\$a.t.Ypallhlou]/misthos

<b>insertParaggeliaYpokatasthmatos</b>	<ol style="list-style-type: none"> <li>4. \$a.t.Ypallhlou as xs:string</li> <li>1. \$kwidikos as xs:string</li> <li>2. \$posothta as xs:float</li> <li>3. \$ypokatasthma as xs:string</li> </ol>	Following doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma]/ypallhlous
<b>insertParaggeliaApothikis</b>	<ol style="list-style-type: none"> <li>1. \$kwidikos as xs:string</li> <li>2. \$posothta as xs:float</li> <li>3. \$promhtheyth as xs:string</li> <li>4. \$timhAgoras as xs:float</li> <li>5. \$apothiki as xs:string</li> </ol>	Following doc('SuperMarket.xml') //apothiki[@onoma=\$apothiki]/promhtheythsApothikis
<b>insertApothemaKatasthmatos</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$ypokatasthma as xs:string</li> </ol>	Following doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma]/paraggelia
<b>insertApothemaApothikis</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$apothiki as xs:string</li> </ol>	Following doc('SuperMarket.xml') //apothiki[@onoma=\$apothiki]/ypokatasthma
<b>insertPromhtheythsApothikis</b>	<ol style="list-style-type: none"> <li>1. \$promitheutisIDREF as xs:string</li> <li>2. \$timhPwlhshs as xs:float</li> <li>3. \$apothiki as xs:string</li> </ol>	Following doc('SuperMarket.xml') //apothiki[@onoma=\$apothiki]/proiontaApothema
<b>insertPromhtheyths</b>	<ol style="list-style-type: none"> <li>1. \$a.f.m. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$phoneNumber as xs:string</li> <li>4. \$eidosthlefwnou as xs:string</li> </ol>	Following doc('SuperMarket.xml') /SuperMarket/proionta
<b>insertDieythinsiPromhtheyth</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$a.f.mPromhtheyth as xs:string</li> </ol>	Following doc('SuperMarket.xml') //promhtheyths[afm=\$a.f.mPromhtheyth]/onomatepwnymo
<b>insertProion</b>	<ol style="list-style-type: none"> <li>1. \$kwidikos as xs:string</li> </ol>	Following doc('SuperMarket.xml')/SuperMarket/apothiki

2. \$perigraph as  
xs:string
3. \$eidos as xs:string
4. \$paragwgos as  
xs:string
5. \$promitheutisIDRE  
F as xs:string
6. \$timh as xs:float
7. \$timhPwlshs as  
xs:float

Πίνακας 6.2: XQuery 'createSuperMarket', συναρτήσεις εισαγωγής

```

declare function local:insertDieythinsiApothikis ($odos as xs:string, $arithmos as xs:integer, $poli as
xs:string, $sonoma as xs:string){
update insert local:createDieythinsi($odos, $arithmos, $poli) into
doc('SuperMarket.xml')/SuperMarket/apothiki[ @onoma=$sonoma]
};

declare function local:insertYpokatasthma ($sonomaYpokata as xs:string,$sonomaApothikis as
xs:string){
update insert local:createYpokatasthma($sonomaYpokata) into
doc('SuperMarket.xml')/SuperMarket/apothiki[ @onoma=$sonomaApothikis]
};

declare function local:insertDieythinsiYpokatasthma ($odos as xs:string, $arithmos as xs:integer, $poli
as xs:string,$sonomaYpokatasthmatos as xs:string){
update insert local:createDieythinsi($odos, $arithmos,$poli) into
doc('SuperMarket.xml')//ypokatasthma[ @onoma=$sonomaYpokatasthmatos]
};

declare function local:insertYpallhlo ($a.t. as xs:string, $sonomatepwnymo as xs:string, $misthos as
xs:float,$phoneNumber as xs:string,$eidosThlefwnou as xs:string, $ypokatasthma as xs:string){
update insert
local:createYpallhlos($a.t.,$sonomatepwnymo,$misthos,$phoneNumber,$eidosThlefwnou) into
doc('SuperMarket.xml')//ypokatasthma[ @onoma=$ypokatasthma]
};

declare function local:insertDieythinsiYpallhlou ($odos as xs:string, $arithmos as xs:integer, $poli as
xs:string,$a.t.Ypallhlou as xs:string) {
update insert local:createDieythinsi($odos, $arithmos,$poli) following
doc('SuperMarket.xml')//ypallhlos[ @at=$a.t.Ypallhlou]/misthos
};

declare function local:insertParaggeliaYpokatasthmatos ($kwdikos as xs:string, $posothta as xs:float,
$ypokatasthma as xs:string ){
update insert local:createParaggelia ($kwdikos , $posothta ) following
doc('SuperMarket.xml')//ypokatasthma[ @onoma=$ypokatasthma]/ypallhlos
};

declare function local:insertParaggeliaApothikis ($kwdikos as xs:string, $posothta as
xs:float,$promhtheyth as xs:string,$timhAgoras as xs:float, $apothiki as xs:string ){
update insert local:createParaggeliaApothikis($kwdikos , $posothta,$promhtheyth,$timhAgoras )
following doc('SuperMarket.xml')//apothiki[ @onoma=$apothiki]/promhtheythsApothikis
};

```

```

declare function local:insertApothemaKatasthmatos ($proionIDREF as xs:string, $apothema as
xs:float,$minApothema as xs:float, $ypokatasthma as xs:string){
update insert local:createApothema($proionIDREF,$apothema,$minApothema ) following
doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia
};
declare function local:insertApothemaApothikis ($proionIDREF as xs:string, $apothema as
xs:float,$minApothema as xs:float, $apothiki as xs:string){
update insert local:createApothema($proionIDREF,$apothema, $minApothema ) following
doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/ypokatasthma
};
declare function local:insertPromhtheythsApothikis ($promitheutisIDREF as xs:string, $timhPwlshs
as xs:float,$apothiki as xs:string){
update insert local:createpromhtheythsApothikis($promitheutisIDREF , $timhPwlshs ) following
doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/proiontaApothema
};

declare function local:insertPromhtheyths ($a.f.m. as xs:string,$onomatepwnymo as xs:string,
$phoneNumber as xs:string, $seidosThlefwnou as xs:string){
update insert local:createPromhtheyths($a.f.m., $onomatepwnymo , $phoneNumber , $seidosThlefwnou)
following doc('SuperMarket.xml')/SuperMarket/proionta
};

declare function local:insertDieythinsiPromhtheyth ($sodos as xs:string, $arithmos as xs:integer, $poli
as xs:string,$a.f.mPromhtheyth as xs:string) {
update insert local:createDieythinsi($sodos, $arithmos,$poli) following
doc('SuperMarket.xml')//promhtheyths[afm=$a.f.mPromhtheyth]/onomatepwnymo
};

declare function local:insertProion ($kwdikos as xs:string, $perigraph as xs:string, $seidos as xs:string,
$paragwgos as xs:string, $promitheutisIDREF as xs:string, $timh as xs:float,$timhPwlshs as xs:float)
{
update insert local:createProion($kwdikos , $perigraph , $seidos , $paragwgos , $promitheutisIDREF ,
$timh,$timhPwlshs) following doc('SuperMarket.xml')/SuperMarket/apothiki
};

```

Τελειώνοντας το XQuery createSuperMarket , κλείνουμε με τις κλήσεις των συναρτήσεων εισαγωγής, μέσα σε ετικέτες επιστροφής(return tags), χρησιμοποιώντας τις τιμές που επιθυμούμε εμείς στα ορίσματα.

```

<return>
{local:insertApothiki ('Αποθήκη_1')}
{local:insertDieythinsiApothikis ('Οδός1_Αποθήκης', 1, 'Πόλη1_Αποθήκης', 'Αποθήκη_1')}
{local:insertYpokatasthma
  ('Υποκατάστημα_1', 'Αποθήκη_1')}
{local:insertDieythinsiYpokatasthma
  ('Οδός1_Υποκαταστήματος', 12, 'Πόλη1_Υποκαταστήματος', 'Υποκατάστημα_1' )}
{local:insertYpallhlo
  ('AA 111111', 'Υπάλληλος_1' ,652.00, '6978123654', 'κινητό', 'Υποκατάστημα_1')}
{local:insertDieythinsiYpallhlou
  ('Οδός1_Υπαλλήλου', 15, 'Πόλη1_Υπαλλήλου', 'AA 111111')}
{local:insertParaggeliaYpokatasthmatos
  ('ID1122',10.00, 'Υποκατάστημα_1')}
{local:insertApothemaKatasthmatos
  ('ID1122',25.00,20.00, 'Υποκατάστημα_1' )}
{local:insertApothemaApothikis
  ('ID1122',50.00,40.00, 'Αποθήκη_1' )}
{local:insertPromhtheythsApothikis
  ('ID3344',15.00, 'Αποθήκη_1' )}
{local:insertParaggeliaApothikis
  ('ID3344',15.00, 'ID3344',15.0, 'Αποθήκη_1' )}
{local:insertProion
  ('ID1122', 'Περιγραφή1', 'Είδος1', 'Παραγωγός1', 'ID3344',2.00,2.50)}
{local:insertPromhtheyths
  ('ID3344', 'Προμηθευτής1', '2810235689', 'FAX')}
{local:insertDieythinsiPromhtheyth
  ('Οδός1_Προμηθευτή',15, 'Πόλη1_Προμηθευτή', 'ID3344')}
</return>

```

Εικόνα 6.3:Return of XQuery createSuperMarket

Στο σημείο αυτό έχουμε δημιουργήσει ένα έγγραφο XML που ακολουθεί πιστά τους κανόνες του XML Schema που δημιουργήσαμε στο κεφάλαιο 5. Τα υπόλοιπα XQuery που θα δούμε έχουν σκοπό να ‘κρεμάσουν’ πάνω στο ήδη υπάρχον XML έγγραφο, περισσότερα του ενός στοιχεία(elements). Να μπορούμε για παράδειγμα να προσθέσουμε έναν υπάλληλο σε ένα υποκατάστημα ή μία αποθήκη στο SuperMarket ή έναν προμηθευτή στην αποθήκη κ.τ.λ. Όπως καταλαβαίνουμε λοιπόν, οι συναρτήσεις δημιουργίας και εισαγωγής που δημιουργήσαμε για το XQuery ‘createSuperMarket’ θα μας χρειαστούν κι εδώ. Τα καινούρια αυτά Xquery θα είναι σαν μικρά κομμάτια από το XQuery ‘createSuperMarket’ αλλά θα αλλάζει λίγο η θέση στην οποία θα τοποθετούμε τα στοιχεία(elements), αφού δε μιλάμε πλέον για ένα κενό έγγραφο XML. Ακολουθούν πίνακες με πληροφορίες για τις συναρτήσεις απο κάθε Xquery προσθήκης στοιχείου(element) αλλά και ολόκληρος ο κώδικάς τους.

➤ Xquery ‘addApothiki’

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createApothiki</b>	1. \$onoma as xs:string	στοιχείο(element) ‘apothiki’
<b>createDieythinsi</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string	στοιχείο(element) ‘dieythinsi’
<b>insertApothiki</b>	1. \$onoma as xs:string	Το στοιχείο(element) ‘apothiki’ στη θέση: preceding doc('SuperMarket.xml') /SuperMarket/apothiki[1]
<b>insertDieythinsiApothikis</b>	2. \$odos as xs:string 3. \$arithmos as xs:integer 4. \$poli as xs:string 5. \$onoma as xs:string	Το στοιχείο(element) ‘dieythinsi’ στη θέση: into doc('SuperMarket.xml') /SuperMarket/apothiki[@onoma=\$onoma]

Πίνακας 6.3: XQuery 'addApothiki', συναρτήσεις

```

1 declare function local:createApothiki($onoma as xs:string)
2 as element(apothiki) {
3   let $first := xs:QName('first')
4   return (element apothiki {
5     attribute onoma {$onoma}
6   });
7
8 declare function local:createDieythinsi ($odos as xs:string,
9   $arithmos as xs:integer, $poli as xs:string)
10 as element (dieythinsi) {
11   let $first := xs:QName('first')
12   return (element dieythinsi {
13     element Odos {$odos},
14     element Arithmos{$arithmos},
15     element Poli{$poli}
16   });
17
18
19 declare function local:insertApothiki ($onoma as xs:string){
20   update insert local:createApothiki($onoma) preceding
21   doc('SuperMarket.xml')/SuperMarket/apothiki[1]
22 };
23
24 declare function local:insertDieythinsiApothikis ($odos as xs:string,
25   $arithmos as xs:integer, $poli as xs:string, $onoma as xs:string){
26   update insert local:createDieythinsi($odos, $arithmos, $poli)
27   into doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$onoma]
28 };
29
30 <return>
31 {local:insertApothiki('Αποθήκη_2')}
32 {local:insertDieythinsiApothikis('Οδός2_Αποθήκης', 2, 'Πόλη2_Αποθήκης', 'Αποθήκη_2')}
33 </return>

```

Εικόνα 6.4: Xquery 'addApothiki'

➤ Xquery 'addCompleteApothiki'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createApothiki</b>	1. \$onoma as xs:string	Στοιχείο(element) 'apothiki'
<b>createDieythinsi</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string	Στοιχείο(element) 'dieythinsi'
<b>createYpokatasthma</b>	1. \$onoma as xs:string	Στοιχείο(element) 'ypokatasthma'
<b>createYpallhlos</b>	1. \$a.t. as xs:string 2. \$sonomatepwny mo as xs:string 3. \$misthos as xs:float 4. \$phoneNumber as xs:string 5. \$seidosThlefwno u as xs:string	Στοιχείο(element) 'ypallhlos'
<b>createParaggelia</b>	1. \$skwdikos as	Στοιχείο(element) 'paraggelia'

	<ul style="list-style-type: none"> <li>xs:int</li> <li>2. \$posothta as xs:float</li> </ul>	
<b>createApothema</b>	<ul style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> </ul>	Στοιχείο(element) 'proiontaApothema'
<b>createpromhtheythsApothikis</b>	<ul style="list-style-type: none"> <li>1. \$promitheutisIDREF as xs:string</li> <li>2. \$timhPwlhshs as xs:float</li> </ul>	Στοιχείο(element) 'promhtheythsApothikis'
<b>createParaggeliaApothikis</b>	<ul style="list-style-type: none"> <li>1. \$onomaApothikis as xs:string</li> <li>2. \$kwdikos as xs:string</li> <li>3. \$posothta as xs:float</li> <li>4. \$promhthyths as xs:string</li> <li>5. \$timhAgoras as xs:float</li> </ul>	Στοιχείο(element) 'paraggeliaApothikis'
<b>insertApothiki</b>	<ul style="list-style-type: none"> <li>1. \$onoma as xs:string</li> </ul>	Στοιχείο(element) 'apothiki' στη θέση: preceding doc('SuperMarket.xml')/SuperMarket/proionta
<b>insertDieythinsiApothikis</b>	<ul style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$onoma as xs:string</li> </ul>	Στοιχείο(element) 'dieythinsi' στη θέση: into doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=\$onoma]
<b>insertYpokatasthma</b>	<ul style="list-style-type: none"> <li>1. \$onomaYpokata as xs:string</li> <li>2. \$onomaApothikis as xs:string</li> </ul>	Στοιχείο(element) 'ypokatasthma' στη θέση: into doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=\$onomaApothikis]
<b>insertDieythinsiYpokatasthma</b>	<ul style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$onomaYpokatasthmatos as xs:string</li> </ul>	Στοιχείο(element) 'dieythinsi' στη θέση: into doc('SuperMarket.xml')/ypokatasthma[@onoma=\$onomaYpokatasthmatos]
<b>insertYpallhlo</b>	<ul style="list-style-type: none"> <li>1. \$a.t. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$misthos as xs:float</li> <li>4. \$phoneNumber as xs:string</li> <li>5. \$eidosthlefwno</li> </ul>	Στοιχείο(element) 'ypallhlo' στη θέση: into doc('SuperMarket.xml')/ypokatasthma[@onoma=\$ypokatasthma]



	u as xs:string	
	6. \$ypokatasthma as xs:string	
<b>insertDieythinsiYpallh lou</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$a.t.Ypallhlou as xs:string</li> </ol>	Στοιχείο(element) 'dieythinsi' στη θέση: following doc('SuperMarket.xml')//ypallhlos[@at=\$a.t.Ypallhlu]/misthos
<b>insertParaggeliaYpoka tasthmatos</b>	<ol style="list-style-type: none"> <li>1. \$kwdikos as xs:int</li> <li>2. \$posothta as xs:float</li> <li>3. \$ypokatasthma as xs:string</li> </ol>	Στοιχείο(element) 'paraggelia' στη θέση: following doc('SuperMarket.xml')//ypokatasthma[@onoma=\$ypokatasthma]/ypallhlos
<b>insertApothemaKatast hmatos</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$ypokatasthma as xs:string</li> </ol>	Στοιχείο(element) 'apothema' στη θέση: following doc('SuperMarket.xml')//ypokatasthma[@onoma=\$ypokatasthma]/paraggelia
<b>insertApothemaApothi kis</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$apothiki as xs:string</li> </ol>	Στοιχείο(element) 'apothema' στη θέση: following doc('SuperMarket.xml')//apothiki[@onoma=\$apothiki]/ypokatasthma
<b>insertPromtheythsApothi khis</b>	<ol style="list-style-type: none"> <li>1. \$promitheutisIDREF as xs:string</li> <li>2. \$timhPwlhshs as xs:float</li> <li>3. \$apothiki as xs:string</li> </ol>	Στοιχείο(element) 'promtheythsApothikis' στη θέση: following doc('SuperMarket.xml')//apothiki[@onoma=\$apothiki]/proiontaApothema
<b>insertParaggeliaApothi khis</b>	<ol style="list-style-type: none"> <li>1. \$onomaApothikis as xs:string</li> <li>2. \$kwdikos as xs:string</li> <li>3. \$posothta as xs:float</li> <li>4. \$promtheyth as xs:string</li> <li>5. \$timhAgoras as xs:float</li> </ol>	Στοιχείο(element) 'paraggeliaApothikis' στη θέση: following doc('SuperMarket.xml')//apothiki[@onoma=\$apothiki]/promtheythsApothikis

Πίνακας 6.4: XQuery 'addCompleteApothiki', συναρτήσεις

```
declare function local:createApothiki($onoma as xs:string)
as element(apothiki) {
let $first := xs:QName('first')
return (element apothiki {
```

```
    attribute onoma {$onoma}
  });

declare function local:createDieythinsi
($sodos as xs:string, $arithmos as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
  element Odos {$sodos},
  element Arithmos{$arithmos},
  element Poli{$poli}
});

declare function local:createYpokatasthma ($sonoma as xs:string)
as element (ypokatasthma) {
let $first := xs:QName('first')
return (element ypokatasthma {
  attribute onoma {$sonoma}
});

declare function local:createYpallhlos
($a.t. as xs:string, $sonomatepwnymo as xs:string,
$misthos as xs:float,$phoneNumber as xs:string, $seidosThlefwnou as xs:string)
as element (ypallhlos) {
let $first := xs:QName('first')
return (element ypallhlos {
  attribute at {$a.t.},
  element onomatepwnymo {$sonomatepwnymo},
  element misthos {$misthos},
  element thlefwno {
    element eidosthlefwnou {$seidosThlefwnou},
    element number {$phoneNumber}
  }
});

declare function local:createParaggelia (
$kw dikos as xs:int, $posothta as xs:float)
as element (paraggelia) {
let $first := xs:QName('first')
return (element paraggelia {
  attribute kw dikosParaggelias {1},
  element kw dikos {$kw dikos},
  element posothta {$posothta}
});

declare function local:createApothema($proionIDREF as xs:string,
$apothema as xs:float, $minApothema as xs:float)
as element (proiontaApothema) {
let $first := xs:QName('first')
return (element proiontaApothema {
  element proionIDREF {$proionIDREF},
  element apothema {$apothema},
  element minApothema {$minApothema}
});
```

```
declare function local:createPromhtheythsApothikis
($promitheutisIDREF as xs:string, $timhPwlhshs as xs:float)
as element (promhtheythsApothikis) {
let $first := xs:QName('first')
return (element promhtheythsApothikis {
    element promitheutisIDREF {$promitheutisIDREF},
    element timhPwlhshs {$timhPwlhshs}
}});

declare function local:createParaggeliaApothikis
($onomaApothikis as xs:string,$kwdikos as xs:string,
$posothta as xs:float,$promhthyths as xs:string,$timhAgoras as xs:float)
as element (paraggeliaApothikis) {
let $first := xs:QName('first')
return (element paraggeliaApothikis {
    attribute onoma {$onomaApothikis}
    attribute kwdikosParaggeliasApothikis {1},
    element kwdikos {$kwdikos},
    element posothta {$posothta},
    element promhtheyths {$promhthyths},
    element timhAgoras {$timhAgoras}
}});

declare function local:insertApothiki ($onoma as xs:string){
update insert local:createApothiki($onoma) preceding
doc('SuperMarket.xml')/SuperMarket/proionta
};

declare function local:insertDieythinsiApothikis
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string, $onoma as xs:string){
update insert local:createDieythinsi($odos, $arithmos, $poli)
into doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$onoma]
};

declare function local:insertYpokatasthma
($onomaYpokata as xs:string,$onomaApothikis as xs:string){
update insert local:createYpokatasthma($onomaYpokata)
into doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$onomaApothikis]
};

declare function local:insertDieythinsiYpokatasthma
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string,$onomaYpokatasthmatos as xs:string) {
update insert local:createDieythinsi($odos, $arithmos,$poli)
into doc('SuperMarket.xml')//ypokatasthma[@onoma=$onomaYpokatasthmatos]
};

declare function local:insertYpallhlo
($a.t. as xs:string, $sonomatepwnymo as xs:string, $misthos as xs:float,
$phoneNumber as xs:string, $seidosThlefwnou as xs:string, $ypokatasthma as xs:string){
update insert
local:createYpallhlos($a.t.,$sonomatepwnymo,$misthos,$phoneNumber,$seidosThlefwnou)
into doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]
};
```

```

declare function local:insertDieythinsiYpallhlou
($sodos as xs:string, $arithmos as xs:integer, $poli as xs:string,$a.t.Ypallhlou as xs:string) {
update insert local:createDieythinsi($sodos, $arithmos,$poli)
following doc('SuperMarket.xml')//ypallhlos[@at=$a.t.Ypallhlou]/misthos
};

declare function local:insertParaggeliaYpokatasthmatos
$skwdikos as xs:int, $posothta as xs:float, $ypokatasthma as xs:string ){
update insert local:createParaggelia($skwdikos , $posothta )
following doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/ypallhlos
};

declare function local:insertApothemaKatasthmatos
($sproionIDREF as xs:string, $apothema as xs:float,$minApothema as xs:float, $ypokatasthma as
xs:string){
update insert local:createApothema($sproionIDREF,$apothema,$minApothema )
following doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia
};

declare function local:insertApothemaApothikis
($sproionIDREF as xs:string, $apothema as xs:float,$minApothema as xs:float, $apothiki as xs:string){
update insert local:createApothema($sproionIDREF,$apothema, $minApothema )
following doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/ypokatasthma
};

declare function local:insertPromhtheythsApothikis
($promitheutisIDREF as xs:string, $timhPwlhshs as xs:float,$apothiki as xs:string){
update insert local:createpromhtheythsApothikis($promitheutisIDREF , $timhPwlhshs )
following doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/proiontaApothema
};

declare function local:insertParaggeliaApothikis
($onomaApothikis as xs:string,$skwdikos as xs:string, $posothta as xs:float,
$promhtheyth as xs:string,$timhAgoras as xs:float){
update insert local:createParaggeliaApothikis($onomaApothikis, $skwdikos ,
$posothta,$promhtheyth,$timhAgoras)
following doc('SuperMarket.xml')//apothiki[@onoma=$ onomaApothikis]/promhtheythsApothikis
};
<return>
{local:insertApothiki('Αποθήκη_2')}
{local:insertDieythinsiApothikis('Οδός1_Αποθήκης', 1, 'Πόλη1_Αποθήκης', 'Αποθήκη_2')}
{local:insertYpokatasthma('Υποκατάστημα_3','Αποθήκη_2')}
{local:insertDieythinsiYpokatasthma('Οδός1_Υποκαταστήματος', 12,
'Πόλη1_Υποκαταστήματος','Υποκατάστημα_3' )}
{local:insertYpallhlo('AA 3','Υπάλληλος_1' ,652.00,'6978123654','κινητό','Υποκατάστημα_3')}
{local:insertDieythinsiYpallhlou('Οδός1_Υπαλλήλου', 15, 'Πόλη1_Υπαλλήλου', 'AA 3')}
{local:insertParaggeliaYpokatasthmatos('ID1122',10.00,'Υποκατάστημα_3')}
{local:insertApothemaKatasthmatos('ID1122',15.00,20.00,'Υποκατάστημα_3' )}
{local:insertApothemaApothikis('ID1122',15.00,20.00,'Αποθήκη_2' )}
{local:insertPromhtheythsApothikis('ID3344',15.00,'Αποθήκη_2' )}
{local:insertParaggeliaApothikis('Αποθήκη_2', 'ID1122',15.00,'ID3344',15.0)}</return>

```

## ➤ Xquery 'addYpokatasthma'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createYpokatasthma</b>	1. \$onoma as xs:string	Στοιχείο(element) 'ypokatasthma'
<b>createDieythinsi</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string	Στοιχείο(element) 'dieythinsi'
<b>insertYpokatasthma</b>	1. \$onomaYpokata as xs:string 2. \$onomaApothikis as xs:string	Στοιχείο(element) 'ypokatasthma' στη θέση: preceding doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=\$onomaApothikis]/ypokatasthma[1]
<b>insertDieythinsiYpokatasthma</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string 4. \$onomaYpokatasthmatos as xs:string	Στοιχείο(element) 'dieythinsi' στη θέση into doc('SuperMarket.xml')//ypokatasthma[@onoma=\$onomaYpokatasthmatos]

Πίνακας 6.5: XQuery 'addYpokatasthma', συναρτήσεις

```

declare function local:createYpokatasthma ($onoma as xs:string)
as element (ypokatasthma) {
let $first := xs:QName('first')
return (element ypokatasthma {
attribute onoma { $onoma }
});

declare function local:createDieythinsi
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
element Odos { $odos },
element Arithmos { $arithmos },
element Poli { $poli }
});

declare function local:insertYpokatasthma ($onomaYpokata as xs:string,$onomaApothikis as
xs:string){
update insert local:createYpokatasthma($onomaYpokata)
preceding
doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$onomaApothikis]/ypokatasthma[1]
};

declare function local:insertDieythinsiYpokatasthma
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string,$onomaYpokatasthmatos as xs:string) {
update insert local:createDieythinsi($odos, $arithmos,$poli)
into doc('SuperMarket.xml')//ypokatasthma[@onoma=$onomaYpokatasthmatos]
};
<return>
{local:insertYpokatasthma('Υποκατάστημα_3','Αποθήκη_1')}
{local:insertDieythinsiYpokatasthma('Οδός2_Υποκαταστήματος', 32,
'Πόλη2_Υποκαταστήματος','Υποκατάστημα_3')}
</return>

```

➤ Xquery 'addCompleteYpokatasthma'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createYpokatasthma</b>	1. \$onoma as xs:string	Στοιχείο(element) 'ypokatasthma'
<b>createYpallhlos</b>	1. \$a.t. as xs:string 2. \$onomatepwnymo as xs:string 3. \$misthos as xs:float 4. \$phoneNumber as xs:string 5. \$eidosthlefwnou as xs:string	Στοιχείο(element) 'ypallhlos'
<b>createDieythinsi</b>	1. \$odos as xs:string 2. \$arithmos as xs:integer 3. \$poli as xs:string	Στοιχείο(element) 'dieythinsi'
<b>createParaggelia</b>	1. \$kwdikos as xs:int 2. \$posothta as xs:float	Στοιχείο(element) 'paraggelia'
<b>createApothema</b>	1. \$proionIDREF as xs:string 2. \$apothema as xs:float 3. \$minApothema as xs:float	Στοιχείο(element) 'proiontaApothema'
<b>insertYpokatasthma</b>	1. \$onomaYpokata as xs:string 2. \$onomaApothikis as xs:string	Στοιχείο(element) 'ypokatasthma' στη θέση: preceding doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=\$onomaApothikis]/ypokatasthma[1]
<b>insertDieythinsiYpokatasthma</b>	1. \$odos as xs:string, \$arithmos as xs:integer 2. \$poli as xs:string 3. \$onomaYpokatasthmatos as xs:string	Στοιχείο(element) 'dieythinsi' στη θέση: into doc('SuperMarket.xml')//ypokatasthma[@onoma=\$onomaYpokatasthmatos]
<b>insertYpallhlo</b>	1. \$a.t. as xs:string 2. \$onomatepwnymo as xs:string 3. \$misthos as xs:float 4. \$phoneNumber as xs:string 5. \$eidosthlefwnou as xs:string 6. \$ypokatasthma	Στοιχείο(element) 'ypallhlos' στη θέση: into doc('SuperMarket.xml')//ypokatasthma[@onoma=\$ypokatasthma]

<b>insertDieythinsiYpallhlou</b>	<ul style="list-style-type: none"> <li>as xs:string</li> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$a.t.Ypallhlou as xs:string</li> </ul>	Στοιχείο(element) 'dieythinsi' στη θέση: following doc('SuperMarket.xml')//ypallhlos[@at=\$a.t.Ypallhlou]/misthos
<b>insertParaggeliaYpokatasthmatos</b>	<ul style="list-style-type: none"> <li>1. \$ypokatasthma as xs:string</li> <li>2. \$kwdikos as xs:string</li> <li>3. \$posothta as xs:float</li> </ul>	Στοιχείο(element) 'paraggelia' στη θέση: following doc('SuperMarket.xml')//ypokatasthma[@onoma=\$ypokatasthma]/ypallhlos
<b>insertApothemaKatasthmatos</b>	<ul style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$ypokatasthma as xs:string</li> </ul>	Στοιχείο(element) 'proionta' στη θέση: following doc('SuperMarket.xml')//ypokatasthma[@onoma=\$ypokatasthma]/paraggelia

Πίνακας 6.6: XQuery 'addCompleteYpokatasthma', συναρτήσεις

```

declare function local:createYpokatasthma ($onoma as xs:string)
as element (ypokatasthma) {
let $first := xs:QName('first')
return (element ypokatasthma {
attribute onoma {$onoma}
});
}

declare function local:createYpallhlos
($a.t. as xs:string, $onomatepwnymo as xs:string,
$misthos as xs:float,$phoneNumber as xs:string, $eidosthlefwnou as xs:string)
as element (ypallhlos) {
let $first := xs:QName('first')
return (element ypallhlos {
attribute at {$a.t.},
element onomatepwnymo {$onomatepwnymo},
element misthos {$misthos},
element thlefwno {
element eidosthlefwnou {$eidosthlefwnou},
element number {$phoneNumber}
}
});
}

declare function local:createDieythinsi
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
element Odos {$odos},
element Arithmos{$arithmos},
element Poli{$poli}
});
}

```

```
});  
  
declare function local:createParaggelia  
($kwdikos as xs:int, $posothta as xs:float)  
as element (paraggelia) {  
  let $first := xs:QName('first')  
  return (element paraggelia {  
    attribute kwdikosParaggelias {1},  
    element kwdikos {$kwdikos},  
    element posothta {$posothta}  
  });  
});  
  
declare function local:createApothema($proionIDREF as xs:string, $apothema as xs:float,  
$minApothema as xs:float)  
as element (proiontaApothema) {  
  let $first := xs:QName('first')  
  return (element proiontaApothema {  
    element proionIDREF {$proionIDREF},  
    element apothema {$apothema},  
    element minApothema {$minApothema}  
  });  
});  
  
declare function local:insertYpokatasthma ($onomaYpokata as xs:string,$onomaApothikis as  
xs:string){  
  update insert local:createYpokatasthma($onomaYpokata)  
  preceding  
  doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$onomaApothikis]/ypokatasthma[1]  
};  
  
declare function local:insertDieythinsiYpokatasthma  
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string,$onomaYpokatasthmatos as xs:string) {  
  update insert local:createDieythinsi($odos, $arithmos,$poli)  
  into doc('SuperMarket.xml')//ypokatasthma[@onoma=$onomaYpokatasthmatos]  
};  
declare function local:insertYpallhlo  
($a.t. as xs:string, $onomatepwnymo as xs:string, $misthos as xs:float,$phoneNumber as xs:string,  
$eidosthlefwnou as xs:string, $ypokatasthma as xs:string){  
  update insert  
  local:createYpallhlos($a.t.,$onomatepwnymo,$misthos,$phoneNumber,$eidosthlefwnou)  
  into doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]  
};  
  
declare function local:insertDieythinsiYpallhlou  
($odos as xs:string, $arithmos as xs:integer, $poli as xs:string,$a.t.Ypallhlou as xs:string) {  
  update insert local:createDieythinsi($odos, $arithmos,$poli)  
  following doc('SuperMarket.xml')//ypallhlos[@at=$a.t.Ypallhlou]/misthos  
};  
  
declare function local:insertParaggeliaYpokatasthmatos($ypokatasthma as xs:string,  
$kwdikos as xs:string, $posothta as xs:float ){  
  update insert local:createParaggelia( $kwdikos , $posothta )  
  following doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/ypallhlos
```



```
};

declare function local:insertApothemaKatasthmatos
($proionIDREF as xs:string, $apothema as xs:float,$minApothema as xs:float, $ypokatasthma as
xs:string){
update insert local:createApothema($proionIDREF,$apothema,$minApothema )
following doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia
};

<return>
{local:insertYpokatasthma('Υποκατάστημα_2','Αποθήκη_1')}
{local:insertDieythinsiYpokatasthma('Οδός1_Υποκαταστήματος', 12,
'Πόλη1_Υποκαταστήματος','Υποκατάστημα_2' )}
{local:insertYpallhlo('AA 2','Υπάλληλος_1',652.00,'6978123654','κινητό','Υποκατάστημα_2')}
{local:insertDieythinsiYpallhλου('Οδός1_Υπαλλήλου', 15, 'Πόλη1_Υπαλήλου', 'AA 2')}
{local:insertParaggeliaYpokatasthmatos('Υποκατάστημα_2','ID1122',10.00)}
{local:insertApothemaKatasthmatos('ID1122',15.00,20.00,'Υποκατάστημα_2' )}
</return>
```

➤ Xquery 'addParaggeliaYpokatasthmatos'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createParaggelia</b>	<ol style="list-style-type: none"> <li>1. \$apothiki as xs:string</li> <li>2. \$ypokatasthma as xs:string</li> <li>3. \$kwdikos as xs:string</li> <li>4. \$posothta as xs:float</li> </ol>	Στοιχείο(element) 'paraggelia'
<b>insertParaggeliaYpokatasthmatos</b>	<ol style="list-style-type: none"> <li>1. \$apothiki as xs:string</li> <li>2. \$ypokatasthma as xs:string</li> <li>3. \$kwdikos as xs:string</li> <li>4. \$posothta as xs:float</li> </ol>	Στοιχείο(element) 'paraggelia' στη θέση: preceding doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma]/paraggelia[1]

Πίνακας 6.7: XQuery 'addParaggeliaYpokatasthmatos', συναρτήσεις

```
declare function local: createParaggelia ($apothiki as xs:string, $ypokatasthma as xs:string,
$kwdikos as xs:string, $posothta as xs:float)
as element (paraggelia) {
let $first := xs:QName('first')
let $max := doc('SuperMarket.xml')
/SuperMarket/apothiki[ @onoma=$apothiki]/ypokatasthma[ @onoma=$ypokatasthma]
/paraggelia[ @kwdikosParaggelias = max(/SuperMarket/apothiki[ @onoma=$apothiki]
/ypokatasthma[ @onoma=$ypokatasthma]/paraggelia/@kwdikosParaggelias)]/@kwdikosParaggelias

return (element paraggelia {
attribute kwdikosParaggelias {($max+1)},
element kwdikos {$kwdikos},
element posothta {$posothta}
})
};
```

```
declare function local:insertParaggeliaYpokatasthmatos
($apothiki as xs:string, $ypokatasthma as xs:string,$kwdikos as xs:string, $posothta as xs:float ){
update insert local:createParaggelia( $apothiki, $ypokatasthma, $kwdikos , $posothta )
preceding doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia[1]
};
```

<return>

```
{local:insertParaggeliaYpokatasthmatos('Αποθήκη_1','Υποκατάστημα_1','ID8877',15.00)}
```

</return>

➤ Xquery 'addParaggeliaApothikis'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createParaggeliaApothikis</b>	<ol style="list-style-type: none"> <li>1. \$kwdikos as xs:string</li> <li>2. \$posothta as xs:float</li> <li>3. \$promhthyths as xs:string</li> <li>4. \$timhAgoras as xs:float</li> <li>5. \$apothiki as xs:string</li> </ol>	Στοιχείο(element) 'paraggeliaApothikis'
<b>insertParaggeliaApothikis</b>	<ol style="list-style-type: none"> <li>1. \$kwdikos as xs:string</li> <li>2. \$posothta as xs:float</li> <li>3. \$promhtheyth as xs:string</li> <li>4. \$timhAgoras as xs:float</li> <li>5. \$apothiki as xs:string</li> </ol>	Στοιχείο(element) 'paraggeliaApothikis' στη θέση: following doc('SuperMarket.xml') //apothiki[@onoma=\$apothiki] /promhtheythsApothikis

Πίνακας 6.8: XQuery 'addParaggeliaApothikis', συναρτήσεις

```
declare function local:createParaggeliaApothikis
( $kwdikos as xs:string, $posothta as xs:float,$promhthyths as xs:string,
$timhAgoras as xs:float,$apothiki as xs:string)
as element (paraggeliaApothikis) {
let $first := xs:QName('first')
let $max := doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$apothiki]/
paraggeliaApothikis[@kwdikosParaggeliasApothikis = max(/SuperMarket/
apothiki[@onoma=$apothiki]/paraggeliaApothikis/@kwdikosParaggeliasApothikis)]
/@kwdikosParaggeliasApothikis
return (element paraggeliaApothikis {
attribute kwdikosParaggeliasApothikis {$max+1},
element kwdikos {$kwdikos},
element posothta {$posothta},
element promhtheyths {$promhthyths},
element timhAgoras {$timhAgoras}
}});
```

```
declare function local:insertParaggeliaApothikis
($kwdikos as xs:string, $posothta as xs:float,$promhtheyth as xs:string,$timhAgoras as xs:float,
$apothiki as xs:string ){
update insert local:createParaggeliaApothikis($kwdikos
,$posothta,$promhtheyth,$timhAgoras,$apothiki )
following doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/promhtheythsApothikis
};local:insertParaggeliaApothikis('ID1122',15.00,'ID3344',15.0,'Αποθήκη_1' )
```

➤ Xquery 'addProion'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
------------------	----------	-----------

March 25, 2015

<b>createProion</b>	<ol style="list-style-type: none"> <li>1. \$skwdikos as xs:string</li> <li>2. \$perigraph as xs:string</li> <li>3. \$seidos as xs:string</li> <li>4. \$paragwgos as xs:string</li> <li>5. \$promitheutisIDREF as xs:string</li> <li>6. \$timh as xs:float</li> <li>7. \$timhPwlshs as xs:float</li> </ol>	Στοιχείο(element) 'proionta'
<b>insertProion</b>	<ol style="list-style-type: none"> <li>1. \$skwdikos as xs:string</li> <li>2. \$perigraph as xs:string</li> <li>3. \$seidos as xs:string</li> <li>4. \$paragwgos as xs:string</li> <li>5. \$promitheutisIDREF as xs:string</li> <li>6. \$timh as xs:float</li> <li>7. \$timhPwlshs as xs:float</li> </ol>	Στοιχείο(element) 'proionta' στη θέση: preceding doc('SuperMarket.xml') /SuperMarket/proionta[1]

Πίνακας 6.9: XQuery 'addProion', συναρτήσεις

```

declare function local:createProion
($skwdikos as xs:string, $perigraph as xs:string, $seidos as xs:string, $paragwgos as xs:string,
$promitheutisIDREF as xs:string, $timh as xs:float,$timhPwlshs as xs:float )
as element (proionta) {
let $first := xs:QName('first')
return (element proionta {
    element proionKwdikos {$skwdikos},
    element perigraph {$perigraph},
    element eidos {$seidos},
    element paragwgos {$paragwgos},
    element timh {$timh},
    element promtheyths {
        element promitheutisIDREF {$promitheutisIDREF},
        element timhPwlshs {$timhPwlshs}
    }
})
});

declare function local:insertProion
($skwdikos as xs:string, $perigraph as xs:string, $seidos as xs:string, $paragwgos as xs:string,
$promitheutisIDREF as xs:string, $timh as xs:float,$timhPwlshs as xs:float) {
update insert local:createProion($skwdikos , $perigraph , $seidos , $paragwgos , $promitheutisIDREF ,
$timh,$timhPwlshs)
preceding doc('SuperMarket.xml')/SuperMarket/proionta[1]
};
<return>
{local:insertProion('ID2016','Περιγραφή1','Είδος1','Παραγωγός1','ID3344',2.00,2.50)}
</return>
    
```

➤ Xquery 'addProionApothikis'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createApothema</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> </ol>	Στοιχείο(element) 'proiontaApothema'
<b>insertApothemaApothikis</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$apothiki as xs:string</li> </ol>	Στοιχείο(element) 'proiontaApothema' στη θέση: preceding doc('SuperMarket.xml')

```
//apothiki[@onoma=$apothiki]
/proiontaApothema[1]
```

Πίνακας 6.10: XQuery 'addProionApothiki', συναρτήσεις

```
1 declare function local:createApothema($proionIDREF as xs:string,
2 $apothema as xs:float, $minApothema as xs:float)
3 as element (proiontaApothema) {
4 let $first := xs:QName('first')
5 return (element proiontaApothema {
6     element proionIDREF {$proionIDREF},
7     element apothema {$apothema},
8     element minApothema {$minApothema}
9 });
10
11 declare function local:insertApothemaApothikis($proionIDREF as xs:string,
12 $apothema as xs:float,$minApothema as xs:float, $apothiki as xs:string){
13 update insert local:createApothema($proionIDREF,$apothema, $minApothema )
14 preceding doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/proiontaApothema[1]
15 };
16
17 <return>
18 {local:insertApothemaApothikis('ID2020',15.00,20.00,'Αποθήκη_1' )}
19 </return>
```

Εικόνα 6.5: XQuery addProionApothikis

➤ Xquery 'addProionYpokatasthmatos'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createApothema</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> </ol>	Στοιχείο(element) 'proiontaApothema'
<b>insertApothemaKatasthmatos</b>	<ol style="list-style-type: none"> <li>1. \$proionIDREF as xs:string</li> <li>2. \$apothema as xs:float</li> <li>3. \$minApothema as xs:float</li> <li>4. \$ypokatasthma as xs:string</li> </ol>	Στοιχείο(element) 'proiontaApothema' στη θέση: preceding doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma] /proiontaApothema[1]

Πίνακας 6.11: XQuery 'addProionYpokatasthmatos', συναρτήσεις

```

1 declare function local:createApothema($proionIDREF as xs:string,
2   $apothema as xs:float, $minApothema as xs:float)
3 as element (proiontaApothema) {
4   let $first := xs:QName('first')
5   return (element proiontaApothema {
6     element proionIDREF {$proionIDREF},
7     element apothema {$apothema},
8     element minApothema {$minApothema}
9   });
10
11 declare function local:insertApothemaKatasthmatos($proionIDREF as xs:string,
12   $apothema as xs:float,$minApothema as xs:float, $ypokatasthma as xs:string){
13 update insert local:createApothema($proionIDREF,$apothema,$minApothema )
14 preceding doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/proiontaApothema[1]
15 };
16
17 <return>
18 {local:insertApothemaKatasthmatos('ID8888',15.00,20.00,'Υποκατάστημα_1' )}
19 </return>

```

Εικόνα 6.6: XQuery ddProionYpokatasthmatos

➤ Xquery ‘addPromhtheyth’

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createPromhtheyths</b>	<ol style="list-style-type: none"> <li>1. \$a.f.m. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$phoneNumber as xs:string</li> <li>4. \$seidosThlefwnou as xs:string</li> </ol>	Στοιχείο(element) ‘promhtheyths’
<b>createDieythinsi</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> </ol>	Στοιχείο(element) ‘dieythinsi’
<b>insertPromhtheyths</b>	<ol style="list-style-type: none"> <li>1. \$a.f.m. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$phoneNumber as xs:string</li> <li>4. \$seidosThlefwnou as xs:string</li> </ol>	Στοιχείο(element) ‘promhtheyths’ στη θέση: preceding doc('SuperMarket.xml') /SuperMarket/promhtheyths[1]
<b>insertDieythinsiPromhtheyth</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$a.f.mPromhtheyth as xs:string</li> </ol>	Στοιχείο(element) ‘dieythinsi’ στη θέση: following doc('SuperMarket.xml') //promhtheyths[afm=\$a.f.mPromhtheyth] /onomatepwnymo

Πίνακας 6.12: XQuery 'addPromhtheyth', συναρτήσεις

```

declare function local:createPromhtheyths($a.f.m. as xs:string,
$onomatepwnymo as xs:string, $phoneNumber as xs:string,
$seidosThlefwnou as xs:string)
as element (promhtheyths) {
let $first := xs:QName('first')
return (element promhtheyths {

```

```

element afm {$a.f.m.},
element onomatepwynomo {$onomatepwynomo},
element thlefwno {
element eidosthlefwnou {$eidosthlefwnou},
element number {$phoneNumber}
}
});
declare function local:createDieythinsi ($odos as xs:string,
$arithmetic as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
element Odos {$odos},
element Arithmos{$arithmetic},
element Poli{$poli}
});
declare function local:insertPromtheyths($a.f.m. as xs:string,$onomatepwynomo
as xs:string, $phoneNumber as xs:string, $eidosthlefwnou as xs:string){
update insert local:createPromtheyths($a.f.m., $onomatepwynomo , $phoneNumber ,
$eidosthlefwnou) preceding doc('SuperMarket.xml')/SuperMarket/promtheyths[1]
};
declare function local:insertDieythinsiPromtheyth ($odos as xs:string,
$arithmetic as xs:integer, $poli as xs:string,$a.f.mPromtheyth as xs:string) {
update insert local:createDieythinsi($odos, $arithmetic,$poli)
following doc('SuperMarket.xml')//promtheyths[afm=$a.f.mPromtheyth]/onomatepwynomo
};
<return>
{local:insertPromtheyths('ID2016','Προμηθευτής1','2810235689','FAX')}
{local:insertDieythinsiPromtheyth('Οδός1_Προμηθευτή',15,'Πόλη1_Προμηθευτή','ID2016')}
</return>

```

➤ Xquery 'addPromtheythApothikis'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createpromtheythsApothikis</b>	1. \$promitheutisIDREF as xs:string 2. \$timhPwlshs as xs:float	Στοιχείο(element) 'promtheythsApothikis'
<b>insertPromtheythsApothikis</b>	1. \$promitheutisIDREF as xs:string 2. \$timhPwlshs as xs:float 3. \$apothiki as xs:string	Στοιχείο(element) 'promtheythsApothikis' στη θέση: preceding doc('SuperMarket.xml') //apothiki[@onoma=\$apothiki] /promtheythsApothikis[1]

Πίνακας 6.13: XQuery 'addPromtheythApothikis', συναρτήσεις

```

1 declare function local:createPromtheythsApothikis($promitheutisIDREF as xs:string, $timhPwlhshs as xs:float)
2 as element (promtheythsApothikis) {
3   let $first := xs:QName('first')
4   return (element promtheythsApothikis {
5     element promitheutisIDREF ($promitheutisIDREF),
6     element timhPwlhshs {$timhPwlhshs}
7   });
8
9 declare function local:insertPromtheythsApothikis($promitheutisIDREF as xs:string,
10 $timhPwlhshs as xs:float,$apothiki as xs:string){
11 update insert local:createPromtheythsApothikis($promitheutisIDREF , $timhPwlhshs )
12 preceding doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/promtheythsApothikis[1]
13 };
14
15 <return>
16 {local:insertPromtheythsApothikis('ID2014',15.00,'Αποθήκη_1' )}
17 </return>

```

Εικόνα 6.7: XQuery 'addPromtheythApothikis'

➤ Xquery 'addYpallhlos'

Όνομα συνάρτησης	Ορίσματα	Επιστροφή
<b>createYpallhlos</b>	<ol style="list-style-type: none"> <li>1. \$a.t. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$misthos as xs:float</li> <li>4. \$phoneNumber as xs:string</li> <li>5. \$eidosthlefwonou as xs:string</li> </ol>	Στοιχείο(element) 'ypallhlos'
<b>createDieythinsi</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> </ol>	Στοιχείο(element) 'deythinsi'
<b>insertYpallhlo</b>	<ol style="list-style-type: none"> <li>1. \$a.t. as xs:string</li> <li>2. \$onomatepwnymo as xs:string</li> <li>3. \$misthos as xs:float</li> <li>4. \$phoneNumber as xs:string</li> <li>5. \$eidosthlefwonou as xs:string</li> <li>6. \$ypokatasthma as xs:string</li> </ol>	Στοιχείο(element) 'ypallhlos' στη θέση: preceding doc('SuperMarket.xml') //ypokatasthma[@onoma=\$ypokatasthma] /ypallhlos[1]
<b>insertDieythinsiYpallhlo</b>	<ol style="list-style-type: none"> <li>1. \$odos as xs:string</li> <li>2. \$arithmos as xs:integer</li> <li>3. \$poli as xs:string</li> <li>4. \$a.t.Ypallhlo as xs:string</li> </ol>	Στοιχείο(element) 'dieythinsi' στη θέση: following doc('SuperMarket.xml') //ypallhlos[@at=\$a.t.Ypallhlo]/misthos

Πίνακας 6.14: XQuery addYpallhlos, συναρτήσεις

```

declare function local:createYpallhlos ($a.t. as xs:string, $onomatepwynomo as xs:string, $misthos
as xs:float,$phoneNumber as xs:string, $eidosthlefwnou as xs:string)
as element (ypallhlos) {
let $first := xs:QName('first')
return (element ypallhlos {
attribute at {$a.t.},
element onomatepwynomo {$onomatepwynomo},
element misthos {$misthos},
element thlefwno {
element eidosthlefwnou {$eidosthlefwnou},
element number {$phoneNumber}
}
});

declare function local:createDieythinsi ($odos as xs:string, $arithmos as xs:integer, $poli as xs:string)
as element (dieythinsi) {
let $first := xs:QName('first')
return (element dieythinsi {
element Odos {$odos},
element Arithmos{$arithmos},
element Poli{$poli}
});

declare function local:insertYpallhlo($a.t. as xs:string, $onomatepwynomo as xs:string,
$misthos as xs:float,$phoneNumber as xs:string, $eidosthlefwnou as xs:string, $ypokatasthma as
xs:string){
update insert
local:createYpallhlos($a.t.,$onomatepwynomo,$misthos,$phoneNumber,$eidosthlefwnou)
preceding doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/ypallhlos[1]
};

declare function local:insertDieythinsiYpallhlou ($odos as xs:string, $arithmos as xs:integer,
$poli as xs:string,$a.t.Ypallhlou as xs:string) {
update insert local:createDieythinsi($odos, $arithmos,$poli)
following doc('SuperMarket.xml')//ypallhlos[@at=$a.t.Ypallhlou]/misthos
};
<return>
{local:insertYpallhlo('AA 1','Υπάλληλος_2',450.00,'6978125654','κινητό','Υποκατάστημα_1')}
{local:insertDieythinsiYpallhlou('Οδός2_Υπαλλήλου', 85, 'Πόλη2_Υπαλλήλου', 'AA 1')}
</return>

```

## 6.2 XQuery διαγραφής στοιχείου (element)

Η επόμενη διαδικασία που καλούμαστε να υλοποιήσουμε μέσω των XQuery είναι η διαγραφή όλων των στοιχείων(elements) που συμμετέχουν στο XML Schema. Η διαδικασία αυτή φυσικά προϋποθέτει την ύπαρξη των στοιχείων που πρόκειται να διαγράψουμε. Ας ξεκινήσουμε με τη διαγραφή συγκεκριμένης αποθήκης:



```

1 declare function local:deleteApothiki($onoma as xs:string)
2 {
3   for $apothiki in doc("SuperMarket.xml")//apothiki[@onoma=$onoma]
4   return
5     update delete $apothiki
6 };
7
8 local:deleteApothiki('Αποθήκη_1')

```

Εικόνα 6.8: XQuery 'deleteApothiki'

Στην γραμμή 1 συναντάμε τον ορισμό της συνάρτησης με όνομα deleteApothiki καθώς και το όρισμα που χρειάζεται να δώσουμε στη συνάρτηση αυτή κατά την κλήση της. Στην περίπτωση μας είναι το 'onoma' τύπου string και γενικά ως όρισμα στις συναρτήσεις διαγραφής θα χρησιμοποιούμε το χαρακτηριστικό αυτό που αποτελεί και το πρωτεύον κλειδί του στοιχείου(element) που θέλουμε να διαγράψουμε. Στην γραμμή 3 δίνουμε στην ουσία μία εντολή η οποία ψάχνει μέσα στο έγγραφο 'SuperMarket.xml' την μεταβλητή 'apothiki' της οποίας το γνώρισμα(attribute) 'onoma' ισούται με την μεταβλητή 'onoma' που δώσαμε για όρισμα. Στην γραμμή 5 μας επιστρέφει ανανεωμένο το έγγραφο XML μετά την διαγραφή που του ζητήσαμε να κάνει. Τέλος, στην γραμμή 8 βλέπουμε ένα παράδειγμα κλήσης αυτής της συνάρτησης. Ακριβώς αντίστοιχα δημιουργήσαμε XQuery διαγραφής και για τα υπόλοιπα στοιχεία. Ακολουθεί ένας συνοπτικός πίνακας για τις συναρτήσεις που χρησιμοποιήθηκαν κι έπειτα αναλυτικά ο κώδικας των XQuery.

Όνομα συνάρτησης	Ορίσματα	Παράδειγμα κλήσης
<b>deleteParaggeliasApothikis</b>	1. \$skwdikosParaggeliasApothikis as xs:int 2. \$apothiki as xs:string	local:deleteParaggeliasApothikis (1,'Αποθήκη_1')
<b>deleteParaggeliasYpokatasthmatos</b>	1. \$skwdikosParaggelias as xs:int 2. \$ypokatasthma as xs:string	local:deleteParaggeliasYpokatasthmatos (1,'Υποκατάστημα_1')
<b>deleteProion</b>	1. \$proionKwdikos as xs:string	local:deleteProion('ID1122')
<b>deleteProionApothikis</b>	1. \$proionIDREF as xs:string 2. \$apothiki as xs:string	local:deleteProionApothikis ('ID1122','Αποθήκη_1')
<b>deleteProionYpokatasthmatos</b>	1. \$proionIDREF as xs:string 2. \$ypokatasthma as xs:string	local:deleteProionYpokatasthmatos ('ID1122','Υποκατάστημα_1')
<b>deletePromhtheyth</b>	1. \$safm as xs:string	local:deletePromhtheyth ('ID3344')
<b>deletePromhtheythApothikis</b>	1. \$promitheutisIDREF as xs:string 2. \$apothiki as xs:string	local:deletePromhtheythApothikis ('ID3344','Αποθήκη_1')
<b>deleteYpallhloYpokatasthmatos</b>	1. \$sat as xs:string	local:deleteYpallhloYpokatasthmatos ('AA 11111')
<b>deleteYpokatasthmaApothikis</b>	1. \$onoma as xs:string	local:deleteYpokatasthmaApothikis ('Υποκατάστημα_1')

Πίνακας 6.15: XQuery διαγραφών, συναρτήσεις

```
declare function local:deleteParaggeliasApothikis
($kwdikosParaggeliasApothikis as xs:int,$apothiki as xs:string)
{
for $kwdikosParaggeliasApothikis in doc
("SuperMarket.xml")/SuperMarket/apothiki[@onoma=$apothiki]
/paraggeliaApothikis[@kwdikosParaggeliasApothikis=$kwdikosParaggeliasApothikis]
return
  update delete $kwdikosParaggeliasApothikis
};

local:deleteParaggeliasApothikis(1, 'Αποθήκη_1')
```

Εικόνα 6.9: XQuery 'deleteParaggeliaApothikis'

```
declare function local:deleteParaggeliasYpokatasthmatos
($kwdikosParaggelias as xs:int,$ypokatasthma as xs:string)
{
for $kwdikosParaggelias in doc
("SuperMarket.xml")/SuperMarket/apothiki/ypokatasthma[@onoma=$ypokatasthma]
/paraggelia[@kwdikosParaggelias=$kwdikosParaggelias]
return
  update delete $kwdikosParaggelias
};

local:deleteParaggeliasYpokatasthmatos(1, 'Υποκατάστημα_1')
```

Εικόνα 6.10: XQuery 'deleteParaggeliasYpokatasthmatos'

```
declare function local:deleteProion($proionKwdikos as xs:string)
{
for $proionta in doc
("SuperMarket.xml")//proionta[proionKwdikos=$proionKwdikos]
return
  update delete $proionta
};

local:deleteProion('ID1122')
```

Εικόνα 6.11: XQuery 'deleteProion'

```
declare function local:deleteProionApothikis
($proionIDREF as xs:string,$apothiki as xs:string)
{
for $proionIDREF in doc
("SuperMarket.xml")/SuperMarket/apothiki[@onoma=$apothiki]
/proiontaApothema[proionIDREF=$proionIDREF]
return
    update delete $proionIDREF
};

local:deleteProionApothikis('ID1122','Αποθήκη_1')
```

Εικόνα 6.12: XQuery 'deleteProionApothikis'

```
declare function local:deleteProionYpokatasthmatos
($proionIDREF as xs:string,$ypokatasthma as xs:string)
{
for $proionIDREF in
doc("SuperMarket.xml")/SuperMarket/apothiki/ypokatasthma[@onoma=$ypokatasthma]
/proiontaApothema[proionIDREF=$proionIDREF]
return
    update delete $proionIDREF
};

local:deleteProionYpokatasthmatos('ID1122','Υποκατάστημα_1')
```

Εικόνα 6.13: XQuery 'deleteProionYpokatasthmatos'

```
declare function local:deletePromhtheyth($afm as xs:string)
{
for $promhtheyths in doc("SuperMarket.xml")//promhtheyths[afm=$afm]
return
    update delete $promhtheyths
};

local:deletePromhtheyth('ID3344')
```

Εικόνα 6.14: XQuery 'deletePromhtheyth'

```

declare function local:deletePromhtheythApothikis(
  $promitheutisIDREF as xs:string, $apothiki as xs:string)
{
  for $promhtheythsApothikis in
  doc("SuperMarket.xml")/SuperMarket/apothiki[@onoma=$apothiki]
  /promhtheythsApothikis[promitheutisIDREF=$promitheutisIDREF]
  return
    update delete $promhtheythsApothikis
};

local:deletePromhtheythApothikis('ID3344', 'Αποθήκη_1')

```

Εικόνα 6.15: XQuery 'deletePromhtheythApothikis'

```

declare function local:deleteYpallhloYpokatasthmatos($at as xs:string)
{
  for $at in
  doc("SuperMarket.xml")/SuperMarket/apothiki/ypokatasthma/ypallhlos[@at=$at]
  return
    update delete $at
};

local:deleteYpallhloYpokatasthmatos('AA 111111')

```

Εικόνα 6.16: XQuery 'deleteYpallhloYpokatasthmatos'

```

declare function local:deleteYpokatasthmaApothikis($onoma as xs:string)
{
  for $onoma
  in doc("SuperMarket.xml")/SuperMarket/apothiki/ypokatasthma[@onoma=$onoma]
  return
    update delete $onoma
};

local:deleteYpokatasthmaApothikis('Υποκατάστημα_1')

```

Εικόνα 6.17: XQuery 'deleteYpokatasthmaApothikis'

### 6.3 XQuery ανανέωσης στοιχείου (element)

Η τρίτη διαδικασία που πρέπει να υποστηρίζεται μέσω χρήσης XQuery είναι η ανανέωση του αποθέματος των προϊόντων τόσο της αποθήκης όσο και του υποκαταστήματος. Δημιουργήσαμε για το σκοπό αυτό λοιπόν τα δύο XQuery 'updateApothemaApothikis' και 'updateApothemaYpokatasthmatos'.

```
1 declare function local:updateApothemaApothikis($apothiki as xs:string,  
2 $proion as xs:string,$newApothema as xs:float){  
3 update value //apothiki[@onoma=$apothiki]/proiontaApothema[proionIDREF=$proion]  
4 /apothema with $newApothema  
5 };  
6  
7 <return>  
8 {local:updateApothemaApothikis('Αποθήκη_1','ID1122',150.00)}  
9 </return>
```

Εικόνα 6.18: XQuery 'updateApothemaApothkis'

Στην πρώτη γραμμή συναντάμε τον ορισμό της συνάρτησης που θα χρησιμοποιήσουμε προκειμένου να πετύχουμε το σκοπό μας. Αμέσως μετά είναι τα ορίσματα που χρειάζεται αυτή η συνάρτηση για να δουλέψει, τα οποία δεν μπορεί να είναι άλλα από:

- τον κωδικό του προϊόντος,
- τη νέα τιμή του αποθέματος και
- το όνομα της αποθήκης που θα γίνει η ανανέωση του αποθέματος

Στις γραμμές 3-4 δίνουμε τη διαδρομή που βρίσκεται η τιμή(value) που θέλουμε να ανανεωθεί(update) κι αμέσως μετά, δίνουμε την τιμή με την οποία θέλουμε να γίνει η αντικατάσταση και στη γραμμή 8 βλέπουμε ένα παράδειγμα κλήσης της. Αντίστοιχα είναι και το XQuery για την ανανέωση του αποθέματος σε ένα υποκατάστημα, με τη μόνη διαφορά να σημειώνεται στα ορίσματα της συνάρτησης, αφού αντί για το όνομα της αποθήκης, θέλουμε το όνομα του υποκαταστήματος που θα γίνει ανανέωση.

```
1 declare function local:updateApothemaYpokatasthmatos($ypokatasthma as xs:string,  
2 $proion as xs:string,$newApothema as xs:float){  
3 update value //ypokatasthma[@onoma=$ypokatasthma]/proiontaApothema[proionIDREF=$proion]  
4 /apothema with $newApothema  
5 };  
6  
7 <return>  
8 {local:updateApothemaYpokatasthmatos('Υποκατάστημα_1','ID1122',50.00)}  
9 </return>
```

Εικόνα 6.19: XQuery 'updateApothemaYpokatasthmatos'

## 7 Αυτόματες παραγγελίες μέσω XQuery

Η τελευταία διαδικασία που μας ζητήθηκε να ολοκληρώσουμε μέσω των XQuery είναι αυτή της αυτόματης παραγγελίας, τόσο των υποκαταστημάτων όσο και των αποθηκών. Σκοπός αυτής της διαδικασίας είναι μετά την πώληση οποιασδήποτε ποσότητας προϊόντων, να μη μένει ούτε η αποθήκη ούτε το υποκατάστημα με απόθεμα, μικρότερο από το ελάχιστο που τους έχουμε ορίσει. Μία σειρά από κλήσεις συναρτήσεων θα μας βοηθήσει σε αυτό, μέσα από τις οποίες θα πρέπει να γίνεται έλεγχος ανάλογα την ποσότητα που πούλησε το υποκατάστημα, για να βλέπουμε αν χρειάζεται ή όχι παραγγελία. Δηλαδή, αν το απόθεμα του προϊόντος στο υποκατάστημα έπεσε κάτω από το ελάχιστο απόθεμα που έχει το συγκεκριμένο υποκατάστημα για το συγκεκριμένο προϊόν.

- Αν δεν ισχύει αυτό, τότε απλά θέλουμε να γίνεται μία ανανέωση στο τρέχον απόθεμα του συγκεκριμένου υποκαταστήματος για το συγκεκριμένο προϊόν αφαιρώντας την ποσότητα που πουλήθηκε.
- Αν ισχύει θα πρέπει να δημιουργείται παραγγελία στο συγκεκριμένο υποκατάστημα για το συγκεκριμένο προϊόν και με ποσότητα τόση ώστε το τελικό απόθεμα του υποκαταστήματος για το προϊόν να ισούται με το διπλάσιο του ελαχίστου του. Η ποσότητα όμως που χρειάζεται το υποκατάστημα θα αφαιρεθεί από την αποθήκη που συνεργάζεται. Επομένως, πρέπει να γίνει επίσης ένας έλεγχος για το αν το απόθεμα της αποθήκης στο συγκεκριμένο προϊόν, μετά τον εφοδιασμό του υποκαταστήματος, έπεσε κάτω από το ελάχιστο.
  - Αν δεν ισχύει αυτό, τότε απλά κάνουμε μία ανανέωση στο τρέχον απόθεμα του προϊόντος της αποθήκης αφαιρώντας την ποσότητα που έστειλε στο υποκατάστημα.
  - Αν όμως ισχύει, τότε θα πρέπει να γίνει μία παραγγελία στην αποθήκη για το συγκεκριμένο προϊόν και με ποσότητα τόση ώστε το τελικό απόθεμα που θα έχει η αποθήκη στον συγκεκριμένο κωδικό προϊόντος, να είναι διπλάσια του ελαχίστου της. Τέλος, ο προμηθευτής από τον οποίο θα παραγγείλει η αποθήκη το προϊόν θα είναι αυτός που της το προσφέρει με τη χαμηλότερη τιμή.

Ας δούμε τώρα αναλυτικότερα κάθε μία από τις συναρτήσεις που χρησιμοποιήσαμε και τη σειρά με την οποία έγινε η κλήση τους. Το XQuery ξεκινάει με την κλήση της συνάρτησης 'checkApothemaYpokatasthmatos':

```
local:checkApothemaYpokatasthmatos('Αποθήκη_1','Υποκατάστημα_1', 'ID1122', 1)
```

```

1 declare function local:checkApothemaYpokatasthmatos
2 ($apothiki as xs:string, $ypokatasthma as xs:string, $proionID as xs:string,
3 $posothta as xs:float)
4 {
5     let $apothema := doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]
6     /proiontaApothema[proionIDREF=$proionID]/apothema/text()
7     let $minApothema := doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]
8     /proiontaApothema[proionIDREF=$proionID]/minApothema/text()
9     let $ypoloipo := ($apothema)-($posothta)
10    let $apofasi := local:between-inclusive($ypoloipo,0,$minApothema)
11
12    return local:makeUpdatesYpokatasthma
13    ($minApothema, $ypoloipo, $apothiki, $ypokatasthma,$proionID, $apofasi)
14 };

```

Εικόνα 7.1: Συνάρτηση 'checkApothemaYpokatasthmatos'

Στις γραμμές 2-3 συναντάμε τα ορίσματα της συνάρτησης που τα χρειάζεται προκειμένου να κάνει τον έλεγχο για το απόθεμα που μένει στο μαγαζί μετά την πώληση μίας ποσότητας από κάποιο προϊόν. Στις γραμμές 5-6 παίρνουμε από το έγγραφο XML το απόθεμα που είχε το υποκατάστημα πριν την πώληση και αποδίδουμε την τιμή σε μία μεταβλητή με όνομα 'apothema'. Με τον ίδιο τρόπο αντλούμε πάλι από το XML έγγραφο την τιμή για το ελάχιστο απόθεμα που αντιστοιχεί στο συγκεκριμένο μαγαζί για το συγκεκριμένο προϊόν και την αποθηκεύουμε στη μεταβλητή 'minApothema', γραμμές 7-8. Αφαιρώντας από το 'apothema' την 'posothta' που του έχουμε δώσει

σαν όρισμα και αντιστοιχεί στην ποσότητα πώλησης, καταλήγουμε στο απόθεμα που έμεινε στο υποκατάστημα μετά την πώληση, γραμμή 9. Αυτό θα μας βοηθήσει να πάρουμε μία απόφαση για το αν χρειάζεται το υποκατάστημα παραγγελία και το αποθηκεύουμε στην μεταβλητή 'υπολοιπο'.

Στο σημείο αυτό θα καλέσουμε τη συνάρτηση 'between-inclusive' δίνοντάς της για ορίσματα το 'υπολοιπο', τη τιμή μηδέν και το 'minApothema'. Η συνάρτηση αυτή συγκρίνει το πρώτο όρισμα, 'υπολοιπο' με τα άλλα δύο και αν η τιμή του είναι μέσα στο εύρος τιμών που ορίζουν τα άλλα δύο ορίσματα, τότε επιστρέφει 'True', διαφορετικά επιστρέφει 'False'. Τις τιμές στα άκρα, δηλαδή αν είναι π.χ. μικρότερο, ίσο ή μεγαλύτερο, μπορούμε να τα ρυθμίσουμε εμείς στην γραμμή 19 του κώδικα που ακολουθεί.

```

16 declare function local:between-inclusive
17 ($value as xs:anyAtomicType, $minValue as xs:anyAtomicType,
18 $maxValue as xs:anyAtomicType) as xs:boolean {
19 $value >= $minValue and $value < $maxValue
20 };

```

Εικόνα 7.2: Συνάρτηση 'between-inclusive'

Η Boolean τιμή που επιστρέφει λοιπόν η συνάρτηση 'between-inclusive' αποθηκεύεται στην μεταβλητή 'apofasi', γραμμή 10 εικόνα 7.1 κι έπειτα στην τιμή επιστροφής βρίσκουμε την κλήση της επόμενης συνάρτησης που είναι η 'makeUpdatesYpokatasthma', γραμμές 12-13 εικόνα 7.1.

```

23 declare function local:makeUpdatesYpokatasthma
24 ($minApothema as xs:float, $ypoloipo as xs:float, $apothiki as xs:string,
25 $ypokatasthma as xs:string, $proionID as xs:string, $apofasi as xs:boolean )
26 {
27 if(not ($apofasi))
28 then
29     local:updateApothemaYpokatasthmatos($ypokatasthma,$proionID, $ypoloipo)
30 else
31 <return> {
32     local:insertParaggeliaYpokatasthmatos
33     ($apothiki, $ypokatasthma,$proionID, ((2*$minApothema)-$ypoloipo)),
34     local:updateApothemaYpokatasthmatos
35     ($ypokatasthma, $proionID, 2*$minApothema ),
36     local:updateApothemaApothikis($apothiki , $proionID, (2*$minApothema)-$ypoloipo),
37     local:checkApothemaApothikis($apothiki , $proionID, (2*$minApothema)-$ypoloipo)
38     }
39 </return>
40 };

```

Εικόνα 7.3: Συνάρτηση 'makeUpdatesYpokatasthma'

Αφού έχουμε μία μεταβλητή πλέον που δηλώνει με την τιμή της, σε ποιο εύρος βρίσκεται το απόθεμα του προϊόντος στο υποκατάστημα μετά την πώληση, περνάμε στην συνάρτηση 'makeUpdatesYpokatasthma', εικόνα 7.3. Όπως είπαμε παραπάνω, υπάρχουν δύο περιπτώσεις ανάλογα με το νέο απόθεμα του μαγαζιού, ή να χρειάζεται παραγγελία ή όχι. Αυτές τις δύο περιπτώσεις έρχεται να διαχωρίσει η συνάρτηση αυτή, χρησιμοποιώντας την μεταβλητή 'apofashi' και καλώντας σε κάθε περίπτωση τις ανάλογες συναρτήσεις.

Στις γραμμές 24-25 βρίσκουμε τα ορίσματα που πρέπει να της περάσουμε κατά την κλήση της και στη γραμμή 27 ξεκινάει ένας βρόγχος ελέγχου(if-then-else). Αν η 'apofashi' ισούται με 'False', γραμμή 27, γεγονός που σημαίνει ότι το απόθεμα του μαγαζιού δεν έπεσε κάτω από το ελάχιστο στο συγκεκριμένο προϊόν, τότε θέλουμε απλά να ενημερωθεί το απόθεμα με τη νέα τιμή. Για το σκοπό αυτό καλούμε στην γραμμή 29 τη συνάρτηση 'updateApothemaYpokatasthmatos':

```

42 declare function local:updateApothemaYpokatasthmatos
43 ($ypokatasthma as xs:string,$proion as xs:string,$newApothema as xs:float){
44 update value //ypokatasthma[@onoma=$ypokatasthma]
45 /proiontaApothema[proionIDREF=$proion]/apothema with $newApothema
46 };

```

Εικόνα 7.4: Συνάρτηση 'updateApothemaYpokatasthmatos'

Στη γραμμή 43 έχουμε τα ορίσματα της συνάρτησης 'updateApothemaYpokatasthmatos' ενώ στη γραμμή 44-45 δίνουμε την εντολή ανανέωσης της ήδη υπάρχουσας τιμής στο συγκεκριμένο απόθεμα, με τη νέα τιμή του αποθέματος.

Επιστρέφοντας στη συνάρτηση 'makeUpdatesYpokatasthma', εικόνα 7.3, συνεχίζουμε με τις ενέργειες που θέλουμε να γίνουν αν η μεταβλητή 'apofash' ισούται με 'True', δηλαδή αν το υποκατάστημα πρέπει να τροφοδοτηθεί με προϊόντα που πούλησε. Το πρώτο που θέλουμε να υλοποιηθεί, είναι η δημιουργία παραγγελίας για το υποκατάστημα που να περιέχει το προϊόν του οποίου το απόθεμα έπεσε κάτω από το ελάχιστο και τόση ποσότητα ώστε τελικά το υποκατάστημα να έχει το διπλάσιο του ελαχίστου αποθέματος. Έτσι στις γραμμές 32-33 συναντάμε κλήση της συνάρτησης 'insertParaggeliaYpokatasthmatos'.

```

48 declare function local:insertParaggeliaYpokatasthmatos
49 ( $apothiki as xs:string, $ypokatasthma as xs:string,
50 $kwdikos as xs:string, $posothta as xs:float ){
51 update insert local:createParaggelia
52 ( $apothiki, $ypokatasthma, $kwdikos , $posothta )
53 preceding doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia[1]
54 };

```

Εικόνα 7.5: Συνάρτηση 'insertParaggeliaYpokatasthmatos'

Η συνάρτηση αυτή εισάγει στο υποκατάστημα, γραμμή 53, που του δίνουμε την κατάλληλη παραγγελία καλώντας την συνάρτηση 'createParaggelia', στις γραμμές 51-52. Ο κώδικας για την 'createParaggelia' είναι αντίστοιχος με αυτόν που περιγράψαμε στη παράγραφο 6.1 για το XQuery 'addParaggelia'.

```

declare function local:createParaggelia($apothiki as xs:string, $ypokatasthma as xs:string,
$kwdikos as xs:string, $posothta as xs:float)
as element (paraggelia) {
let $first := xs:QName('first')
let $max := doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$apothiki]
/ypokatasthma[@onoma=$ypokatasthma]/paraggelia[@kwdikosParaggelias = max
(/SuperMarket/apothiki[@onoma=$apothiki]/ypokatasthma[@onoma=$ypokatasthma]
/paraggelia/@kwdikosParaggelias)]/@kwdikosParaggelias
return (element paraggelia {
attribute kwdikosParaggelias {($max+1)},
element kwdikos {$kwdikos},
element posothta {$posothta}
}) };

```

Εικόνα 7.6: Συνάρτηση 'createParaggelia'

Συνεχίζουμε δίνοντας εντολή για ανανέωση της τιμής του αποθέματος για το προϊόν που παράγγειλε το υποκατάστημα, γραμμή 34-35 εικόνα 7.3. Αυτό γίνεται μέσω της κλήσης της



συνάρτησης 'updateApothemaYpokatasthmatos', που αναλύσαμε και νωρίτερα στην εικόνα 7.4. Η διαφορά με πριν είναι πως τώρα δίνουμε διαφορετικό όρισμα για το νέο απόθεμα του προϊόντος, αφού μετά την παραγγελία θα ισούται με το διπλάσιο του ελαχίστου του. Εφόσον το υποκατάστημα τροφοδοτήθηκε από την αποθήκη πρέπει τώρα να μειώσουμε και το απόθεμα στο συγκεκριμένο προϊόν της αποθήκης αφαιρώντας από το ήδη υπάρχον, την ποσότητα που έστειλε στο υποκατάστημα. Καλούμε λοιπόν τη συνάρτηση 'updateApothemaApothikis', γραμμή 36 στην εικόνα 7.3, που είναι η εξής:

```

117 declare function local:updateApothemaApothikis
118 ($apothiki as xs:string,$proion as xs:string,$newApothema as xs:float){
119 let $apothemaApothikis := doc('SuperMarket.xml')
120 //apothiki[@onoma=$apothiki]/proiontaApothema[proionIDREF=$proion]/apothema
121 let $ypoloipoApothikis := ($apothemaApothikis)-($newApothema)
122
123 return update value //apothiki[@onoma=$apothiki]/
124 proiontaApothema[proionIDREF=$proion]/apothema with $ypoloipoApothikis
125 };

```

Εικόνα 7.7: Συνάρτηση 'updateApothemaApothikis'

Παίρνει το τρέχον απόθεμα της αποθήκης, γραμμές 119-120 και αποθηκεύει τη τιμή του σε μία μεταβλητή 'apothemaApothikis' κι έπειτα υπολογίζει την υπόλοιπη ποσότητα που μένει στην αποθήκη μετά τον εφοδιασμό του μαγαζιού, γραμμή 121. Μετά από αυτό, ανανεώνει την τιμή του αποθέματος της αποθήκης για το συγκεκριμένο προϊόν, γραμμές 123-124.

Επιστρέφοντας στη συνάρτηση 'makeUpdatesYpokatasthma', εικόνα 7.3, η τελευταία κλήση συνάρτησης που κάνει είναι αυτή για τη συνάρτηση 'checkApothemaApothikis' στη γραμμή 37.

```

71 declare function local:checkApothemaApothikis
72 ($apothiki as xs:string, $proionID as xs:string,$posothtaApostolhs as xs:float)
73 {
74     let $apothemaApothikisNew := doc('SuperMarket.xml')
75     //apothiki[@onoma=$apothiki]/proiontaApothema[proionIDREF=$proionID]
76     /apothema/text()
77     let $minApothemaApothikis := doc('SuperMarket.xml')
78     //apothiki[@onoma=$apothiki]/proiontaApothema[proionIDREF=$proionID]
79     /minApothema/text()
80     let $apofasiApothikis := local:between-inclusive
81     ($apothemaApothikisNew,0,$minApothemaApothikis)
82
83     return local:makeUpdatesApothiki
84     ($minApothemaApothikis, $apothemaApothikisNew, $apothiki,$proionID, $apofasiApothikis)
85 };

```

Εικόνα 7.8: Συνάρτηση 'checkApothemaApothikis'

Η συνάρτηση αυτή είναι υπεύθυνη για να μαζέψουμε τις πληροφορίες που χρειάζεται ώστε καλώντας την 'between-inclusive' να πάρουμε μία απάντηση που στη συνέχεια θα κρίνει αν χρειάζεται και η αποθήκη παραγγελία μετά τον εφοδιασμό του υποκαταστήματος. Στις γραμμές 74-76 έχουμε την απόδοση τιμής του αποθέματος της αποθήκης, μετά τον εφοδιασμό, στην μεταβλητή 'apothemaApothikisNew'. Στις επόμενες γραμμές 77-79 με αντίστοιχο τρόπο αποδίδουμε το ελάχιστο απόθεμα της αποθήκης στη μεταβλητή 'minApothemaApothikis'. Έπειτα, στις γραμμές 80-81, κάνουμε κλήση της συνάρτησης 'between-inclusive', στην οποία αναφερθήκαμε νωρίτερα, δίνοντας τα κατάλληλα ορίσματα κι αποθηκεύοντας την τιμή επιστροφής της στη μεταβλητή

‘apofasiApothikis’. Συνεχίζουμε με τις γραμμές 83-84 όπου βρίσκεται η κλήση της επόμενης συνάρτησης, ‘makeUpdatesApothiki’, η οποία είναι υπεύθυνη να καλέσει τις σωστές συναρτήσεις ανάλογα την τιμή της μεταβλητής ‘apofasiApothikis’.

```

87 declare function local:makeUpdatesApothiki($minApothemaApothikis as xs:float,
88 $apothemaApothikisNew as xs:float, $apothiki as xs:string, $proionID as xs:string,
89 $apofasiApothikis as xs:boolean )
90 {
91 if( not($apofasiApothikis))
92     then
93         local:updateApothemaApothikis
94         ($apothiki,$proionID, $apothemaApothikisNew)
95     else
96         <return> {
97             local:prosfora
98             ($proionID, ((2*$minApothemaApothikis)-$apothemaApothikisNew), $apothiki),
99             local:fullApothiki ($apothiki, $proionID, 2*$minApothemaApothikis)
100         }
101     </return>
102 };

```

Εικόνα 7.9: Συνάρτηση 'makeUpdatesApothiki'

Στην περίπτωση που ο εφοδιασμός του υποκαταστήματος δεν κάνει αναγκαίο και τον εφοδιασμό της αποθήκης, δηλαδή η μεταβλητή ‘apofasiApothikis’ ισούται με ‘False’, τότε χρειαζόμαστε μονάχα την ανανέωση της τιμής αποθέματος της αποθήκης. Στις γραμμές 93-94, εικόνα 7.9, γίνεται η κλήση της συνάρτησης ‘updateApothemaApothikis’ τον κώδικα του οποίου αναλύσαμε παραπάνω, εικόνα 7.7. Αν όμως προκύψει ανάγκη για εφοδιασμό και της αποθήκης, τότε έρχεται η κλήση της συνάρτησης ‘prosfora’ στις γραμμές 97-98 και της συνάρτησης ‘fullApothiki’ στη γραμμή 99.

```

108 declare function local:prosfora
109 ($proionID as xs:string,$posothta as xs:float,$apothiki as xs:string){
110 let $timhAgoras :=doc('SuperMarket.xml')/SuperMarket
111 /proionta[proionKwdikos=$proionID]/promhtheyths[timhPwlhshs=min(
112 /SuperMarket/proionta[proionKwdikos=$proionID]
113 /promhtheyths/timhPwlhshs)]/timhPwlhshs
114
115 let $promhtheyths := doc('SuperMarket.xml')/SuperMarket
116 /proionta[proionKwdikos='ID1122']/
117 promhtheyths[timhPwlhshs=$timhAgoras]/promitheutisIDREF
118
119 return local:insertParaggeliaApothikis
120 ($proionID, $posothta, $promhtheyths, $timhAgoras, $apothiki)
121
122 };

```

Εικόνα 7.10: Συνάρτηση 'prosfora'

Ξεκινώντας από τη συνάρτηση ‘prosfora’, αφού αυτή καλείται πρώτη, να πούμε ότι σκοπό έχει να βρει τον προμηθευτή που δίνει το προϊόν, το οποίο πρόκειται να παραγγείλει η αποθήκη, με την μικρότερη τιμή. Έτσι, στις γραμμές 110-113 αντλεί την πληροφορία για τη μικρότερη τιμή αγοράς με την οποία προμηθεύεται το συγκεκριμένο προϊόν και την τιμή αυτή την αποθηκεύει στη παράμετρο ‘timhAgoras’. Συνεχίζοντας αποθηκεύει στην παράμετρο ‘promhtheyths’ τον προμηθευτή που μας προσφέρει τη μικρότερη τιμή, γραμμές

115-117. Αφού έχουμε βρεί και τον προμηθευτή λοιπόν από τον οποίο θα γίνει η παραγγελία γίνεται η κλήση της συνάρτησης 'insertParaggeliaApothikis' στις γραμμές 119-120, η οποία με τη σειρά της καλεί την 'createPraggeliaApothikis' στις σειρές 154-156. Ο συνδιασμός των δύο τελευταίων συναρτήσεων έχει αναλυθεί και στην ενότητα 6.1 για το XQuery 'addParaggeliaApothikis'

```
declare function local:insertParaggeliaApothikis
($kwdikos as xs:string, $posothta as xs:float, $promhtheyth as xs:string,
$timhAgoras as xs:float, $apothiki as xs:string ) {
update insert local:createParaggeliaApothikis
($kwdikos , $posothta, $promhtheyth, $timhAgoras, $apothiki )
following doc('SuperMarket.xml')//apothiki[@onoma=$apothiki]/promhtheythsApothikis
};
```

Εικόνα 7.11: Συνάρτηση 'insertParaggeliaApothikis'

```
declare function local:createParaggeliaApothikis
($kwdikos as xs:string, $posothta as xs:float, $promhthyths as xs:string,
$timhAgoras as xs:float, $apothiki as xs:string)
as element (paraggeliaApothikis) {
let $first := xs:QName('first')
let $max := doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$apothiki]
/paraggeliaApothikis[@kwdikosParaggeliasApothikis = max(/SuperMarket
/apothiki[@onoma=$apothiki]/paraggeliaApothikis
/@kwdikosParaggeliasApothikis)]/@kwdikosParaggeliasApothikis

return (element paraggeliaApothikis {
attribute kwdikosParaggeliasApothikis {$max+1},
element kwdikos {$kwdikos},
element posothta {$posothta},
element promhtheyths {$promhthyths},
element timhAgoras {$timhAgoras}
}});
```

Εικόνα 7.12: Συνάρτηση 'createParaggeliaApothikis'

Επιστρέφοντας στη συνάρτηση 'makeUpdatesApothiki', εικόνα 7.9, βρίσκουμε την κλήση της 'fullApothiki' στη γραμμή 99. Η συνάρτηση αυτή δεν είναι τίποτα παραπάνω από μία ανανέωση του αποθέματος της αποθήκης, μετά την παραγγελία της. Με αυτή τη συνάρτηση λοιπόν ολοκληρώνεται και η τελευταία διαδικασία μέσω XQuery για αυτόματες παραγγελίες.

```
declare function local:fullApothiki
($apothiki as xs:string, $proion as xs:string, $newApothema as xs:float) {
update value //apothiki[@onoma=$apothiki]/
proiontaApothema[proionIDREF=$proion]/apothema with $newApothema};
```

Εικόνα 7.13: Συνάρτηση 'fullApothiki'

## 8 Επεκτάσεις

Η eXist-db πέρα από βάση δεδομένων για XML παρέχει και μια πλήρη πλατφόρμα για την ανάπτυξη διαδικτυακών εφαρμογών βασισμένες σε XML καθώς και στις συσχετιζόμενες με αυτήν τεχνολογίες, όπως XQuery, XForms, XHTML κλπ. Μελετήσαμε λοιπόν τους τρόπους επέκτασης της παρούσας πτυχιακής εργασίας και ακολουθήσαμε τις οδηγίες που αναφέρονται εδώ [ ]. Αρχικοποιήσαμε έτσι μια διαδικτυακή εφαρμογή που συνδέθηκε με τα XML δεδομένα που συμμορφώνονται με το σχήμα του Super Market. Παρακάτω θα ακολουθήσει η περιγραφή των βημάτων αρχικοποίησης καθώς και ένα κομμάτι της υλοποίησης της web εφαρμογής που δημιουργήσαμε.

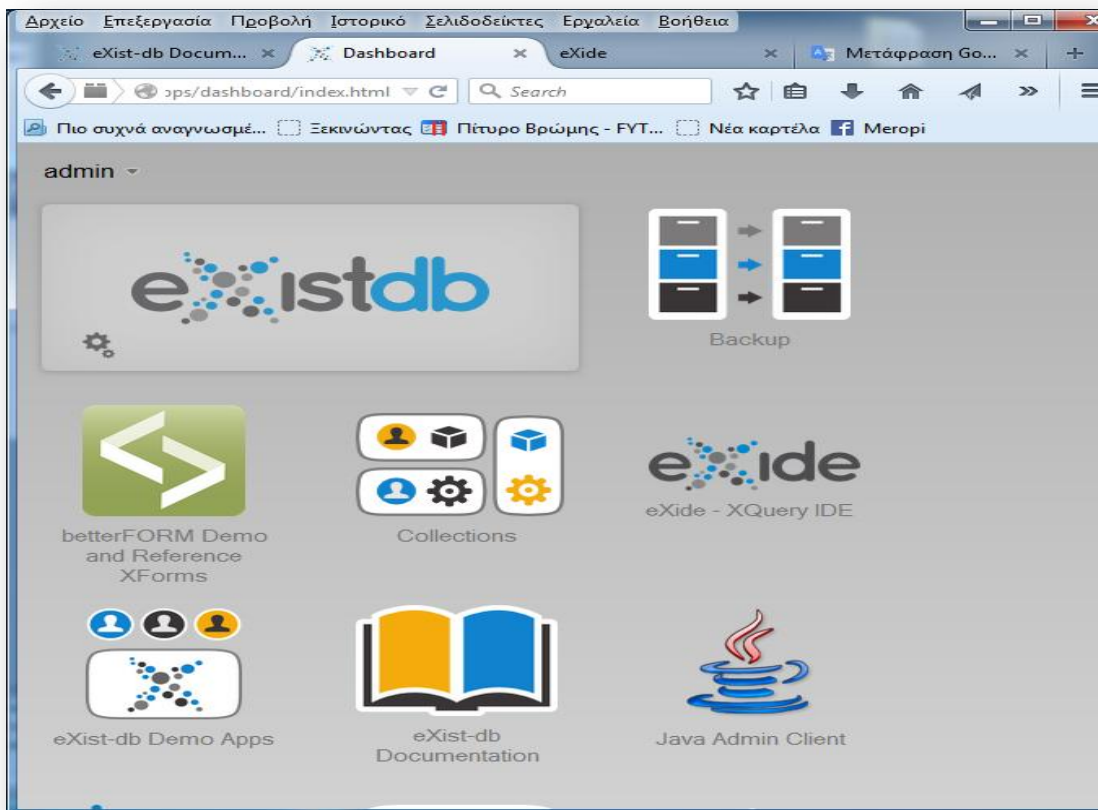
Ας δούμε όμως στο σημείο αυτό τις βασικές συνιστώσες της πλατφόρμας που μας παρέχει η eXist-db:

- Δυνατότητα δημιουργίας πακέτων για σπονδυλωτές(modular) εφαρμογές που μπορούν να ενσωματωθούν σε οποιαδήποτε εγκατάσταση της eXist-db που τρέχει.
- Ένα σύνολο από εργαλεία για την δημιουργία των πακέτων της εφαρμογής που ενσωματώνεται στο περιβάλλον της XQuery, το eXide.
- HTML templating framework που αποτελεί ένα περιβάλλον για τον ορθό διαχωρισμό των HTML σελίδων από τον κώδικα που υλοποιεί την λογική της εφαρμογής.
- Δυνατότητα ενσωμάτωσης με XForms για γρήγορη ανάπτυξη

### 8.1 Εκκίνηση δημιουργίας μια νέας εφαρμογής

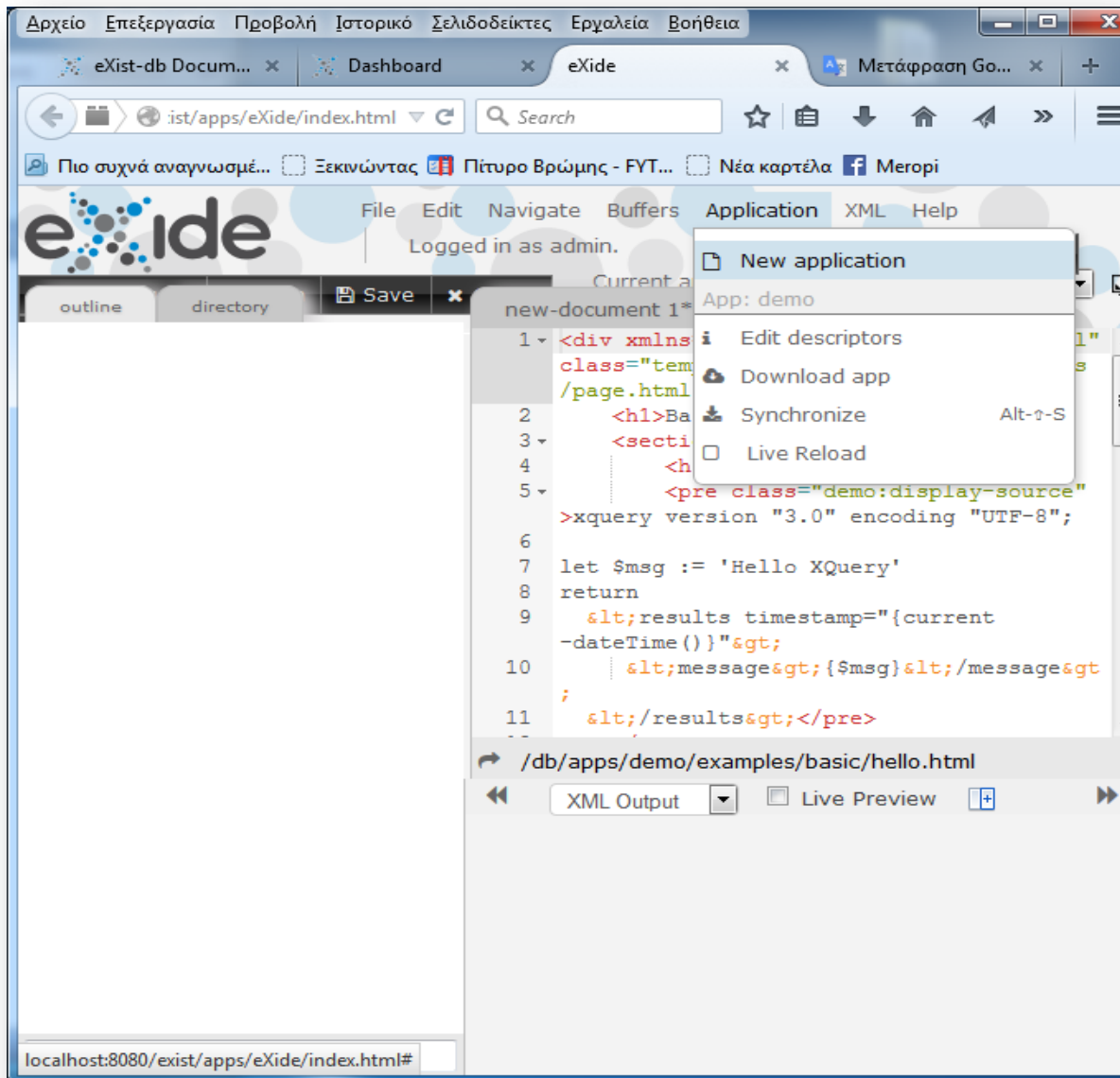
Στην παρούσα ενότητα παρουσιάζεται ένας προτεινόμενος τρόπος δημιουργίας μιας νέα εφαρμογής χρησιμοποιώντας το eXide.

1. Φόρτωση του κεντρικού μενού των εφαρμογών της eXistdb ανοίγοντας το τοπικό URL της εγκατάστασης <http://localhost:8080/exist/apps/dashboard/index.html> όπως φαίνεται στην εικόνα:



Εικόνα 8.1: Τοπικό URL

2. Μετάβαση στο περιβάλλον eXide.
3. Από το κεντρικό μενού επιλέγουμε Applications > New Application και ανοίγει ο Deployment Editor:

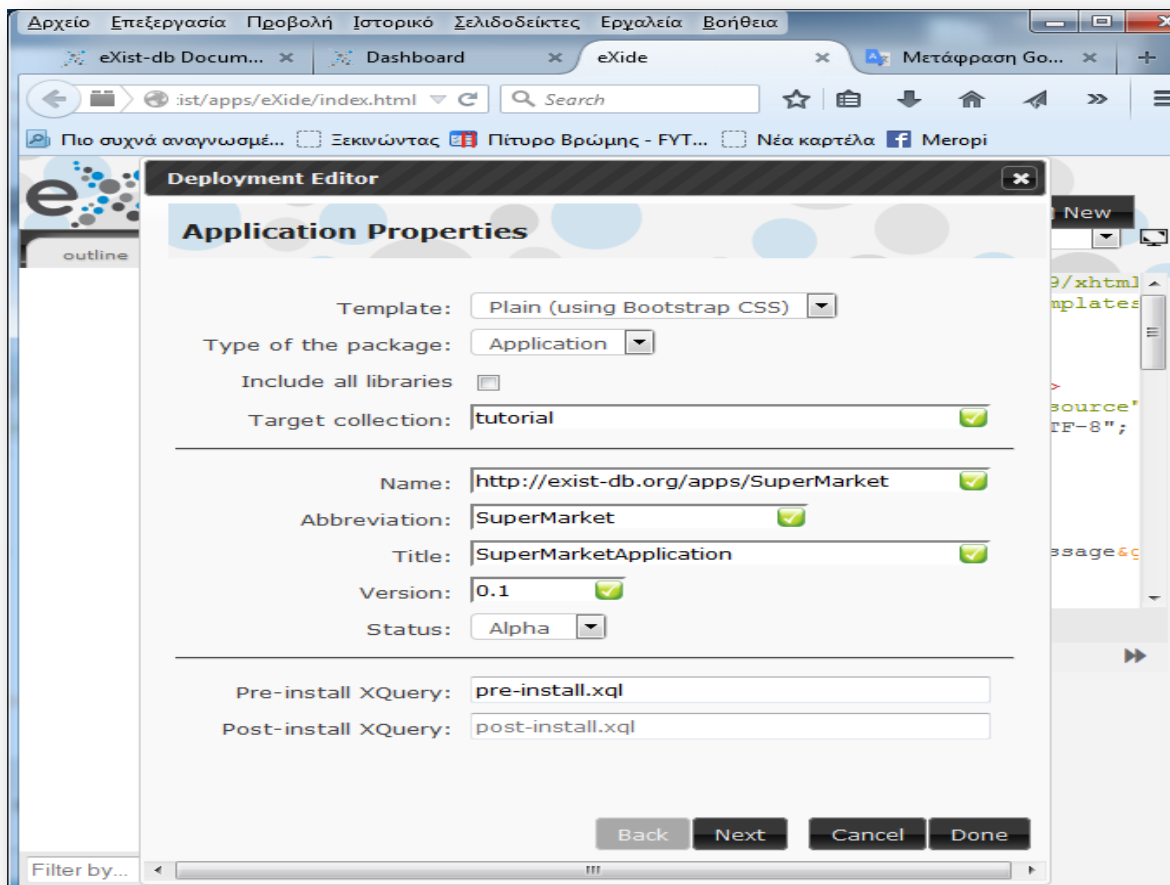


Εικόνα 8.2: eXide, Deployment editor

4. Συμπληρώνουμε την φόρμα επιλέγοντας το πρότυπο(template), target collection, ένα όνομα και μια συντόμευση καθώς και έναν τίτλο για την εφαρμογή. Όλα τα υπόλοιπα πεδία της φόρμας είναι προαιρετικά αλλά ας κάποια περιληπτικά στοιχεία για το καθένα.
  - a. Πρότυπο: Το πρότυπο χρησιμοποιείτε για την δημιουργία της νέας εφαρμογής. Στην παρούσα έκδοση της eXist-db υποστηρίζονται τρία πρότυπα: το 'exist-db design' που βασίζεται στη σχεδίαση και στην διαμόρφωση των σελίδων βάσει ενός standard της eXist-db. Το 'απλό(plain)' πρότυπο που δημιουργεί μια βασική διαμόρφωση χωρίς κάποια πρόσθετες εξειδικεύσεις της eXist-db. Να σημειώσουμε ότι και τα δυο πρότυπα χρησιμοποιούν την CSS βιβλιοθήκη για την δημιουργία των styles καθώς και το Module HTML templating XQuery για να επιτευχθεί ένα καθαρός διαχωρισμός με τα styles. Τέλος, υπάρχει και το 'Empty Package' που δημιουργεί ένα άδειο πακέτο εφαρμογής προσθέτοντας απλά κάποια περιγραφικά αρχεία (descriptor).
  - b. Target collection: Η συλλογή όπου θα γίνει η εγκατάσταση της εφαρμογής.

- c. Όνομα: Ένα μοναδικό αναγνωριστικό της εφαρμογής. Εδώ προτείνεται να γίνει χρήση URI (Unified Resource Identifier) για να αποφευχθούν κολλήματα ονοματοδοσίας.
- d. Συντόμευση (Abbreviation): Μια συντόμευση της εφαρμογής που θα χρησιμοποιηθεί και ως το όνομα του αρχείου για το τελικό .xar πακέτο καθώς και ως όνομα μέσα στην συλλογής στην οποία θα εγκατασταθεί η εφαρμογή.
- e. Τίτλος: Μια σύντομη περιγραφή της εφαρμογής που εμφανίζεται στον χρήστη.

Η φόρμα φαίνεται συμπληρωμένη στην παρακάτω εικόνα και πατώντας 'Done' θα δημιουργηθεί μια ιεραρχία συλλογής για την εφαρμογή με βάση το πρότυπο που επιλέχτηκε.

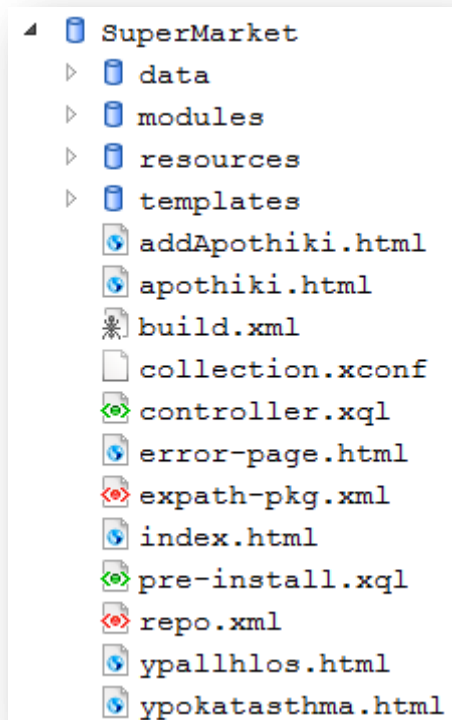


Εικόνα 8.3: Application Properties

Στη συνέχεια εμφανίζεται ο Open Document διάλογος που μας επιτρέπει να ξεκινήσουμε την επεξεργασία των αρχείων. Για να ελέγξουμε αν η εφαρμογή έχει δημιουργηθεί με επιτυχία επιλέγουμε το αρχείο index.html και στη συνέχεια από το μενού επιλέγουμε Run και ανοίγει σε ξεχωριστό παράθυρο του browser το περιεχόμενο της Index.html σελίδας. Η αρχικοποίηση μιας web εφαρμογής αποτελεί την βασική θεμελίωση πάνω στην οποία στη συνέχεια θα χτιστεί ο κώδικας της διεπαφής που θα καλεί όλα τα XQuery functions για την διαχείριση της βάσης δεδομένων του SuperMarket.

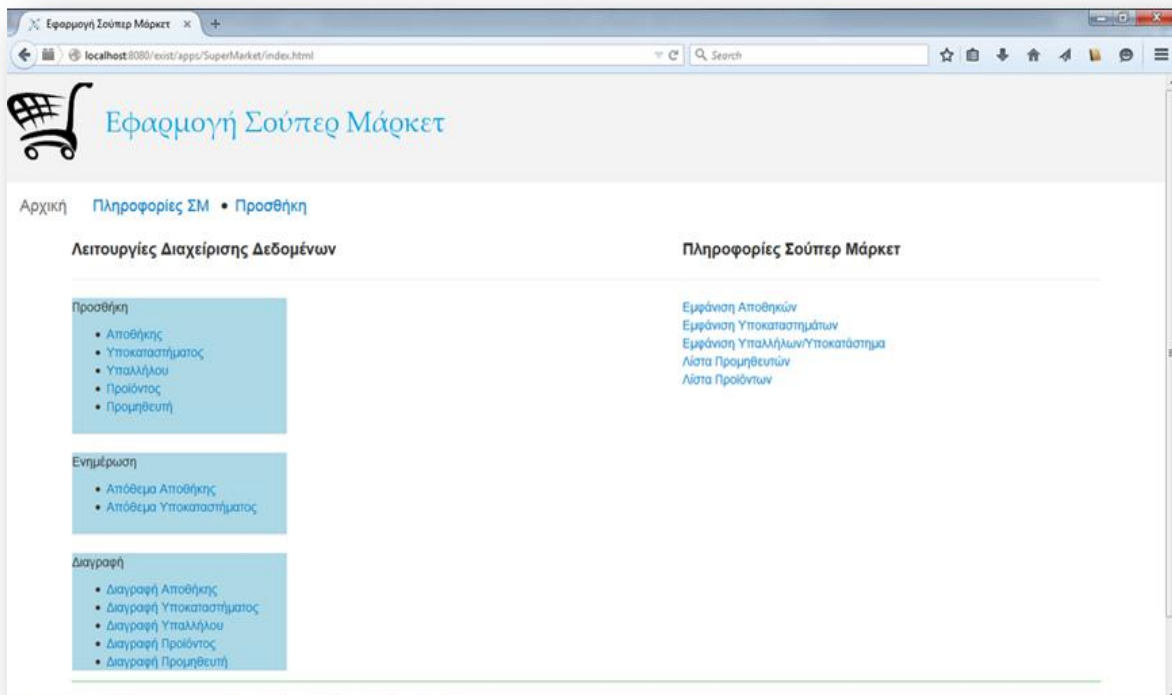
## 8.2 Περιγραφή της διαδικτυακής εφαρμογής

Για την ανάπτυξη του κώδικα της εφαρμογής χρησιμοποιήσαμε το περιβάλλον oXygen και στην εικόνα που ακολουθεί φαίνεται η δομή των φακέλων και αρχείων που δημιουργήθηκαν με τα βήματα που περιγράψαμε παραπάνω.



Εικόνα 8.4: Δομή φακέλων στο oXygen μετά από τα βήματα για δημιουργία της αρχικής εφαρμογής

Η αρχική σελίδα της εφαρμογής μας φαίνεται στην επόμενη εικόνα και αναπτύχθηκε εισάγοντας HTML κώδικα στην index.html σελίδα.



Εικόνα 8.5: Αρχική σελίδα της εφαρμογής

Στη συνέχεια υλοποιήσαμε συναρτήσεις που εμφανίζουν στοιχεία του 'SuperMarket.xml' και τις ενσωματώσαμε στο αρχείο modules > app.xql. Η πλατφόρμα ανάπτυξης ορίζει το συγκεκριμένο αρχείο να χρησιμοποιείται για την ενσωμάτωση του XQuery κώδικα ώστε να διαχωρίζεται από τον

κώδικα που παράγει την διεπαφή. Η παρακάτω συνάρτηση για παράδειγμα υλοποιήθηκε με βάση τις οδηγίες και καλείται μέσα στον HTML κώδικα για να τυπώσει το όνομα της αποθήκης:

```
declare
  %templates:wrap
function app:print-apothiki-onoma($node as node(), $model as map(*)) {
  $model("apothiki")/@onoma/string()
};
```

Η κλήση αντίστοιχα μέσα στην HTML είναι:

```
<table class="app:apothiki" border="1" style="width:80%">
  <thead>
    <tr>
      <th>Όνομα Αποθήκης</th>
      <th>Οδός</th>
      <th>Αριθμός</th>
      <th>Πόλη</th>
    </tr>
  </thead>
  <tr class="templates:each?from=apothikes&to=apothiki">
    <td class="app:print-apothiki-onoma"/>
    <td class="app:print-apothiki-odos"/>
    <td class="app:print-apothiki-Arithmos"/>
    <td class="app:print-apothiki-Poli"/>
    <td class="app:print-ypokatasthma-onoma"/>
  </tr>
</table>
```



## 9 Βιβλιογραφία

- [1] <http://el.wikipedia.org/wiki/SQL>
- [2] <http://el.wikipedia.org/wiki/%CE%9A%CE%B1%CF%84%CE%B1%CE%BD%CE%B5%CE%BC%CE%B7%CE%BC%CE%AD%CE%BD%CE%B5%CF%82%CE%92%CE%AC%CF%83%CE%B5%CE%B9%CF%82%CE%94%CE%B5%CE%B4%CE%BF%CE%BC%CE%AD%CE%BD%CF%89%CE%BD>
- [3] <http://slideplayer.gr/slide/2632897/>
- [4] <http://dide.flo.sch.gr/Plinet/Tutorials/Tutorials-DataBasesTheory.html>
- [5] <http://el.wikipedia.org/wiki/%CE%9C%CE%AD%CF%83%CE%BF%CE%B1%CF%80%CE%BF%CE%B8%CE%AE%CE%BA%CE%B5%CF%85%CF%83%CE%B7%CF%82%CE%B4%CE%B5%CE%B4%CE%BF%CE%BC%CE%AD%CE%BD%CF%89%CE%BD#.CE.9C.CE.B1.CE.B3.CE.BD.CE.B7.CF.84.CE.B9.CE.BA.CE.AE.CF.84.CE.B1.CE.B9.CE.BD.CE.AF.CE.B1>
- [6] [http://stat.med.upatras.gr/db/Intro\\_Databases.pdf](http://stat.med.upatras.gr/db/Intro_Databases.pdf)
- [7] <http://el.wikipedia.org/wiki/XML>
- [8] <http://www.it.uom.gr/project/xml/Home%20Page.htm>
- [9] [http://nefeli.lib.teicrete.gr/browse/stef/epp/2012/ValyrakisGeorgios%2cMilolidakisIoannis/attached-document-1351692526-605712-25415/ValirakisGeorgios\\_MilolidakisIoannis2012.pdf](http://nefeli.lib.teicrete.gr/browse/stef/epp/2012/ValyrakisGeorgios%2cMilolidakisIoannis/attached-document-1351692526-605712-25415/ValirakisGeorgios_MilolidakisIoannis2012.pdf)
- [10] <http://www.w3.org/standards/xml/schema>
- [11] [http://en.wikipedia.org/wiki/Entity%E2%80%93relationship\\_model](http://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model)
- [12] <http://el.wikipedia.org/wiki/HTML#.CE.99.CF.83.CF.84.CE.BF.CF.81.CE.AF.CE.B1>
- [13] “Συστήματα Βάσεων Δεδομένων” 4<sup>η</sup> Έκδοση, Silberschatz, Korth, Sudarshan, Εκδόσεις Μ. Γκούρδας.
- [14] Chengfei Liu and Jianxin Li, Designing Quality XML Schemas from E-R Diagrams.
- [15] [http://www2.informatik.hu-berlin.de/~bourospa/docs/diploma\\_thesis.pdf](http://www2.informatik.hu-berlin.de/~bourospa/docs/diploma_thesis.pdf)
- [16] <http://en.wikipedia.org/wiki/XQuery>
- [17] <http://en.wikipedia.org/wiki/EXist>
- [18] <http://exist-db.org/exist/apps/doc/quickstart.xml>
- [19] [http://www.tutorialspoint.com/html/html\\_layouts.htm](http://www.tutorialspoint.com/html/html_layouts.htm)
- [20] [http://en.wikipedia.org/wiki/Oxygen\\_XML\\_Editor](http://en.wikipedia.org/wiki/Oxygen_XML_Editor)
- [21] [http://nefeli.lib.teicrete.gr/browse/stef/epp/2011/MathioulakisManolis,ChairetisDionysios/attached-document-1320150866-361739-32090/Chairetis\\_Mathioulakis2011.pdf](http://nefeli.lib.teicrete.gr/browse/stef/epp/2011/MathioulakisManolis,ChairetisDionysios/attached-document-1320150866-361739-32090/Chairetis_Mathioulakis2011.pdf)
- [22] ΕΙΣΑΓΩΓΗ ΣΤΙΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ, Intro\_Databases.pdf
- [23] «Τι κρύβεται» πίσω από μια Επιχειρησιακή Αναφορά (report): XML& XSD για αρχάριους, documentmodelling\_xml\_xsd.pptx
- [24] <file:///C:/Users/Vasiliki/Desktop/eXist%20%20NoSQL%20Document%20Database%20and%20Application%20Platform%20-%20Erik%20Siegel,%20Adam%20Retter%20-%20CE%92%CE%B9%CE%B2%CE%BB%CE%AF%CE%B1%20Google.htm>
- [25] <https://books.google.gr/books?id=9uvSBQAAQBAJ&pg=PT40&dq=EXIST+ERIC+SIEGEL&hl=el&sa=X&ei=uEcRVejUGoW8UZCBgfAJ&ved=0CB4Q6AEwAA#v=onepage&q=EXIST%20ERIC%20SIEGEL&f=false>

## 10 Παράρτημα Ν° 1

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="SuperMarket">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="apothiki">
            <xs:complexType>
              <xs:sequence>
                <xs:element ref="dieythinsi"/>
                <xs:element name="ypokatasthma" maxOccurs="unbounded">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element ref="dieythinsi"/>
                      <xs:element name="ypallhlos" maxOccurs="unbounded">
                        <xs:complexType>
                          <xs:sequence>
                            <xs:element name="onomatepwnymo" type="xs:string"/>
                            <xs:element name="misthos" type="xs:float"/>
                            <xs:element ref="dieythinsi"/>
                            <xs:element ref="thlefwno" minOccurs="1"
maxOccurs="unbounded"/>
                          </xs:sequence>
                        <xs:attribute name="at" type="xs:string" use="required"/>
                      </xs:complexType>
                    <xs:key name="key_ypallhlos">
                      <xs:selector xpath="ypallhlos"/>
                      <xs:field xpath="@at"/>
                    </xs:key>
                  </xs:element>
                <xs:element name="paraggelia" minOccurs="0"
maxOccurs="unbounded">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="kwdikos" type="xs:IDREF"/>
                      <xs:element name="posothta" type="xs:float"/>
                    </xs:sequence>
                    <xs:attribute name="kwdikosParaggelias" type="xs:int"
use="required"/>
                  </xs:complexType>
                <xs:key name="key_paraggelia">
                  <xs:selector xpath="paraggelia"/>
                  <xs:field xpath="ypokatasthma/@onoma"/>
                  <xs:field xpath="@kwdikosParaggelias"/>
                </xs:key>
              </xs:element>
            <xs:element name="proiontaApothema" type="proiontaLista"
minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="onoma" type="xs:string" use="required"/>
        </xs:complexType>
      <xs:key name="key_ypokatasthma">
        <xs:selector xpath="ypokatasthma"/>

```

```

        <xs:field xpath="@onoma"/>
    </xs:key>
</xs:element>
<xs:element name="proiontaApothema" type="proiontaLista" minOccurs="1"
    maxOccurs="unbounded"/>
    <xs:element name="promhtheythsApothikis" type="promhtheytesLista"
    minOccurs="1"
maxOccurs="unbounded"/>
    <xs:element name="paraggeliaApothikis" minOccurs="0"
maxOccurs="unbounded">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="kwdikos" type="xs:IDREF"/>
            <xs:element name="posothta" type="xs:float"/>
            <xs:element name="promhtheyths" type="xs:IDREF"/>
            <xs:element name="timhAgoras" type="xs:float"/>
        </xs:sequence>
        <xs:attribute name="kwdikosParaggeliasApothikis" type="xs:int"
            use="required"/>
    </xs:complexType>
    <xs:key name="key_paraggeliaApothikis">
        <xs:selector xpath="paraggeliaApothikis"/>
        <xs:field xpath="apothiki/@onoma"/>
        <xs:field xpath="@kwdikosParaggeliasApothikis"/>
    </xs:key>
    </xs:element>
</xs:sequence>
    <xs:attribute name="onoma" type="xs:string" use="required"/>
</xs:complexType>
    <xs:key name="key_apothiki">
        <xs:selector xpath="apothiki"/>
        <xs:field xpath="@onoma"/>
    </xs:key>
</xs:element>
    <xs:element name="proionta" type="proionPerigrafh" minOccurs="1"
    maxOccurs="unbounded"/>
    <xs:element name="promhtheyths" type="promhtheythsPerigrafh" minOccurs="1"
    maxOccurs="unbounded"/>

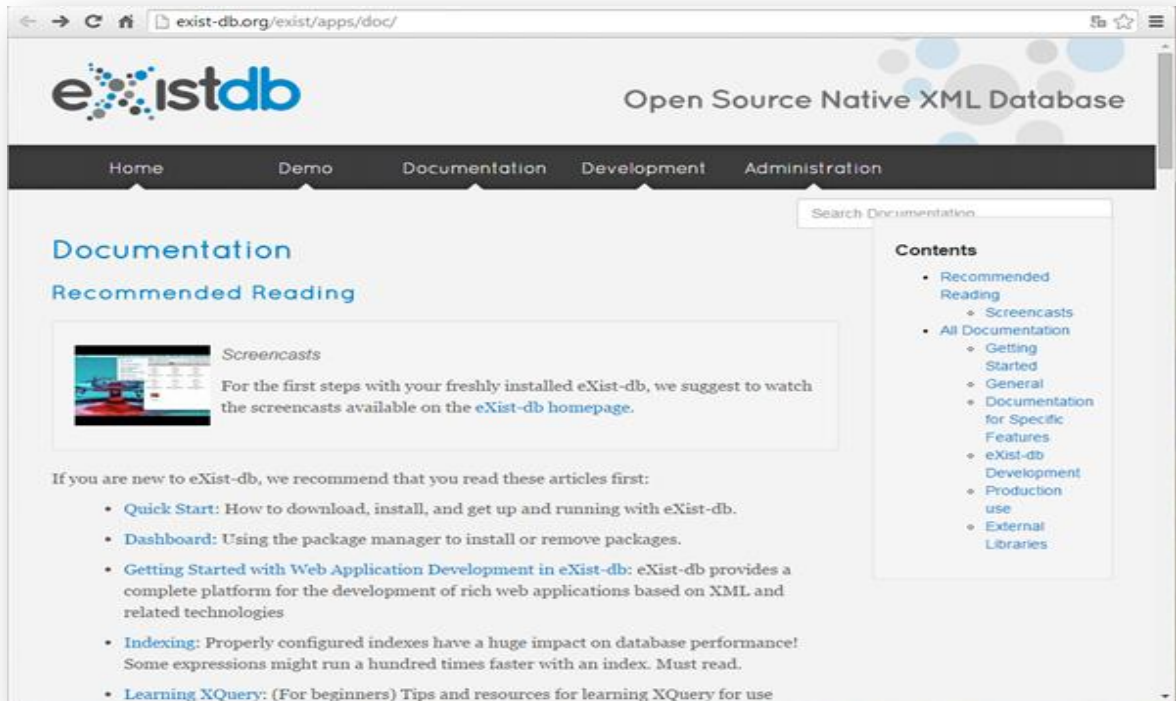
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:complexType name="promhtheytesLista">
    <xs:sequence>
        <xs:element name="promitheutisIDREF" type="xs:IDREF"/>
        <xs:element name="timhPwlhshs" type="xs:float"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="proiontaLista">
    <xs:sequence>
        <xs:element name="proionIDREF" type="xs:IDREF"/>
        <xs:element name="apothema" type="xs:float"/>
        <xs:element name="minApothema" type="xs:float"/>
    </xs:sequence>
</xs:complexType>

```

```
<xs:complexType name="promhtheythsPerigrafh">
  <xs:sequence>
    <xs:element name="afm" type="xs:ID"/>
    <xs:element name="onomatepwnymo" type="xs:string"/>
    <xs:element ref="dieythinsi"/>
    <xs:element ref="thlefwno" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="proionPerigrafh">
  <xs:sequence>
    <xs:element name="proionKwdikos" type="xs:ID"/>
    <xs:element name="perigrafh" type="xs:string"/>
    <xs:element name="eidos" type="xs:string"/>
    <xs:element name="paragwgos" type="xs:string"/>
    <xs:element name="timh" type="xs:float"/>
    <xs:element name="promhtheyths" type="promhtheytesLista" minOccurs="1"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="thlefwno">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="eidosThlefwnou" type="phoneTypes"/>
      <xs:element name="number" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:simpleType name="phoneTypes">
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="κινητό"/>
    <xs:enumeration value="σταθερό"/>
    <xs:enumeration value="FAX"/>
  </xs:restriction>
</xs:simpleType>
<xs:element name="dieythinsi">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Odos" type="xs:string"/>
      <xs:element name="Arithmos" type="xs:int"/>
      <xs:element name="Poli" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

## 11 Παράρτημα Ν° 2

### 11.1 Βήματα εγκατάστασης της eXist-db



Εικόνα 11.1: Εγκατάσταση της eXist-db, βήμα 1

Επισκεπτόμαστε αρχικά την επίσημη ιστοσελίδα της eXist-db, <http://exist-db.org> και μεταφερόμαστε στην καρτέλα “Documentation”. Εκεί θα βρούμε αναλυτικά τα βήματα που θα μας οδηγήσουν στην επιτυχή εγκατάσταση της eXist-db καθώς και τις προϋποθέσεις που χρειάζονται για τον σκοπό αυτό. Όπως θα δούμε στην επόμενη εικόνα, το μόνο που χρειαζόμαστε για να αρχίσουμε την εγκατάσταση της eXist-db, είναι η java.



Εικόνα 11.2: Εγκατάσταση της java, βήμα 1

Αν λοιπόν δεν την έχουμε ήδη εγκατεστημένη, μεταβαίνουμε στην ιστοσελίδα <http://www.java.com/en/download/> και ακολουθούμε τα βήματα εγκατάστασης java, εικόνα 11.3 έως εικόνα 6.6, αλλιώς τα αγνοούμε πηγαίνοντας κατευθείαν στην εγκατάσταση της eXist-db, εικόνα 11.7.



Εικόνα 11.3: Εγκατάσταση της java, βήμα 2



Εικόνα 11.4: Εγκατάσταση της java, βήμα 3

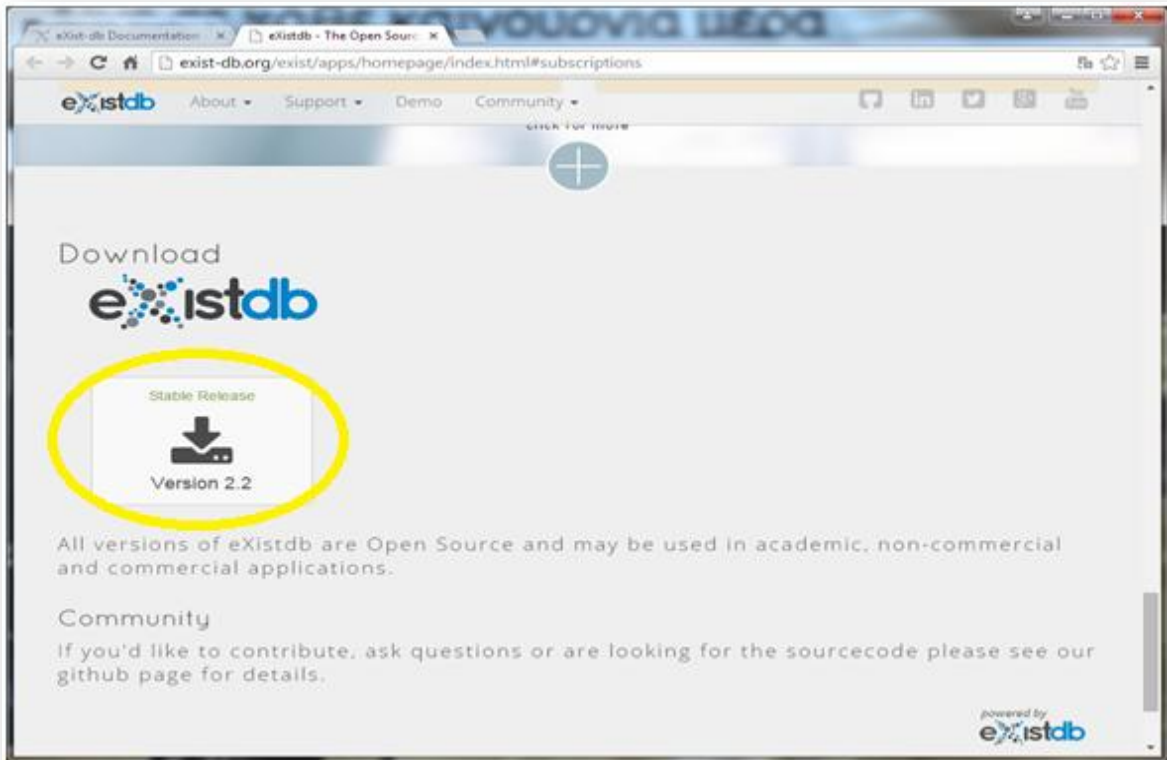


Εικόνα 11.5: Εγκατάσταση της java, βήμα 4

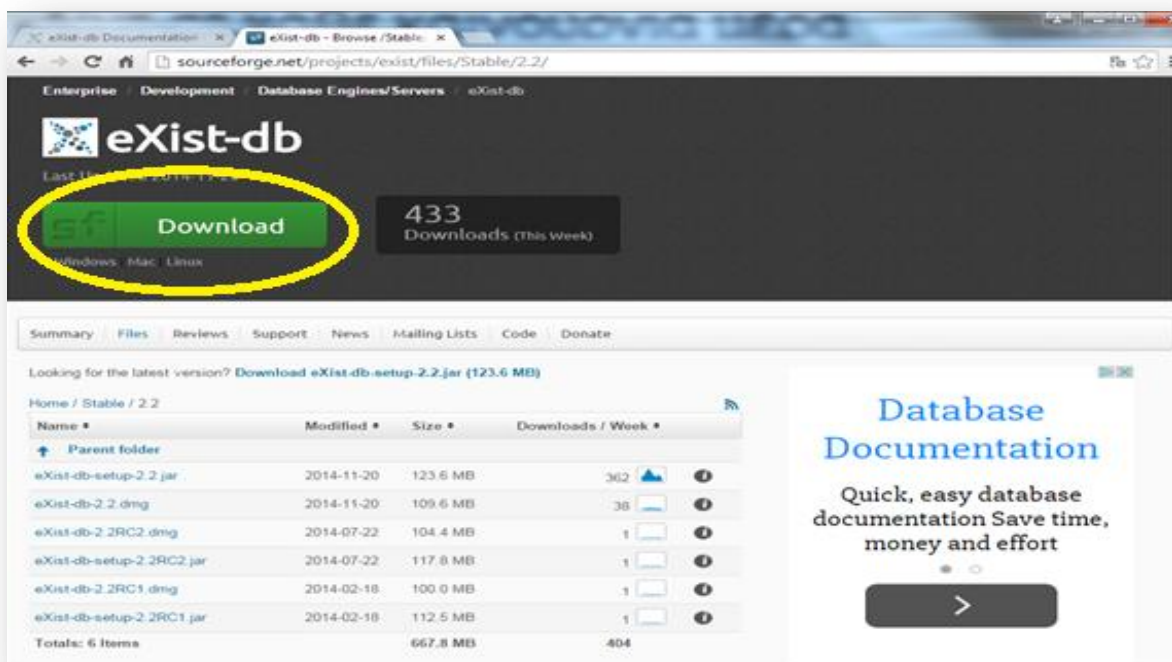


Εικόνα 11.6: Εγκατάσταση της java, βήμα 5

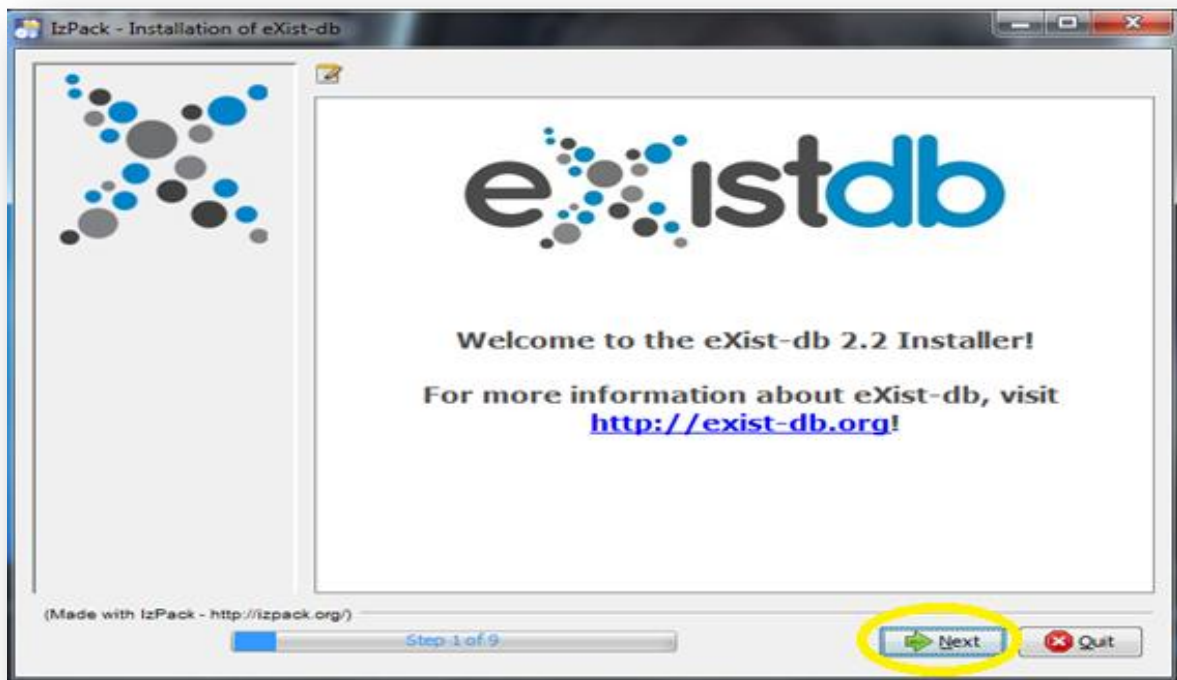




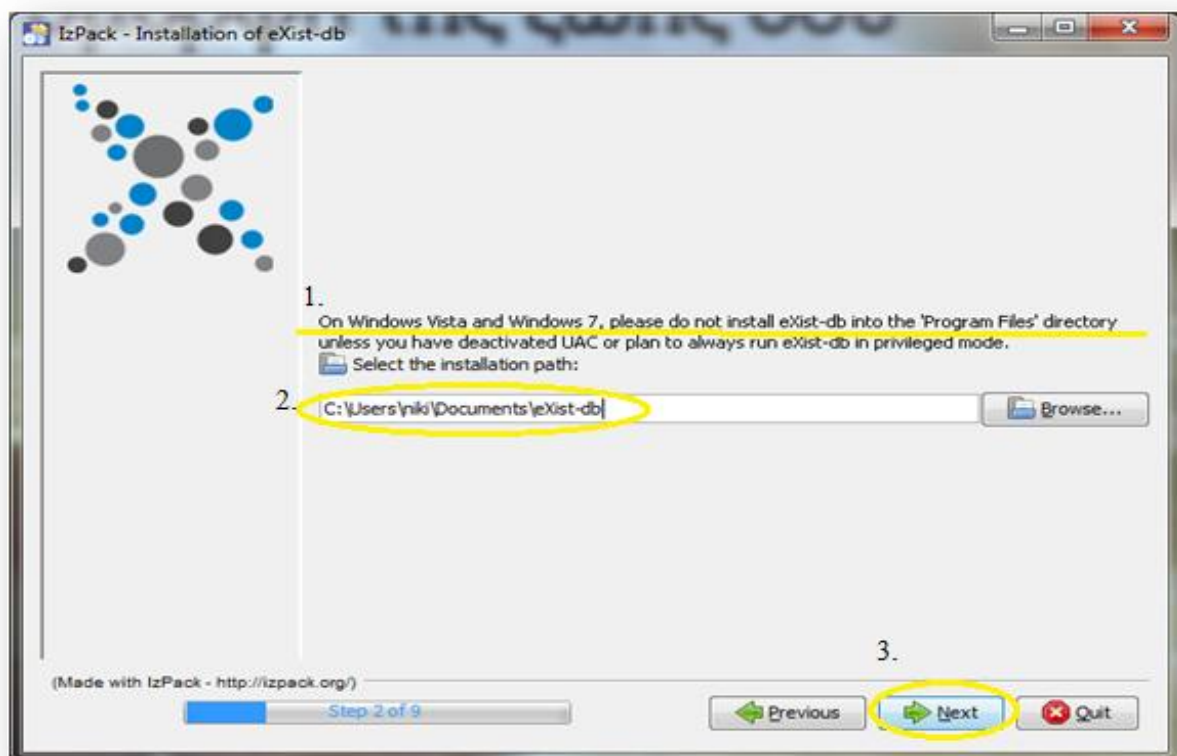
Εικόνα 11.7: Εγκατάσταση της eXist-db, βήμα 2



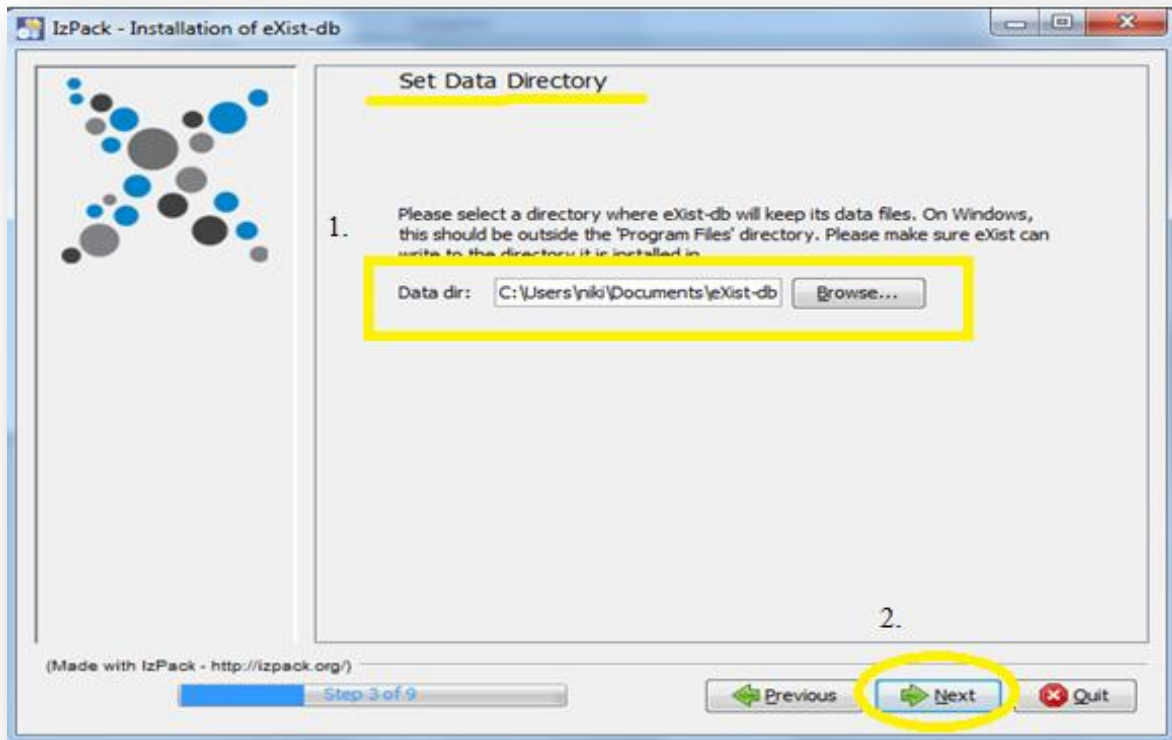
Εικόνα 11.8: Εγκατάσταση της eXist-db, βήμα 3



Εικόνα 11.9: Εγκατάσταση της eXist-db, βήμα 4



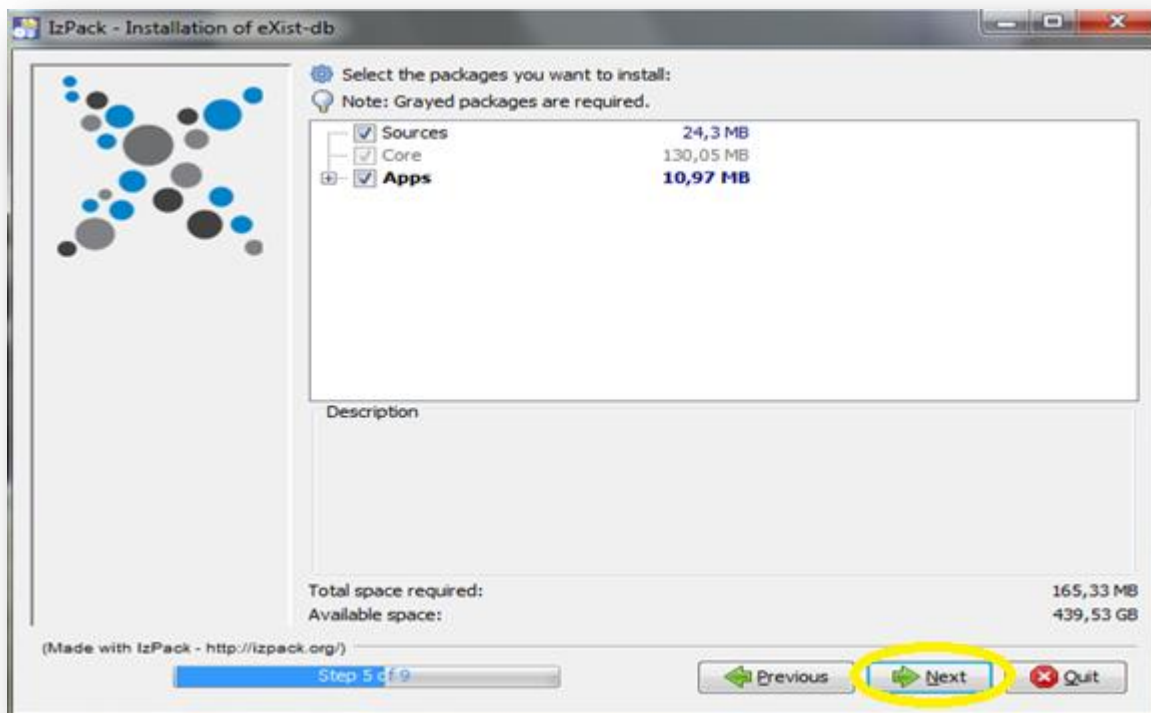
Εικόνα 11.10: Εγκατάσταση της eXist-db, βήμα 5



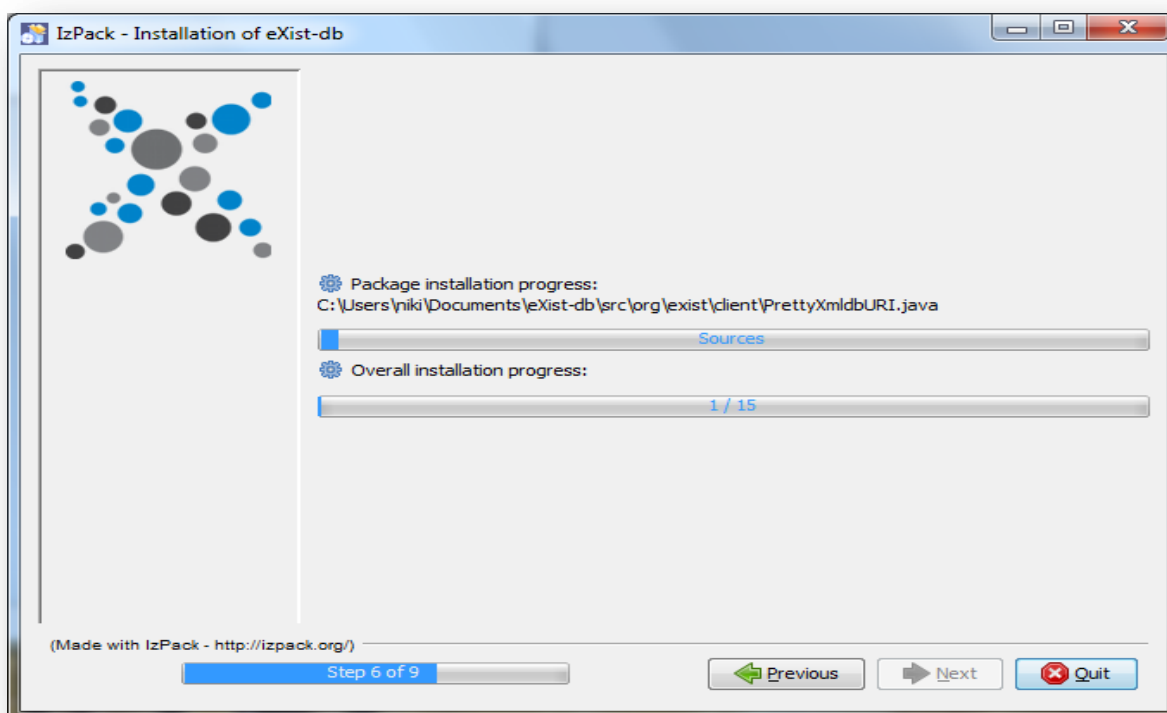
Εικόνα 11.11: Εγκατάσταση της eXist-db, βήμα 6



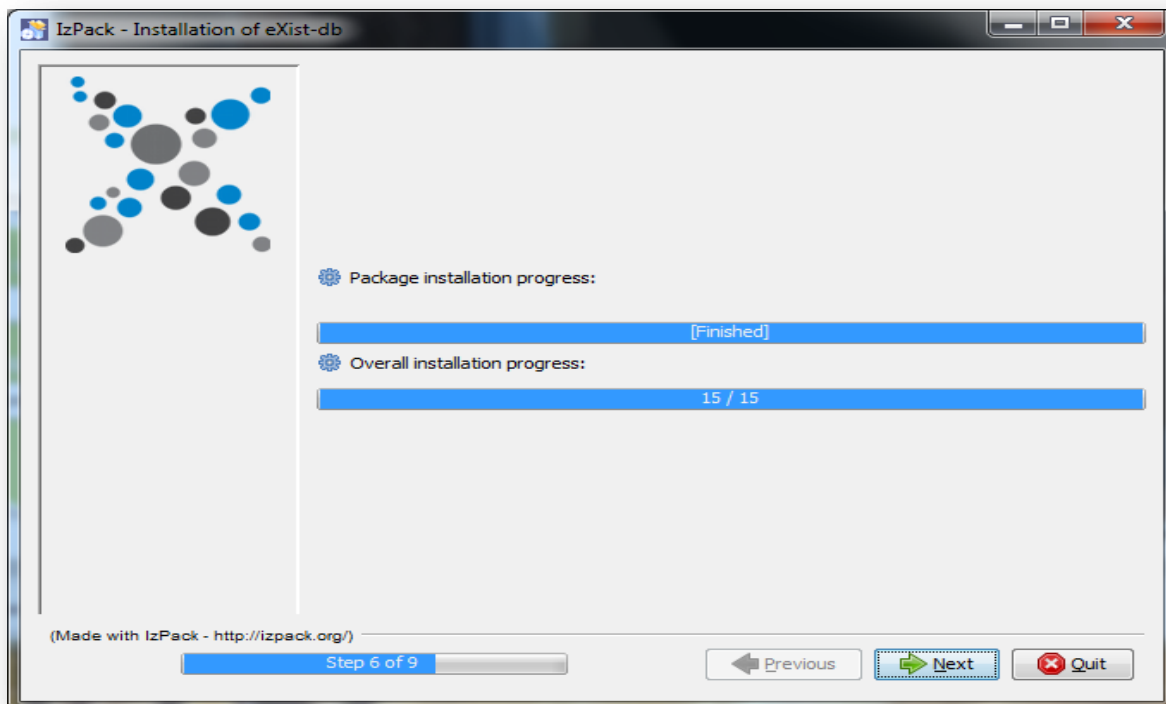
Εικόνα 11.12: Εγκατάσταση της eXist-db, βήμα 7



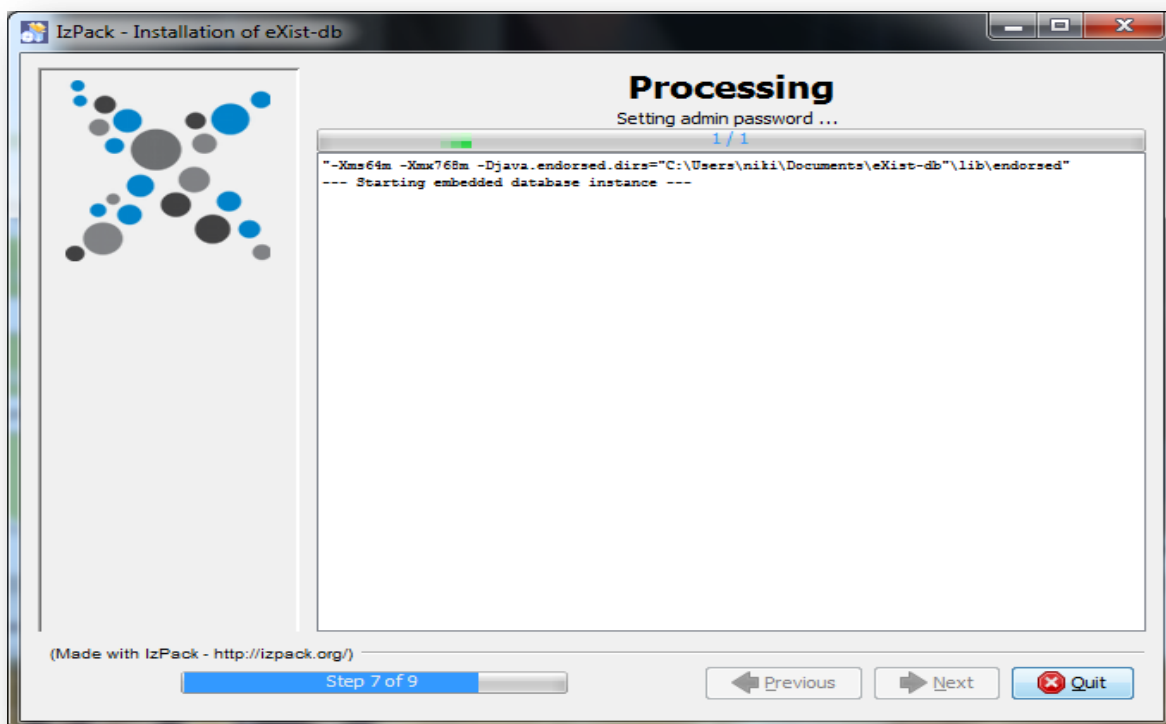
Εικόνα 11.13: Εγκατάσταση της eXist-db, βήμα 8



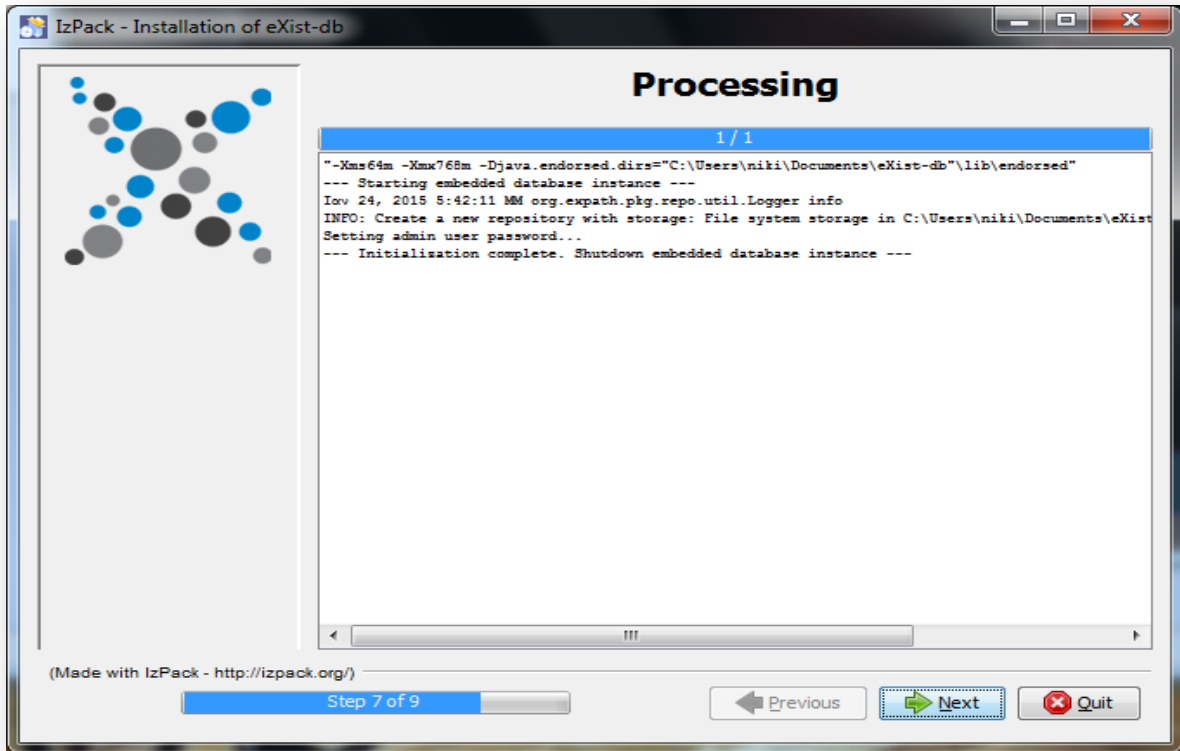
Εικόνα 11.14: Εγκατάσταση της eXist-db, βήμα 9



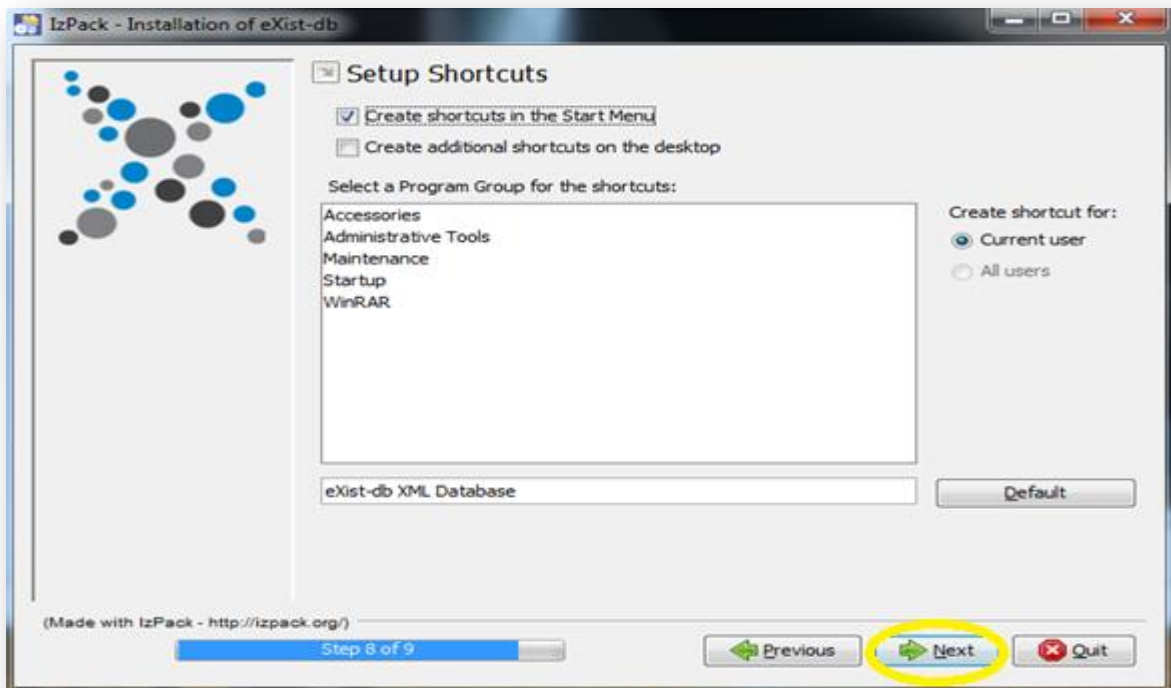
Εικόνα 11.15: Εγκατάσταση της eXist-db, βήμα 10



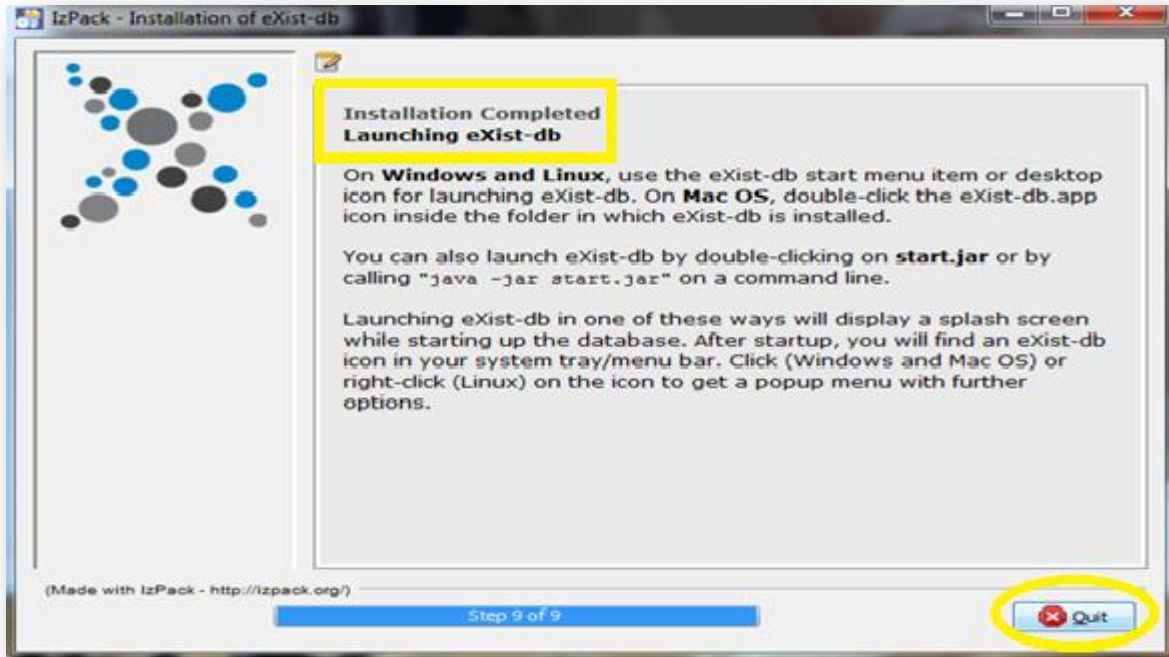
Εικόνα 11.16: Εγκατάσταση της eXist-db, βήμα 11



Εικόνα 11.17: Εγκατάσταση της eXist-db, βήμα 12



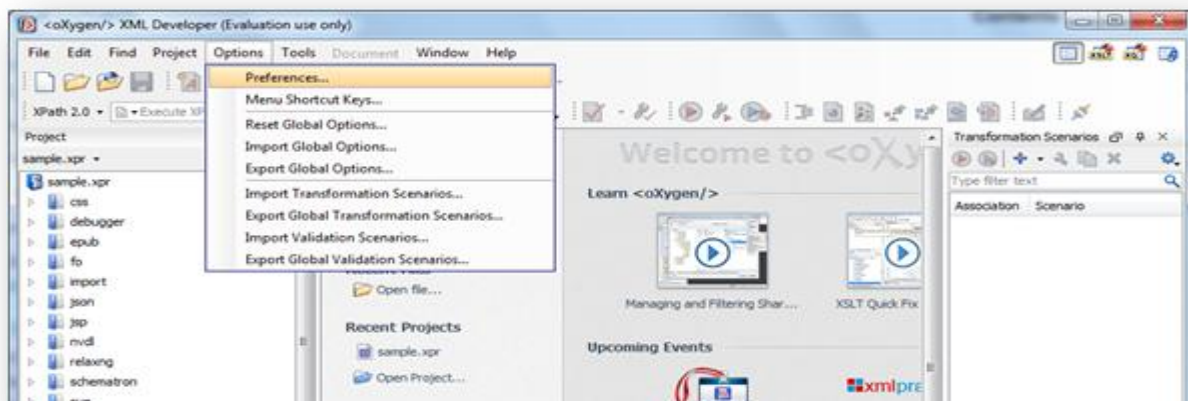
Εικόνα 11.18: Εγκατάσταση της eXist-db, βήμα 13



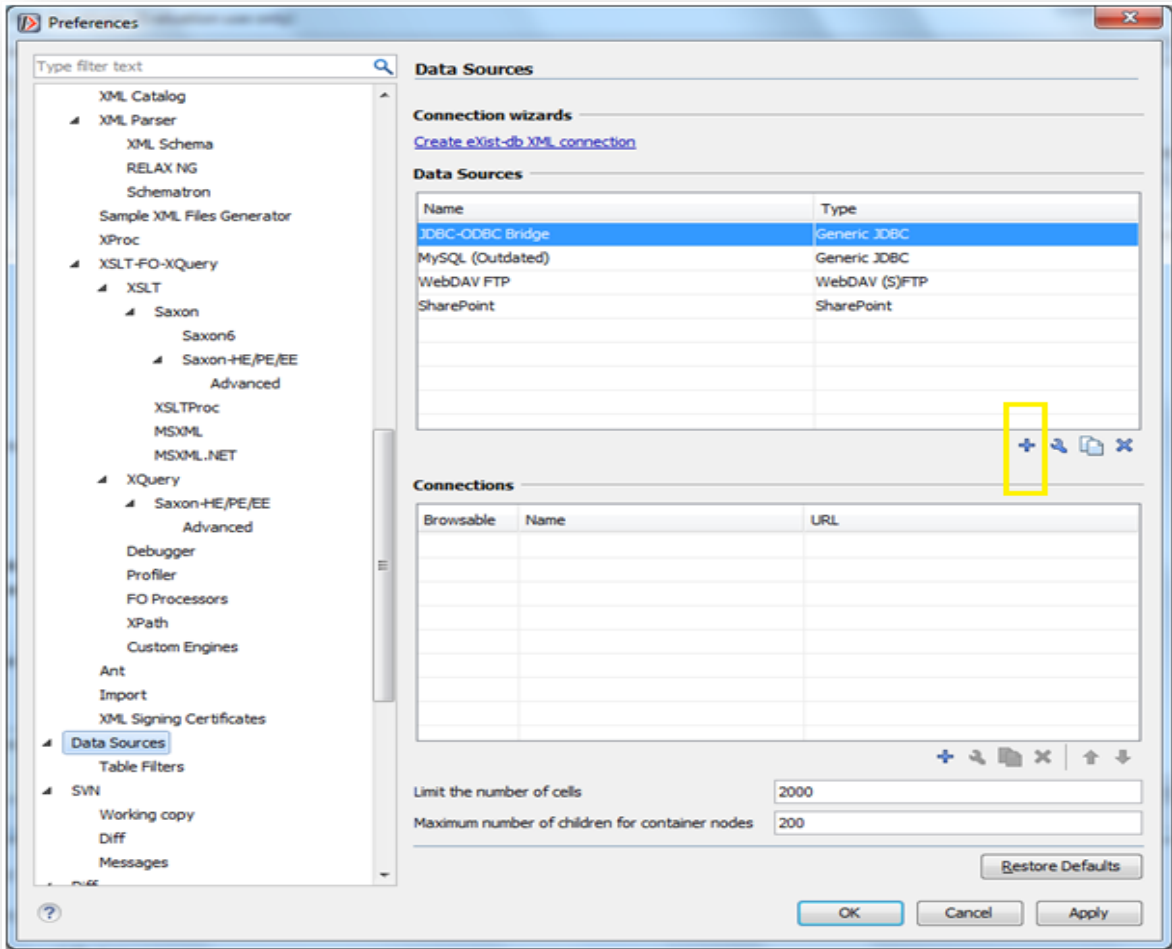
Εικόνα 11.19: Εγκατάσταση της eXist-db, βήμα 14

## 11.2 Σύνδεση της eXist-db με το περιβάλλον ανάπτυξης oxygen developer

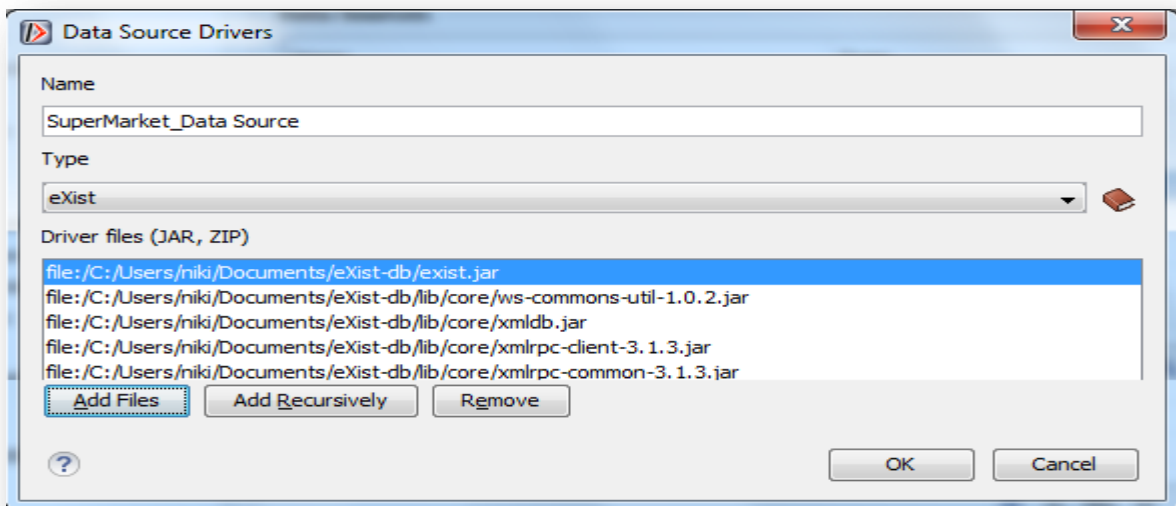
Αφού λοιπόν εγκαταστήσαμε με επιτυχία την eXist-db, προχωράμε τώρα στην ένωσή της με έναν editor ο οποίος θα υποστηρίζει XML και συγκεκριμένα, αυτός που επιλέξαμε εμείς να χρησιμοποιήσουμε είναι ο oxygen. Στην ιστοσελίδα <http://exist-db.org/exist/apps/doc/oxygen.xml> βρήκαμε την βοήθεια που χρειαζόμασταν για τον τρόπο ένωσης της βάσης δεδομένων με το περιβάλλον ανάπτυξης και τα βήματα που ακολουθήσαμε παραθέτουμε σε εικόνες :



Εικόνα 11.20: Ένωση eXist-db με oXygen, βήμα 1

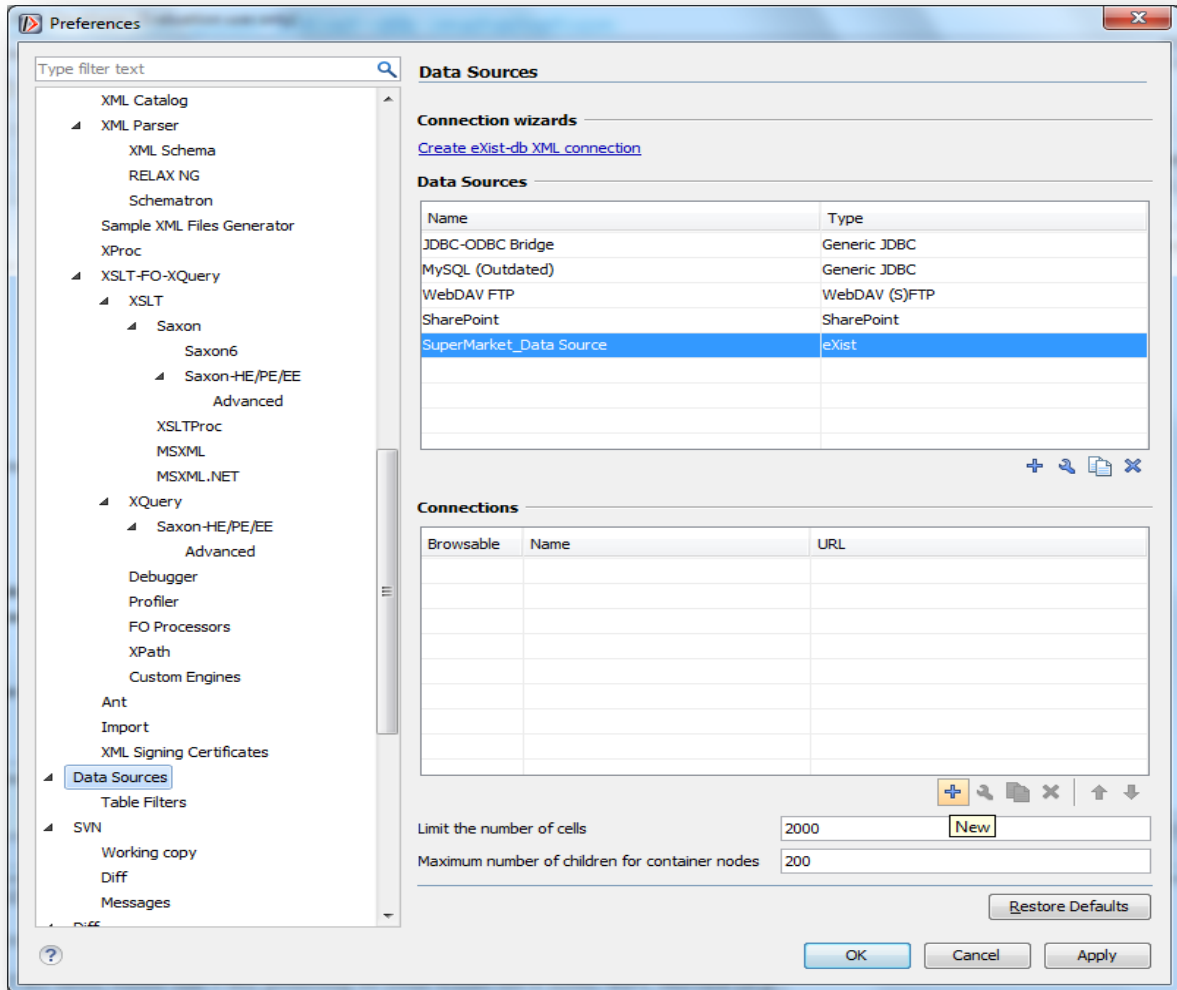


Εικόνα 11.21: Ένωση eXist-db με oXygen, βήμα 2

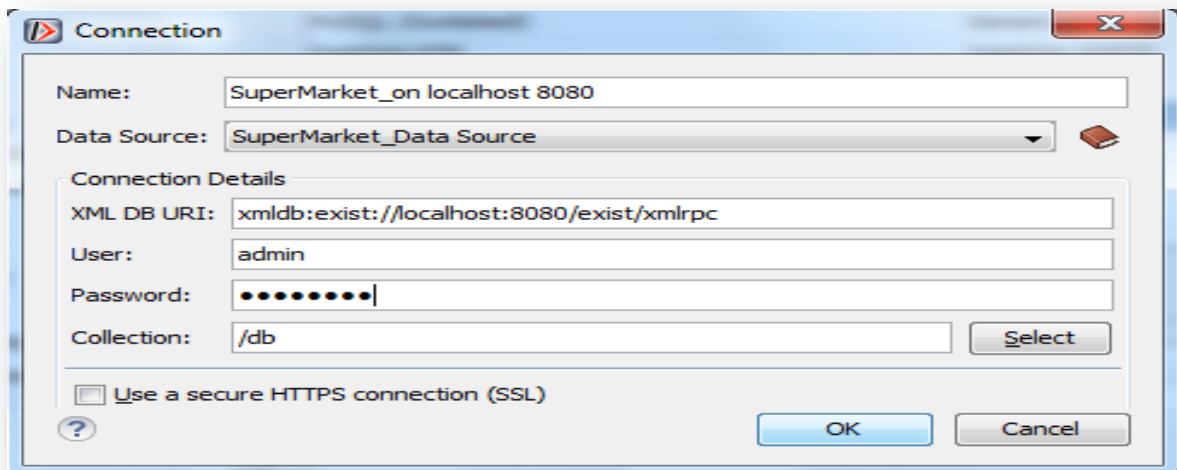


Εικόνα 11.22: Ένωση eXist-db με oXygen, βήμα 3

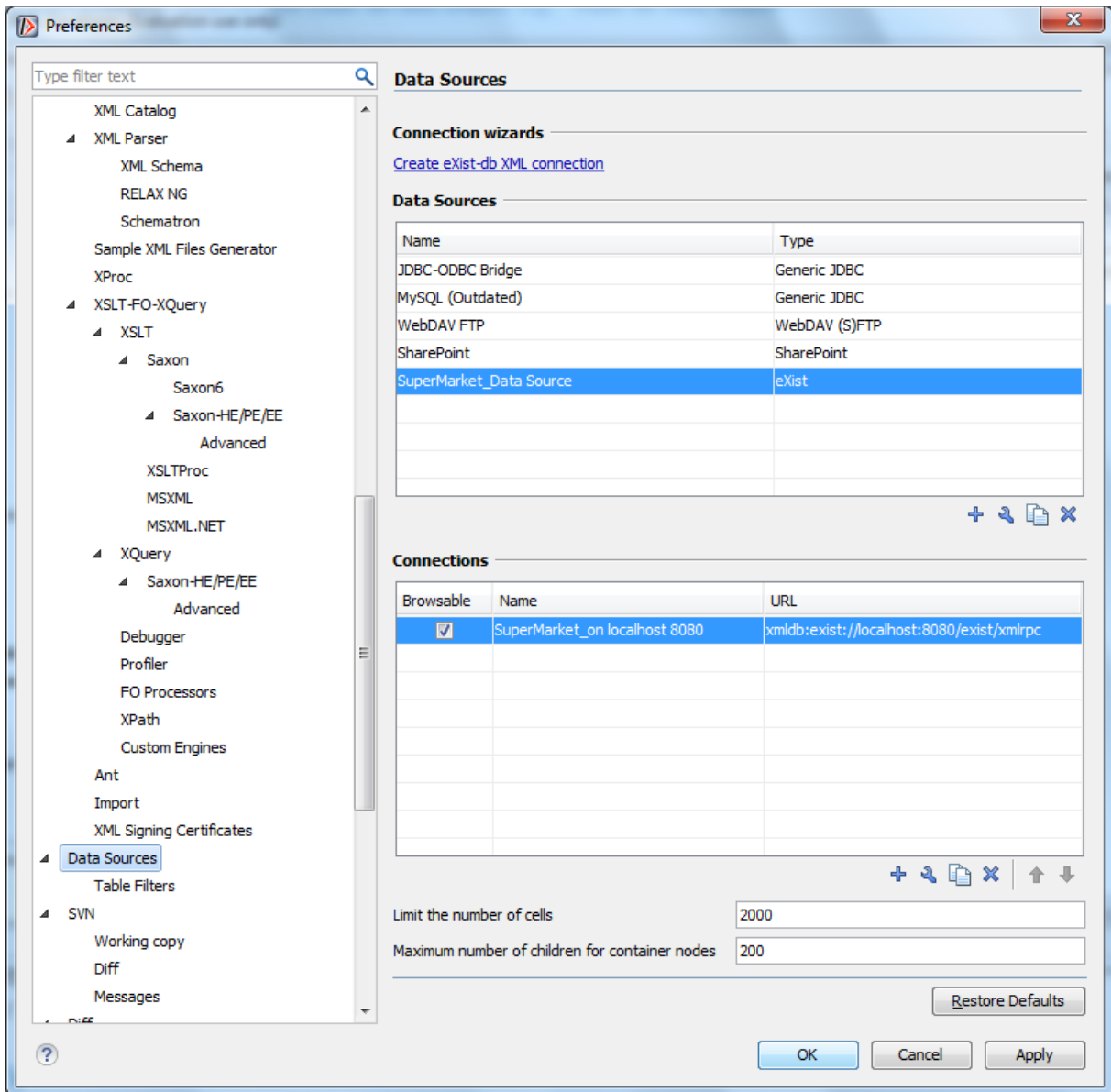




Εικόνα 11.23: Ένωση eXist-db και oXygen, βήμα 4



Εικόνα 11.24: Ένωση eXist-db και oXygen, βήμα 5



Εικόνα 11.25: Ένωση eXist-db και oXygen, βήμα 6

## 12 Παράρτημα Νο 3



Εικόνα 12.1: Παρουσίαση, slide 1



Εικόνα 12.2: Παρουσίαση, slide 2

## Σκοπός εργασίας

Δημιουργία XML βάσης δεδομένων για αλυσίδα Supermarket που θα αποθηκεύει:

- Τις αποθήκες που έχει το Supermarket
- Τα υποκαταστήματά του
- Τους υπαλλήλους που εργάζονται σε αυτό
- Τα στοιχεία κάθε προϊόντος που πουλάει
- Τους προμηθευτές που συνεργάζονται
- Τις παραγγελίες αποθηκών και υποκαταστημάτων

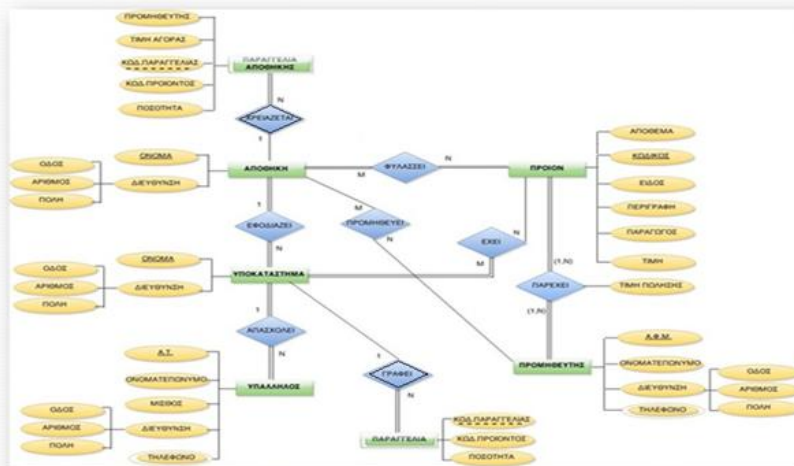
Υλοποίηση διαδικασιών που θα υποστηρίζονται:

- Προσθήκη για καθένα από τα στοιχεία
- Διαγραφή για καθένα από τα στοιχεία
- Ανανέωση του αποθέματος των προϊόντων σε υποκατάστημα και αποθήκη
- Δημιουργία αυτόματης παραγγελίας σε υποκατάστημα και αποθήκη σε περίπτωση μείωσης του αποθέματος κάτω από το αντίστοιχο ελάχιστο

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

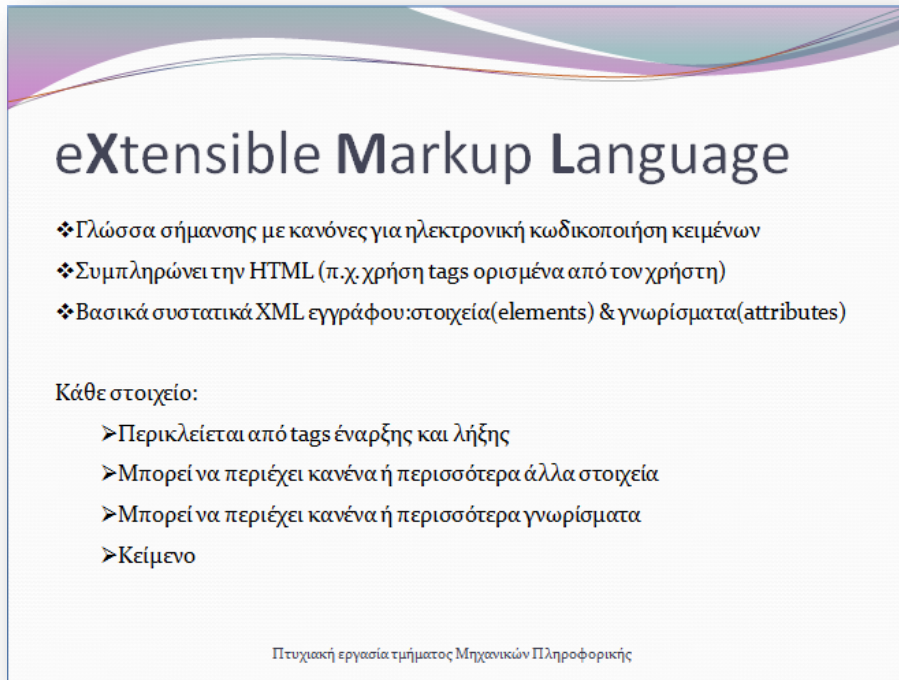
Εικόνα 12.3: Παρουσίαση, slide 3

## Διάγραμμα οντοτήτων-σχέσεων



Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.4: Παρουσίαση, slide 4



**eXtensible Markup Language**

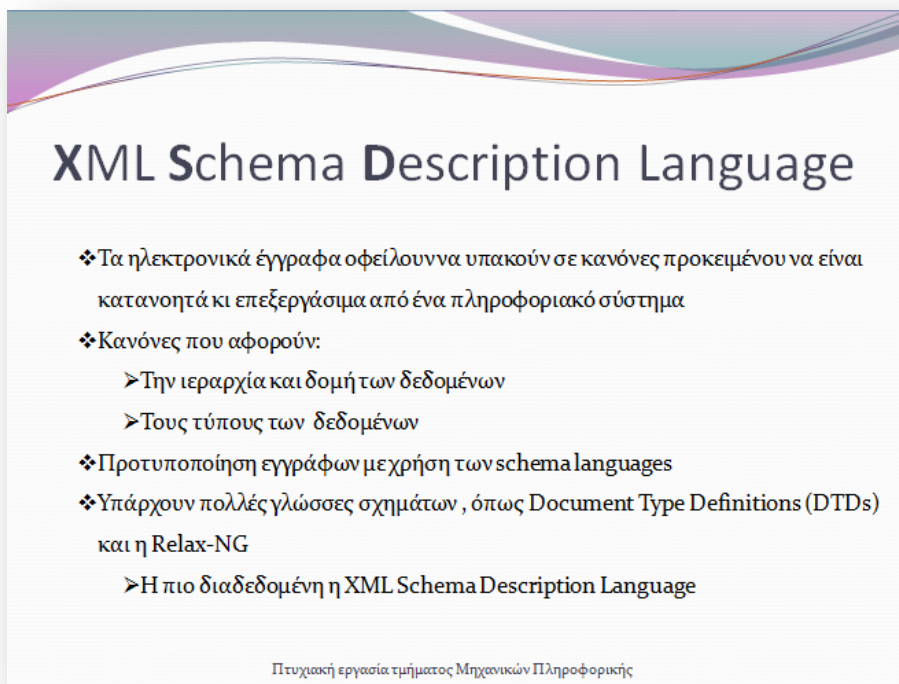
- ❖ Γλώσσα σήμανσης με κανόνες για ηλεκτρονική κωδικοποίηση κειμένων
- ❖ Συμπληρώνει την HTML (π.χ. χρήση tags ορισμένα από τον χρήστη)
- ❖ Βασικά συστατικά XML εγγράφου: στοιχεία (elements) & γνωρίσματα (attributes)

Κάθε στοιχείο:

- Περικλείεται από tags έναρξης και λήξης
- Μπορεί να περιέχει κανένα ή περισσότερα άλλα στοιχεία
- Μπορεί να περιέχει κανένα ή περισσότερα γνωρίσματα
- Κείμενο

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.5: Παρουσίαση, slide 5



**XML Schema Description Language**

- ❖ Τα ηλεκτρονικά έγγραφα οφείλουν να υπακούν σε κανόνες προκειμένου να είναι κατανοητά κι επεξεργάσιμα από ένα πληροφοριακό σύστημα
- ❖ Κανόνες που αφορούν:
  - Την ιεραρχία και δομή των δεδομένων
  - Τους τύπους των δεδομένων
- ❖ Προτυποποίηση εγγράφων με χρήση των schema languages
- ❖ Υπάρχουν πολλές γλώσσες σχημάτων, όπως Document Type Definitions (DTDs) και η Relax-NG
  - Η πιο διαδεδομένη η XML Schema Description Language

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.6: Παρουσίαση, slide 6

## XML Schema(παράδειγμα 1)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="SuperMarket">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="apothiki">
            <xs:element name="proionta" type="proionPerigraph"
              minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="promhtheyths" type="promhtheythsPerigraph"
              minOccurs="1" maxOccurs="unbounded"/>
          </xs:choice>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.7: Παρουσίαση, slide 7

## XML Schema(παράδειγμα 2)

```
<xs:element name="apothiki">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="dieythinsi"/>
      <xs:element name="ypokatasthma" maxOccurs="unbounded">
      <xs:element name="proiontaApothema" type="proiontaLista" minOccurs="1"
        maxOccurs="unbounded"/>
      <xs:element name="promhtheythsApothikis" type="promhtheytesLista"
        minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="paraggeliaApothikis"
        minOccurs="0" maxOccurs="unbounded">
    </xs:sequence>
    <xs:attribute name="onoma" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:key name="key_apothiki">
    <xs:selector xpath="apothiki"/>
    <xs:field xpath="@onoma"/>
  </xs:key>
</xs:element>
```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.8: Παρουσίαση, slide 8

## XML Schema(παράδειγμα 3)

```

<xs:complexType name="proiontaLista">
  <xs:sequence>
    <xs:element name="proionIDREF" type="xs:IDREF"/>
    <xs:element name="apothema" type="xs:float"/>
    <xs:element name="minApothema" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="proionPerigrafh">
  <xs:sequence>
    <xs:element name="proionKwdikos" type="xs:ID"/>
    <xs:element name="perigrafh" type="xs:string"/>
    <xs:element name="eidos" type="xs:string"/>
    <xs:element name="paragwgos" type="xs:string"/>
    <xs:element name="timh" type="xs:float"/>
    <xs:element name="promhtheyths" type="promhtheytesLista"
      minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.9: Παρουσίαση, slide 9

## XML Schema(παράδειγμα 4)

```

<xs:element name="thleftwmo">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="eidosThleftwnou" type="phoneTypes"/>
      <xs:element name="number" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:simpleType name="phoneTypes">
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="κινητό"/>
    <xs:enumeration value="σταθερό"/>
    <xs:enumeration value="FAX"/>
  </xs:restriction>
</xs:simpleType>

```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.10: Παρουσίαση, slide 10

## XQuery

- ❖ Προϋπόθεση η σύνδεση του XML Schema με την eXist-db
- ❖ Επέκταση των XPath's
- ❖ XPath's: εντοπισμός ενός δομικού στοιχείου του XML εγγράφου, με βάση κάποιο κριτήριο, όπως τη θέση του σε αυτό
  - Καταλήγουμε στο δομικό στοιχείο ακολουθώντας ένα μονοπάτι (location path), ένα σύνολο άλλων δομικών στοιχείων
- ❖ Δυνατότητα χρήσης μαθηματικών εκφράσεων
- ❖ Υποστήριξη και σχεδίαση συναρτήσεων

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.11: Παρουσίαση, slide 11

## XQuery προσθήκης

```

declare function local:createParaggelia ($apothiki as xs:string, $ypokatasthma as xs:string,
    $kwdikos as xs:string, $posothta as xs:float)
as element (paraggelia) {
    let $max := doc('SuperMarket.xml')/SuperMarket/apothiki[@onoma=$apothiki]
    /ypokatasthma[@onoma=$ypokatasthma]/paraggelia[@kwdikosParaggelias = max
    (/SuperMarket/apothiki[@onoma=$apothiki]/ypokatasthma[@onoma=$ypokatasthma]
    /paraggelia/@kwdikosParaggelias)]/@kwdikosParaggelias

    return (element paraggelia {
        attribute kwdikosParaggelias (($max+1)),
        element kwdikos {$kwdikos},
        element posothta {$posothta}
    }) };

declare function local:insertParaggeliaYpokatasthmatos
($apothiki as xs:string, $ypokatasthma as xs:string,
    $kwdikos as xs:string, $posothta as xs:float ){
    update insert local:createParaggelia($apothiki, $ypokatasthma, $kwdikos, $posothta)
    preceding doc('SuperMarket.xml')//ypokatasthma[@onoma=$ypokatasthma]/paraggelia[1]
};

<return>
{local:insertParaggeliaYpokatasthmatos('Αποθήκη_1', 'Υποκατάστημα_1', 'ID8877', 15.00)}
</return>
    
```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.12: Παρουσίαση, slide 12



## ΧQuery διαγραφής

```
declare function local:deleteYpallhloYpokatasthmatos($sat as xs:string)
{
  for $sat in
  doc("SuperMarket.xml")/SuperMarket/apothiki/ypokatasthma/ypallhlos[@at=$sat]
  return
    update delete $sat
};

local:deleteYpallhloYpokatasthmatos('AA 11111')
```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.13: Παρουσίαση, slide 13

## ΧQuery ανανέωσης

```
declare function local:updateApothemaYpokatasthmatos
($ypokatasthma as xs:string,$proion as xs:string,$newApothema as xs:float){
  update value //ypokatasthma[@onoma=$ypokatasthma]/
  proiontaApothema[proionIDREF=$proion]/apothema with $newApothema
};

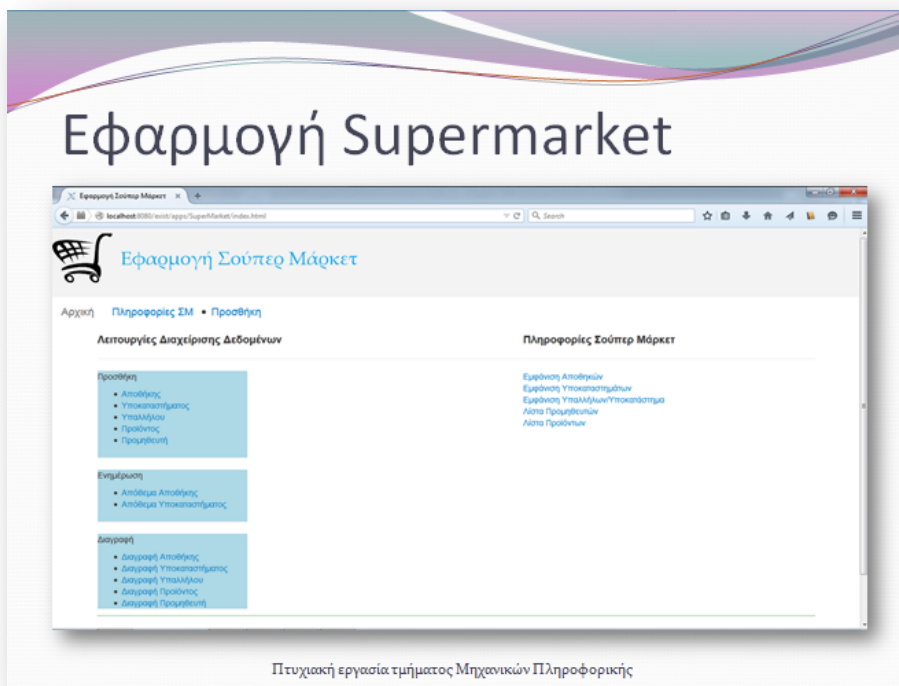
<return>
{local:updateApothemaYpokatasthmatos('Υποκατάστημα_1', 'ID1122', 50.00)}
</return>
```

Πτυχιακή εργασία τμήματος Μηχανικών Πληροφορικής

Εικόνα 12.14: Παρουσίαση, slide 14



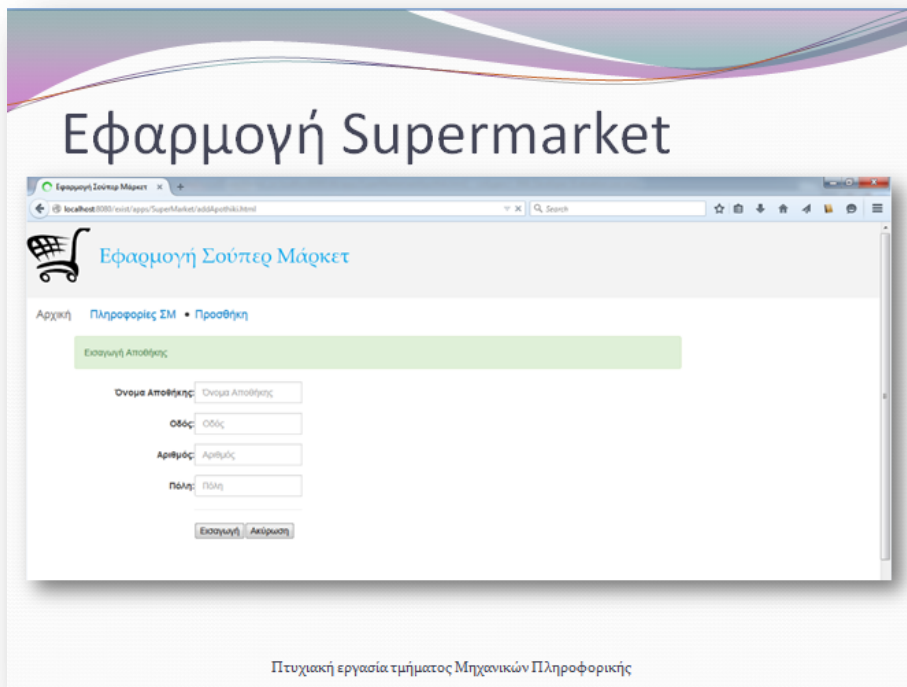
Εικόνα 12.15: Παρουσίαση, slide 15



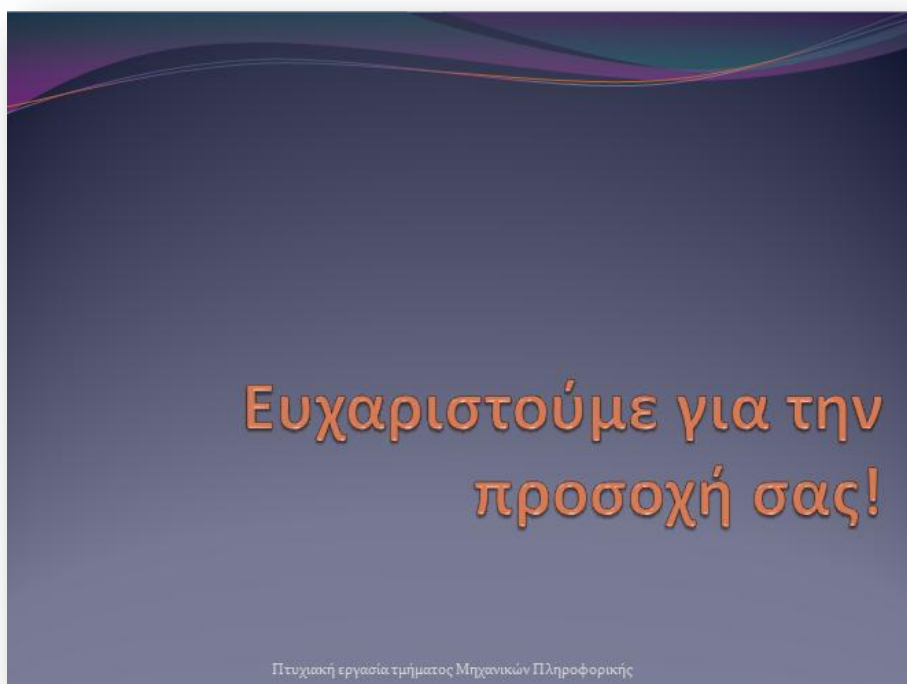
Εικόνα 12.16: Παρουσίαση, slide 16



Εικόνα 12.17: Παρουσίαση, slide 17



Εικόνα 12.18: Παρουσίαση, slide 18



**Εικόνα 12.19: Παρουσίαση, slide 19**