Πανεπιστήμιο Πειραιώς
Τμήμα Ψηφιακών Συστημάτων
Π.Μ.Σ. «Διδακτική της Τεχνολογίας & Ψηφιακά Συστήματα»
Κατεύθυνση Ψηφιακών Επικοινωνιών και Δικτύων

Διπλωματική Εργασία

«Σχεδίαση και Ανάπτυξη Δικτυοκεντρικού Συστήματος Επιχειρηματικής Ευφυΐας»

Νικόλαος Ε. Ψαρράς
[ΑΜ : ΜΕ09099]

Επιβλέπων Καθηγητής: κ. Απόστολος Μηλιώνης

Πειραιάς
Μάρτιος 2013

Περίληψη:

Η παρούσα εργασία αποτελεί μια μελέτη των μεθόδων σχεδίασης και ανάπτυξης ενός περιβάλλοντος επιχειρηματικής ευφυΐας. Αρχικά πραγματοποιείται μια θεωρητική αναφορά στις αρχές, την ορολογία καθώς και την μεθοδολογία που ακολουθείται στη σχεδίαση και την ανάπτυξη τέτοιων συστημάτων. Έπειτα γίνεται ανάλυση και περιγραφή της μεθοδολογίας που χρησιμοποιήθηκε για τη δημιουργία του πρακτικού μέρους της διπλωματικής εργασίας. Το πρακτικό μέρος αποτελείται από μια λύση επιχειρηματικής ευφυΐας που περιλαμβάνει τη σχεδίαση και την υλοποίηση του Data Warehouse, των Analysis Κύβων και ενός Web Portal για την προσπέλαση των πληροφοριών. Τέλος, αναφέρονται χρήσεις και προτάσεις που σχετίζονται με συστήματα Επιχειρηματικής Ευφυΐας.

Θεματική Περιοχή: Συστήματα Επιχειρηματικής Ευφυΐας

Λέξεις Κλειδιά: Επιχειρηματική Ευφυΐα, Business Intelligence, BI, Data Warehouse, Κύβοι, Business Analytics, Decision Making.

Στους γονείς μου Ευάγγελο και Βασιλική.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Απόστολο Μηλιώνη για την αμέριστη υποστήριξη και την εξαιρετική συνεργασία που είχαμε στα πλαίσια της παρούσας διπλωματικής εργασίας. Επιπλέον θέλω να ευχαριστήσω τη γυναίκα μου Γεωργία για την παρότρυνση και την υποστήριξή της κατά τη διάρκεια των μεταπτυχιακών μου σπουδών. Τέλος οφείλω να ευχαριστήσω του γονείς μου, Ευάγγελο και Βασιλική, για την πίστη τους στις δυνάμεις μου, την υποστήριξή τους σε όλη την πορεία των σπουδών μου καθώς και για την επιλογή τους να επενδύσουν στη μόρφωσή μου.

Πίνακας Περιεχομένων

1. Εισαγωγή

1.1. Γενική Επισκόπηση

Οι επιχειρήσεις, οι εταιρίες και οι οργανισμοί γενικότερα προσπαθούν να διατηρήσουν και να διαχειριστούν όσο το δυνατόν περισσότερα δεδομένα και πληροφορίες έτσι ώστε αναλύοντας τα να είναι σε θέση να αξιολογήσουν την πορεία τους, να εντοπίσουν προβλήματα και δυσαρμονίες ώστε να οδηγηθούν στη λήψη αποφάσεων, λύσεων αλλά και προβλέψεων για το παρόν και το μέλλον. Τα στελέχη διαφόρων βαθμίδων μιας επιχείρησης καλούνται συνεχώς να λαμβάνουν αποφάσεις οι οποίες έμμεσα ή άμεσα βασίζονται σε ανάλυση συσχετιζόμενων στοιχείων και δεδομένων. Παλαιότερα, όταν η τεχνολογία στις επιχειρήσεις δεν διαδραμάτιζε τον σημερινό νευραλγικό ρόλο, η λήψη αποφάσεων, κυρίως στις υψηλές βαθμίδες της διοίκησης, γίνονταν πολλές φορές με γνώμονα τη διαίσθηση, χωρίς να στηρίζεται σε στοιχεία και δείκτες, γεγονός που αύξανε το ρίσκο λήψης αποφάσεων με αρνητικό αντίκτυπο στην εταιρία. Όμως ο ολοένα και υψηλότερος ανταγωνισμός καθώς και η πληθώρα των πληροφοριών που σχετίζονται με μια εταιρία, κάνουν επιτακτική την ανάγκη συλλογής και ανάλυσης δεδομένων προκειμένου να επιτευχθεί πλήρης έλεγχος αλλά και αποτελεσματική λήψη αποφάσεων με όσο το δυνατόν μειωμένη πιθανότητα αποτυχίας και λάθους.

Στην εποχή μας η τεχνολογία παίζει πρωταρχικό ρόλο στην διαχείριση των εταιρικών δεδομένων μιας και η πλειοψηφία των

επιχειρήσεων διατηρούν τις πληροφορίες σε ψηφιακή μορφή και την επεξεργάζονται/αναλύουν με προγράμματα λογισμικού. Ο όγκος των δεδομένων συνεχώς αυξάνεται, επομένως θα πρέπει να υπάρχουν μηχανισμοί οι οποίοι θα συλλέγουν, θα αναλύουν και θα εξάγουν την κατάλληλη πληροφορία στην τέτοια μορφή που θα διευκολύνει τους υπευθύνους λήψεων αποφάσεων (Decision Makers) στην εκτέλεση του έργου τους. Άλλος ένας σημαντικός παράγοντας είναι η ταχύτητα με την οποία μπορεί να είναι διαθέσιμη η πληροφορία καθώς και η ευκολία άντλησης της.

Η μορφή των καταγεγραμμένων αρχείων εξαρτάται από τη χρονική περίοδο στην οποία δραστηριοποιείται η εταιρία, η υπάρχουσα τεχνολογία καθώς και το μέγεθος της επιχείρησης. Κάποια παραδείγματα είναι η χειρόγραφη καταγραφή σε βιβλία, σε αρχεία κειμένου ψηφιακής μορφής, σε βάσεις δεδομένων κτλ. Μια εταιρία μπορεί χρησιμοποιεί έναν ή περισσότερους τρόπους αποθήκευσης των δεδομένων της ή να διατηρεί πολλαπλά αρχεία καταγραφής όπως για παράδειγμα αρκετές βάσεις δεδομένων, υπολογιστικά φύλλα κτλ. Άλλη μια ανάγκη λοιπόν είναι η συλλογή και η επεξεργασία δεδομένων από πολλαπλές πηγές έτσι ώστε η εικόνα να είναι πληρέστερη και πιο κοντά στην πραγματικότητα.

Μια λύση σε τέτοιου είδους προβλήματα και απαιτήσεις είναι ένα Σύστημα Επιχειρηματικής Ευφυΐας (Business Intelligence System). Με την Επιχειρηματική Ευφυΐα μπορούμε μέσω των ηλεκτρονικών υπολογιστών και της σύγχρονης τεχνολογίας να αντλήσουμε δεδομένα από πολλαπλές

πηγές, να τα μετασχηματίσουμε ώστε να έχουν κατάλληλη μορφή, να τα επεξεργαστούμε και να τα παρέχουμε για ανάλυση.

Αναζητώντας την προέλευση των δυο όρων του Business Intelligence, σύμφωνα με το (1) ο όρος "Intelligence" προέρχεται από την στρατιωτική επιστήμη. Οι αρχαίοι ρωμαίοι όριζαν ως Strategic Intelligence την ανάλυση οποιουδήποτε στοιχείου και γεγονότος που ελάμβανε χώρα πριν τη μάχη περιλαμβάνοντας ακόμα και μακροπρόθεσμες πληροφορίες που θα μπορούσαν να επηρεάσουν την έκβαση της μάχης. Επιπλέον Tactical Intelligence ήταν η περισυλλογή πρόσφατου υλικού που θα μπορούσε να καθόριζε την έκβαση της μάχης. Στη σύγχρονη εποχή ο όρος Intelligence ορίζεται ως η επίσημη μυστική συλλογή και επεξεργασία πληροφοριών άλλων χωρών με σκοπό τη διαμόρφωση και εφαρμογή της εξωτερικής πολικής καθώς και τη διαμόρφωση των μυστικών δραστηριοτήτων στο εξωτερικό ώστε να διευκολυνθεί η εφαρμογή της εξωτερικής πολιτικής. Ερευνητικά, στον τομέα των πληροφοριακών συστημάτων ο όρος Business Intelligence εμφανίστηκε για πρώτη φορά στην σεμιναριακή εργασία του Luhn (2) που ορίζει την έννοια του "Business" ως «μια συλλογή από δραστηριότητες οι οποίες διεξάγονται για οποιονδήποτε σκοπό, είτε πρόκειται για την επιστήμη, την τεχνολογία, το εμπόριο, τη βιομηχανία, το δίκαιο, την κυβέρνηση, την άμυνα, κ.λπ. »

1.2. Περιγραφή

Στα πλαίσια αυτής της μεταπτυχιακής διπλωματικής εργασίας θα αναλυθούν θεωρητικά τα συστήματα Επιχειρηματικής Ευφυΐας κυρίως από τεχνολογικής πλευράς και θα παρουσιαστεί ένα Custom Project το οποίο υλοποιεί ένα τέτοιο σύστημα. Για την υλοποίηση του Project χρησιμοποιείται ως πηγή δεδομένων η παραγωγική βάση δεδομένων Adventure Works που παρέχει η Microsoft για ακαδημαϊκούς σκοπούς. Η εταιρία Adventure Works εμπορεύεται ποδήλατα και εξοπλισμό για ποδηλάτες. Υποθέτουμε ότι έχει ζητήσει την δημιουργία ενός συστήματος Επιχειρηματικής Ευφυΐας παρέχοντας μας την παραγωγική βάση της από την οποία θα αντλήσουμε τα δεδομένα. Στα πλαίσια της εργασίας ακολουθούνται και περιγράφονται όλα τα βήματα που χρειάζονται για τη δημιουργία του συστήματος (ανάλυση απαιτήσεων, προδιαγραφές, σχεδιασμός Data Warehouse, δημιουργία κύβων και Business Intelligence Web Site σε τεχνολογία ASP.NET).

Σκοπός της διπλωματικής εργασίας είναι να γίνει μια εκτενής θεωρητική αναφορά στο τι είναι Επιχειρηματική Ευφυΐα, ποια τα πλεονεκτήματα και τα μειονεκτήματά της, οι τρόποι και οι κανόνες με τους οποίους δημιουργείται ένα ολοκληρωμένο σύστημα Business Intelligence. Επιπρόσθετα, ένας στόχος είναι να οδηγηθούμε από τη θεωρία στην πράξη με την αναλυτική περιγραφή των διαδικασιών και των βημάτων που χρησιμοποιήθηκαν για τη δημιουργία ενός συστήματος Επιχειρηματικής Ευφυΐας.

1.3. Περίληψη Κεφαλαίων

Στο κεφάλαιο 1 γίνεται μια εισαγωγή στα συστήματα Επιχειρηματικής Ευφυΐας και μια περιγραφή του προβλήματος και των περιορισμών. Στο 2ο κεφάλαιο αναλύεται το θεωρητικό υπόβαθρο που σχετίζεται με τον όρο Business Intelligence καθώς και το Data Warehouse. Στο κεφάλαιο 3 παρουσιάζεται το πρακτικό μέρος της διπλωματικής εργασίας και ουσιαστικά ο τρόπος υλοποίησης του Data Warehouse, των Analysis Services κύβων καθώς και του Web Site που έχει πρόσβαση στα δεδομένα του Data Warehouse. Τέλος, στο κεφάλαιο 4 παρουσιάζονται συμπεράσματα, προτάσεις και βελτιώσεις που σχετίζονται με τη θεματική ενότητα της διπλωματικής εργασίας.

2. Επιχειρηματική Ευφυΐα (Business Intelligence)

2.1. Συστήματα Επιχειρηματικής Ευφυΐας

2.1.1. Λήψη Αποφάσεων και BI Συστήματα

Λήψη Αποφάσεων

Οι αποφάσεις οι οποίες λαμβάνονται σε μια εταιρία καθορίζουν την κατεύθυνση και γενικότερα το μέλλον της, μιας και η λήψη αυτών των αποφάσεων μπορούν να την οδηγήσουν είτε στην ανάπτυξη είτε σε δύσκολες καταστάσεις (σε διάφορους τομείς). Σύμφωνα με το (3) μπορούμε να διαχωρίσουμε τις αποφάσεις σε αυτές οι οποίες λαμβάνονται από τα υψηλότερα κλιμάκια της επιχείρησης, σε αυτές που λαμβάνονται από τα μεσαία και αυτές που λαμβάνονται από χαμηλότερα κλιμάκια. Συνήθως οι πρώτες είναι γενικότερες και ορίζουν ένα πλαίσιο για το παρόν και το μέλλον βάσει του οποίου κινούνται οι αποφάσεις των χαμηλότερων κλιμακίων οι οποίες είναι πιο περιορισμένου εύρους και εισέρχονται σε πιο λεπτομερείς αναλύσεις.

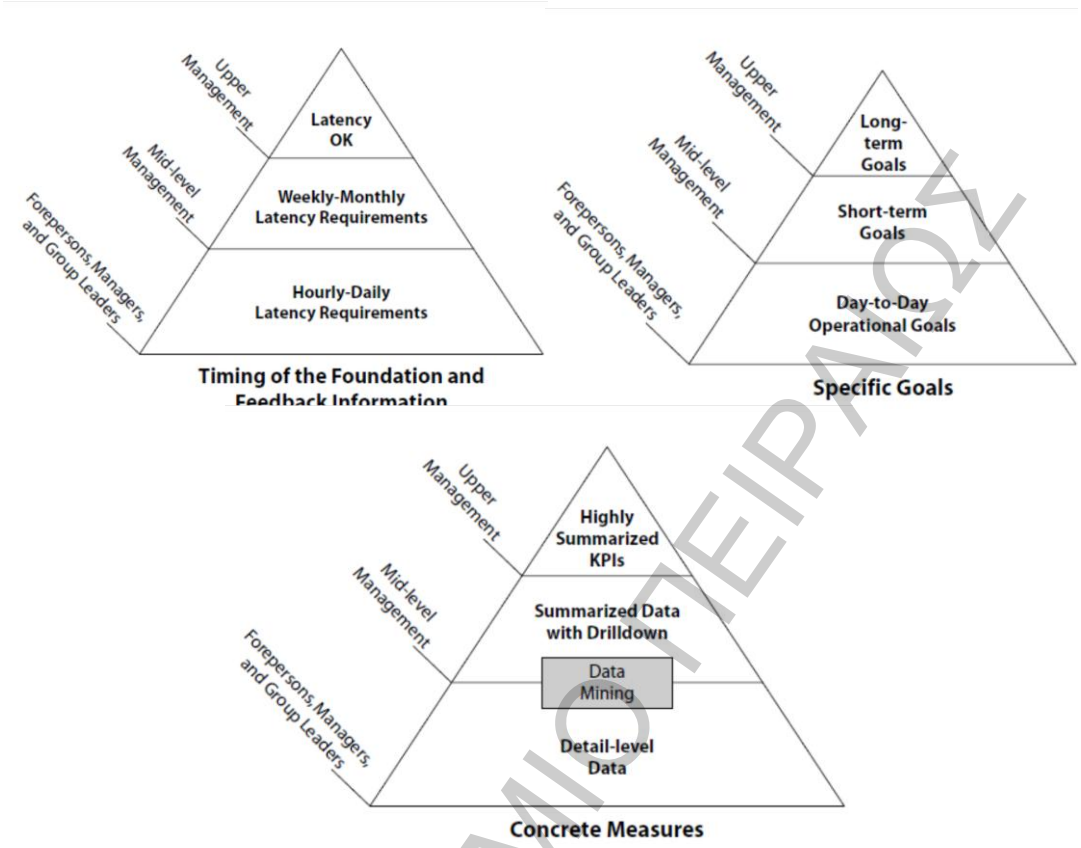
Οι υπεύθυνοι λήψης αποφάσεων που βρίσκονται στο υψηλό κλιμάκιο χρειάζονται συνοπτικές και περιληπτικές πληροφορίες, που περιγράφουν πιο συνοπτικά τα δεδομένα της εταιρίας χωρίς μεγάλες λεπτομέρειες (πχ τζίρος, συνολικό κόστος κ.α.). Αυτή η ομάδα περισσότερο ενδιαφέρεται για το αν οι τιμές αυτές βρίσκονται μεταξύ κάποιων ορίων, παρά για το ποια δεδομένα διαμόρφωσαν αυτές τις τιμές. Επιπλέον αναλύουν τις τάσεις αυτών των τιμών έτσι ώστε να μπορέσουν να οδηγηθούν σε συμπεράσματα και αποφάσεις για το παρόν αλλά και για το μέλλον της επιχείρησης. Σε αυτό το επίπεδο μπορεί να γίνει

ανεκτή κάποια σχετική καθυστέρηση ενημέρωσης των δεδομένων, μιας και είναι πιο συγκεντρωτικά και σχετίζονται με αποφάσεις μακροπρόθεσμες. (3)

Στο μεσαίο κλιμάκιο των υπευθύνων λήψεων αποφάσεων οι αποφάσεις σχετίζονται με τους διάφορους τομείς-διευθύνσεις της επιχείρησης και οι πληροφορίες που χρειάζονται για την λήψη αυτών των αποφάσεων είναι μεν συνοπτικές αλλά ίσως χρειάζεται να υπάρχει μια εμβάθυνση σε πιο λεπτομερή δεδομένα χαμηλότερης βαθμίδας τα οποία οδήγησαν σε αυτές τις τιμές. Η καθυστέρηση ανανέωσης των δεδομένων μπορεί να κυμαίνεται από μια εβδομάδα έως ένα μήνα, το ίδιο και οι στόχοι οι οποίοι τίθενται. (3)

Οι υπεύθυνοι λήψεων αποφάσεων του **χαμηλότερου κλιμακίου** βασίζονται στη λεπτομέρεια, επομένως χρειάζονται δεδομένα τα οποία αλλάζουν μέρα με την ημέρα και επομένως η καθυστέρηση ανανέωσης κυμαίνεται σε αυτά τα επίπεδα (ημερήσια). Οι στόχοι είναι βραχυπρόθεσμοι, ο όγκος των δεδομένων είναι σχετικά μεγάλος και οι υπεύθυνοι πρέπει να μπορούν να έχουν πρόσβαση σε χαμηλού επιπέδου πληροφορίες σχετικά με το τμήμα της επιχείρησης για το οποίο είναι υπεύθυνοι. (3)

Αυτά τα τρία κλιμάκια των υπευθύνων λήψεων αποφάσεων καθώς και τα χαρακτηριστικά συνοψίζονται στην Εικόνα 2.1.



Εικόνα 2.1: Κλιμάκια Υπευθύνων Αποφάσεων Και Χαρακτηριστικά (3)

Το κοινό όλων των αποφάσεων που πρέπει να ληφθούν σε μια εταιρία είναι ότι πρέπει να κινούνται βάσει κάποιων στόχων και επιπλέον να αξιολογούνται βάσει του κατά πόσο τα αποτελέσματά τους πλησιάζουν ή όχι αυτούς τους στόχους (3). Τις περισσότερες φορές, για να μπορεί η κάθε απόφαση να οδηγήσει σε επιθυμητά αποτελέσματα βάσει των στόχων, θα πρέπει να στηρίζεται σε μελέτη και ανάλυση δεδομένων. Επιπλέον, μεγάλος αριθμός των στόχων θα πρέπει να διαμορφώνονται με βάσει τα στοιχεία που προκύπτουν από την ανάλυση δεδομένων. Τα δεδομένα διαχωρίζονται σε αυτά που μέσω της ανάλυσής τους οδηγούν στην λήψη της απόφασης και σε αυτά τα οποία μας βοηθούν στην αξιολόγηση της ληφθείσας απόφασης.

Εφόσον έχουν τεθεί οι στόχοι, μια επιτακτική ανάγκη είναι οι υπεύθυνοι λήψεων αποφάσεων (decision makers) να εφοδιαστούν με τις κατάλληλες πληροφορίες στον κατάλληλο χρόνο και σε τέτοια μορφή ώστε να μπορέσουν να λάβουν γρήγορα τις βέλτιστες αποφάσεις.. Επιπλέον πρέπει να υπάρχει η δυνατότητα εμφάνισης των δεδομένων σε διάφορες μορφές (πχ διαγράμματα, πίνακες, γραφήματα κτλ) έτσι ώστε με την οπτικοποίηση τους να οδηγούμαστε σε γρήγορη και ευκολότερη κατανόηση και ανάλυση. Αυτά μπορεί να τα αντλήσει κάποιος μέσω της επιχειρηματικής ευφυΐας. «***Επιχειρηματική ευφυΐα*** *είναι η παροχή χρήσιμης και κατάλληλης πληροφορίας στους υπεύθυνους λήψεων αποφάσεων στο κατάλληλο χρονικό πλαίσιο αλλά και με την κατάλληλη μορφή ώστε να τους βοηθήσει να οδηγηθούν με ασφάλεια στη λήψη ορθών και τεκμηριωμένων αποφάσεων* ». (3)

Υπάρχουν πολλοί ορισμοί για την επιχειρηματική ευφυΐα λόγω του ότι διαφέρει κάθε φορά η επιστημονική οπτική γωνία από την οποία ασχολούμαστε με αυτή. Οι Arnott & Pervan (4) αναφέρουν πως «*ο όρος Business Intelligence είναι ένας επαρκώς καθορισμένος όρος και η βιομηχανική του προέλευση υποδεικνύει ότι οι διάφοροι κατασκευαστές λογισμικού και οι συμβουλευτικές εταιρίες του προσδίδουν ορισμούς που να ταιριάζουν με το προϊόν τους: κάποιοι χρησιμοποιούν τον όρο BI για ολόκληρο το εύρος των προσεγγίσεων λήψης αποφάσεων*». Τα επιστημονικά και επαγγελματικά literature reviews έχουν ένα μεγάλο πλήθος ορισμών:

- Το BI είναι η διαδικασία ανάκτησης και ανάλυσης εσωτερικής και εξωτερικής επιχειρηματικής πληροφορίας. (5)

- Το BI δεν είναι ούτε προϊόν ούτε σύστημα. Είναι μια αρχιτεκτονική και μια συλλογή ενσωματωμένων λειτουργικών εφαρμογών λήψης αποφάσεων και βάσεων δεδομένων τα οποία παρέχουν στην επιχειρηματική κοινότητα ευκολία πρόσβασης στα εταιρικά δεδομένα. (6)
- Το BI είναι ένας γενικευμένος όρος για εφαρμογές, πλατφόρμες, εργαλεία και τεχνολογίες που υποστηρίζουν τις διαδικασίες εξερεύνησης εταιρικών δεδομένων, συσχετισμών δεδομένων και τάσεων. Το BI παρέχει στους ανώτερους υπαλλήλους έγκαιρες και ακριβείς πληροφορίες ώστε να κατανοήσουν καλύτερα την επιχείρησή τους και να λάβουν πιο συνειδητές και real-time αποφάσεις. (7)
- Είναι μια οργανωμένη και συστηματική διαδικασία μέσω της οποίας οι οργανισμοί αποκτούν, αναλύουν και διαδίδουν τις πληροφορίες που λαμβάνουν από εσωτερικές και εξωτερικές πηγές πληροφοριών και οι οποίες είναι σημαντικές για τις επιχειρηματικές τους δραστηριότητες και για την λήψη αποφάσεων (8)
- Το BI είναι ένα σύνολο επιχειρηματικών πληροφοριών και αναλύσεων στο πλαίσιο των σημαντικών επιχειρηματικών διαδικασιών οι οποίες οδηγούν σε αποφάσεις και ενέργειες. Ειδικότερα, BI σημαίνει μόχλευση πληροφοριών με σημαντικές εταιρικές διαδικασίας ώστε να επιτευχθεί βελτίωση της επιχειρηματικής απόδοσης. (9)

Το BI σύστημα έρχεται για να γεφυρώσει την απόσταση μεταξύ του μεγάλου όγκου των δεδομένων που περιέχεται στις παραγωγικές βάσεις δεδομένων μιας εταιρίας και της πληροφορίας που εξάγεται από αυτά τα δεδομένα και είναι διαθέσιμη στους υπεύθυνους λήψεως αποφάσεων και το περιβάλλον τους. Το πιο σημαντικό στοιχείο ενός BI συστήματος είναι ο ανθρώπινος παράγοντας αφού ο άνθρωπος είναι αυτός που θα μπορέσει να καθοδηγήσει το σύστημα στην μετατροπή των δεδομένων σε χρήσιμη πληροφορία. Γενικά το μέγεθος των πληροφοριών αυξάνεται με μικρότερους ρυθμούς από τον αριθμό των αποφάσεων που πρέπει να ληφθούν βασιζόμενες σε πληροφορίες. Επομένως, είναι επιτακτικό να έχουμε πρόσβαση σε όσο το δυνατόν περισσότερο όγκο πληροφορίας στο μικρότερο δυνατό χρόνο. (10)

Αυτό το κενό μεταξύ δεδομένων και πληροφορίας σύμφωνα με το (10) προέρχεται από διάφορους λόγους, οι σημαντικότεροι εκ των οποίων είναι:

- Τα δεδομένα που απαιτούνται για ανάλυση βρίσκονται σε διαφορετικές πηγές δεδομένων που μπορεί να έχουν εντελώς διαφορετική δομή και είναι δύσκολο να ενσωματωθούν.
- Η διοίκηση λαμβάνει εκτεταμένες αναφορές που μπορεί να είναι ακατάλληλες ή να χρησιμοποιούνται σπάνια.
- Υπάρχουν αρκετά δεδομένα σε μια εταιρία για το οποία οι υπεύθυνοι λήψεων αποφάσεων αγνοούν την ύπαρξή τους.

- Τα δεδομένα στις λειτουργικές βάσεις δεδομένων δεν είναι κατάλληλα σχεδιασμένα και διαμορφωμένα ώστε να υποστηρίζουν τη λήψη αποφάσεων.
- Για αναλυτές με έλλειψη τεχνικών γνώσεων η προετοιμασία των reports και η εκτέλεση επερωτήσεων στη βάση αποτελεί μια χρονοβόρα και περίπλοκη δραστηριότητα. Τα παραδοσιακά εργαλεία που χρησιμοποιούνται για αυτό το σκοπό είναι δύσκολα στη χρήση (παρά την ύπαρξη γραφικού περιβάλλοντος).
- Λόγω της αυξημένης ζήτησης πληροφορίας στις διαδικασίας ανάλυσης και λήψης αποφάσεων, το δυναμικό του IS παίζει το ρόλο του διαχειριστή δεδομένων: ενσωματώνουν δεδομένα από διαφορετικές πηγές, προετοιμάζουν reports, υπολογίζουν τιμές δεδομένων κτλ.
- Οι αναλυτές σπαταλούν περισσότερο χρόνο να συλλέξουν τις απαραίτητες πληροφορίες από τον χρόνο που χρειάζεται για να τις αναλύσουν.
- Υπάρχει έλλειψη εξωτερικών πληροφοριών που είναι απαραίτητες για τη λήψη αποφάσεων: Οι ιδιοκτήτες των δεδομένων είναι υπερπροστατευτικοί ως προς τις πληροφορίες που παρέχουν και επιπλέον υπάρχουν περιορισμοί και ασυμβατότητες μεταξύ των συστημάτων Software και Hardware.

Οφέλη Και Χαρακτηριστικά των BI Systems

Όσο πιο πολλές είναι οι πληροφορίες οι οποίες πρέπει να διαχειριστούμε τόσο απαραίτητη είναι η χρήση ενός συστήματος επιχειρηματικής ευφυΐας. Οι υπεύθυνοι λήψεων αποφάσεων πρέπει να έχουν πρόσβαση στα επιθυμητά δεδομένα με μια εύκολη και γρήγορη διαδικασία εντοπισμού. Επιπλέον, όταν αυτά τα δεδομένα προέρχονται από διαφορετικές πηγές (πχ Microsoft SQL DB, Microsoft Access DB, Oracle DB, Αρχεία xml, text, Excel κ.α.) πρέπει ένα τέτοιο σύστημα να τα συγκεντρώνει και να τα μετασχηματίζει σε μια κοινή μορφή με μια διαδικασία διάφανη για τον τελικό χρήστη, ο οποίος έχει πρόσβαση στα ομαδοποιημένα δεδομένα χωρίς να τον ενδιαφέρει το αν προήλθαν ή όχι από διαφορετικές πηγές ή αν προέκυψαν από συνδυασμό άλλων δεδομένων.

Τα πιο σημαντικά πλεονεκτήματα της χρήσης BI συστημάτων στην ανάλυση των δεδομένων, σύμφωνα με το (11) είναι τα εξής:

- **Ανάλυση που υποστηρίζει Cross Selling και Up Selling**: Το Cross/Up Selling περιλαμβάνει την πώληση προϊόντων σε συγκεκριμένους πελάτες βασιζόμενη σε προηγούμενες αγορές. Ένα παράδειγμα Cross/Up Selling είναι το Market Basket Analysis βάσει του οποίου προτείνεται σε πελάτες ένα πακέτο συσχετιζόμενων προϊόντων ή δεδομένου ενός προϊόντος προτείνονται συσχετιζόμενα προϊόντα με βάση την ανάλυση των προηγούμενων αγορών του ίδιου του πελάτη ή της πλειοψηφίας των πελατών που έχουν επιλέξει το συγκεκριμένο προϊόν.

- **Κατηγοριοποίηση πελατών και Profiling**: Η ανάλυση μέσω BI βοηθάει στην ομαδοποίηση των πελατών με διάφορα κριτήρια (π.χ. δημογραφικά, συμπεριφοράς, κινήτρων κ.α.). Εφαρμόζεται κυρίως για να ξεχωρίσουν οι «καλοί» από τους «κακούς» πελάτες ώστε να ακολουθηθεί διαφορετική πολιτική στην κάθε ομάδα.
- **Ανάλυση της σπουδαιότητας των παραμέτρων**: Όταν υπάρχουν πολλές παράμετροι που περιγράφουν ένα προϊόν, με την BI ανάλυση αυτές οι παράμετροι μπορούν να βαθμονομηθούν έτσι ώστε να δοθεί το αντίστοιχο βάρος όσον αφορά την προώθηση, την τιμολόγηση κτλ.
- **Ανάλυση του Χρόνου Επιβίωσης**: Ο χρόνος επιβίωσης προσομοιώνει τον χρόνο που ο πελάτης επιθυμεί να εξυπηρετείται από μια εταιρία καθώς και την πιθανότητα να οδηγηθεί σε κάποια ανταγωνιστική εταιρία. Στόχος είναι η αύξηση αυτού του χρόνου.
- **Ανάλυσης της εμπιστοσύνης των πελατών καθώς και της στροφής των πελατών στον ανταγωνισμό**: Η ανακάλυψη των παραμέτρων που μπορεί να οδηγήσουν τον πελάτη σε κάποιο ανταγωνιστή είναι σημαντική καθώς η βελτίωσή τους μπορεί να οδηγήσει στην αύξηση της εμπιστοσύνης του πελάτη και την αποφυγή της απομάκρυνσής του προς τον ανταγωνισμό.
- **Βαθμολόγηση της πιστοληπτικής ικανότητας**: Περιλαμβάνει μεθόδους μέσω των οποίων υπολογίζεται η ικανότητα του χρήστη να καταβάλει ή όχι το αντίτιμο για τις υπηρεσίες/προϊόντα που του παρέχονται. Επιπλέον μπορεί να γίνει μια πρόβλεψη της

πιστοληπτικής ικανότητας του πελάτη και να εφαρμοστεί διαφορετική πολιτική κατά περίπτωση.

- **Ανίχνευση απάτης**: Είναι η διαδικασία εντοπισμού περιπτώσεων πελατών και γενικότερα ανθρώπων που λειτουργούν με αθέμιτους τρόπους και με μέσα που βλάπτουν την επιχείρηση.
- **Βελτιστοποίηση των Logistics**: Η σωστή προετοιμασία και σχεδιασμός της αποστολής/παραλαβής προϊόντων με βάση την ανάλυση των συσχετιζόμενων παραμέτρων σε ένα BI σύστημα μπορεί να οδηγήσει σε μεγάλη εξοικονόμηση χρημάτων και για την εταιρία αλλά και για τους πελάτες.
- **Πρόβλεψη της ανάπτυξης στρατηγικών εταιρικών διαδικασιών**: Η δημιουργία πολύ-επίπεδων προβλέψεων που βασίζεται σε στοιχεία και πληροφορίες μπορεί να οδηγήσει σε βέλτιστες αποφάσεις και ως προς τον χρόνο και ως προς το περιεχόμενο. Επιπλέον μπορεί να προβλέψει και να προλάβει καταστάσεις ζημιογόνες για την εταιρία.
- **Web Mining (Ανάλυση και αξιολόγηση της απόδοσης υπηρεσιών Internet)**: Με την ανάλυση της συμπεριφοράς των χρηστών των υπηρεσιών διαδικτύου μπορούμε να κατηγοριοποιήσουμε τους χρήστες και να προβλέψουμε τις προτιμήσεις τους ώστε να οδηγηθούμε σε εξατομικευμένες λύσεις και πολιτικές.
- **Web Farming (ανάλυση του περιεχομένου του διαδικτύου)**: Η ανάλυση του περιεχομένου του διαδικτύου βοηθά μια εταιρία να αναλύσει τις τάσεις των χρηστών, να συγκεντρώσει

επιχειρηματικές πληροφορίες και ιδέες οι οποίες δείχνουν να κερδίζουν τον εκάστοτε πελάτη και βοηθούν την εταιρία να κινείται στις σύγχρονες απαιτήσεις οι οποίες συνεχώς αλλάζουν.

Διαφορές της BI και της παραδοσιακής Ανάλυση (Στατικά Reports)

Η ανάλυση των δεδομένων με σκοπό την πρόβλεψη των εμπορικών τάσεων των προϊόντων και των υπηρεσιών είναι αναπόσπαστο κομμάτι των επιχειρήσεων. Παραδοσιακά τέτοιες αναλύσεις γίνονται μέσω μηνιαίων και ετήσιων reports είτε από τα οικονομικά και market τμήματα των εταιριών είτε από ανεξάρτητες εταιρίες οι οποίες αναλαμβάνουν την ανάλυση και την εξαγωγή συμπερασμάτων. Ωστόσο, ο ολοένα και αυξανόμενος ανταγωνισμός καθώς και οι ανάγκες και η ανάπτυξη της τεχνολογίας έχουν κάνει τους Decision Makers να θεωρούν ότι δε μπορούν να λάβουν την κατάλληλη πληροφορία από τα scheduled reports με προκαθορισμένα δεδομένα. (12) Στα παραδοσιακά στατικά Reports προκαθορίζονται η επιθυμητή πληροφορία και οι παράμετροι που έχουν τεθεί και έπειτα δημιουργείται ένα Report το οποίο εμφανίζει τα συγκεκριμένα πεδία και κάθε φορά που το εκτελούμε ενημερώνονται οι τιμές των συγκεκριμένων πεδίων στην προκαθορισμένη αυτή μορφή (3). Επομένως οι παράμετροι και τα δεδομένα ορίζονται μια φορά κατά στη δημιουργία του Report και όταν το είδος της πληροφορίας που χρειαζόμαστε αλλάζει και δεν είναι σταθερό, τότε δημιουργούνται τόσα στατικά Line Reports όσες και οι ανάγκες, μια διαδικασία η οποία είναι αρκετά χρονοβόρα. Τα συστήματα Επιχειρηματικής ευφυΐας παρακάμπτουν τους συγκεκριμένους περιορισμούς αφού κατά την διάρκεια του σχεδιασμού του συστήματος ορίζεται ένας μεγάλος αριθμός

επιθυμητών και διαθέσιμων παραμέτρων, μεταβλητών και δεδομένων, τα οποία μπορεί ο χρήστης να χρησιμοποιήσει ad-hoc την ώρα που τα χρειάζεται και να τα οπτικοποιήσει σε διάφορες μορφές (διάγραμμα, pivot Table, εμφάνιση σε χάρτη κτλ). Επιπλέον αν χρειάζεται να έχουμε πρόσβαση σε δεδομένα που σχετίζονται με ιεραρχικές παραμέτρους (πχ χρονική ιεραρχία) τότε τα συστήματα επιχειρηματικής ευφυΐας μας δείχνουν σε πραγματικό χρόνο πως διαμορφώνεται η πληροφορία στα διάφορα επίπεδα της ιεραρχίας αυτής (πχ ανά χρόνο, ανά τετράμηνο, μήνα κτλ).

Για παράδειγμα ας πάρουμε μια λίστα με τις πωλήσεις προϊόντων ενός super market χρονικής διάρκειας μιας εβδομάδας. Ο Decision Maker θέλει να δει από τα Top 20 προϊόντα σε πωλήσεις, ποια από αυτά τα κατασκευάζει το brand του Super Market και ποια είναι προϊόντα άλλων εταιριών. Μια λύση είναι με τη χρήση αριθμομηχανής να βρει όλα τα κοινά προϊόντα, να τα αθροίσει και να οδηγηθεί στα Top 20. Έπειτα να δει ποια και πόσα από αυτά παράγονται από την εταιρία και έτσι να βρει το επιθυμητό ποσοστό. Άλλη μια λύση είναι να χρησιμοποιήσει ηλεκτρονικά συστήματα που θα τον βοηθήσουν να υπολογίσει αυτό το ποσό. Το Business Intelligence σύστημα είναι αυτό που θα αποδώσει αυτή τη πληροφορία στο βέλτιστο χρόνο και στην καλύτερη μορφή.

Χρήση των BI Συστημάτων

Οι εφαρμογές BI μπορούν να χρησιμοποιηθούν από ένα μεγάλο εύρος χρηστών, από τους ειδικούς του Controlling και του Financial Reporting, τους πωλητές καθώς και τα ανώτερα και τα ανώτατα στελέχη

των εταιριών. Είναι οι εφαρμογές που θα συλλέξουν τα δεδομένα από διάφορες πηγές της εταιρίας και σε διάφορες μορφές και θα τα μετατρέψουν σε χρήσιμες πληροφορίες για τους τελικούς χρήστες τους.

Σύμφωνα με το (11) τα είδη των εταιριών που κατά βάση χρησιμοποιούν συστήματα Business Intelligence καθώς και οι βασικότεροι λόγοι που κάνουν χρήσιμη την ανάλυση BI από αυτές συνοψίζονται στον παρακάτω πίνακα:

Ομάδες Εταιριών	Τομείς και Οφέλη Χρήσης BI
Retail Industry (Εταιρίες Λιανικού Εμπορίου)	• **Forecasting**: Γίνεται χρήση των δεδομένων για πρόβλεψη τιμών και ορισμό των απαιτήσεων με μεγαλύτερη ακρίβεια. • **Ordering and Replenishment**: Λήψη πιο γρήγορων αποφάσεων για παραγγελία προϊόντων και τον καθορισμό της βέλτιστης ποσότητας αυτών. • **Marketing**: Παρέχεται ανάλυση των συναλλαγών των πελατών (ποιο προϊόν πουλάει, ποιος το αγοράζει κτλ) • **Merchandising**: Καθορισμός του σωστού merchandise στην αγορά σε οποιαδήποτε χρονική στιγμή, σχεδιασμός του store level, βελτίωση της απογραφής • **Distribution and Logistics**: Βοηθά τα κέντρα διανομής στη σωστή διαχείριση μεγάλου όγκου εμπορευμάτων. Επιπλέον βοηθά στον προγραμματισμό αποστολής εξερχόμενων και εισερχόμενων εμπορευμάτων μειώνοντας το κόστος. • **Transportation Management**: Ανάπτυξη βέλτιστων load consolidation πλάνων και χρονοδιαγραμμάτων.

	• **Inventory Planning**: Βοηθά στον εντοπισμό του σωστού επιπέδου απογραφής, εξασφαλίζοντας ένα ορισμένο βαθμό εξυπηρέτησης.
Insurance (Ασφαλιστικές Εταιρίες)	• **Claims and Premium Analysis**: Η δυνατότητα ανάλυσης λεπτομερών απαιτήσεων και τιμολόγησης ανά προϊόν, πολιτική, αξίωση κτλ • **Customer Analysis**: Ανάλυση των απαιτήσεων του πελάτη και των product usage patterns, ανάπτυξη προγραμμάτων που βασίζονται στα χαρακτηριστικά του πελάτη, ανάλυση ρίσκου και βελτίωση της εξυπηρέτησης πελατών. • **Risk Analysis**: Εντοπισμός τμημάτων της αγοράς με υψηλό ρίσκο ή με προοπτική, συσχετισμός τμημάτων της αγοράς, μείωση της συχνότητας απαιτήσεων.
Banking, finance and securities (Οικονομικές εταιρίες και Τράπεζες)	• **Customer Profitability Analysis**: Προσδιορισμός της συνολικής κερδοφορίας του κάθε πελάτη, προσδιορισμός της βάσης για πωλήσεις υψηλού κέρδους, μεγιστοποίηση των πωλήσεων σε πελάτες υψηλής αξίας, μείωση του κόστους σε πελάτες χαμηλής αξίας, παροχή των μέσων για μεγιστοποίηση κερδών σε νέα προϊόντα και υπηρεσίες. • **Credit Management**: Καθιέρωση patterns για προβλήματα πίστωσης ανά τύπο και είδος πελάτη, προειδοποίηση πελατών για αποφυγή προβλημάτων πιστώσεων, διαχείριση ορίων πίστωσης, αξιολόγηση του δανειακού χαρτοφυλακίου της τράπεζας, μείωση πιστωτικών ζημιών. • **Branch Sales**: Βελτίωση της εξυπηρέτησης πελατών, του Account και Cross Selling, ενίσχυση εμπιστοσύνης των πελατών.
Telecommunications (Εταιρίες Τηλεπικοινωνιών)	• **Customer profiling and Segmentation**: Προσδιορισμός προφίλ προϊόντων και ομάδων πελατών υψηλού κέρδους, δημιουργία λεπτομερών προφίλ πελατών, εξατομίκευση προγραμμάτων χρηστών με μεγάλη χρήση

	υπηρεσιών, καθορισμός αναγκών των πελατών. • **Customer Demand Forecasting**: Πρόβλεψη μελλοντικών αναγκών όσον αφορά τα προϊόντα και τις υπηρεσίες, βελτίωση της ανάλυσης και των μεθόδων διατήρησης πελατών.
Manufacturing Industry (Βιομηχανία)	• **Sales**: Παροχή ανάλυσης στοιχείων συναλλαγών συγκεκριμένων πελατών. • **Forecasting**: Πρόβλεψη της ζήτησης και καθορισμός απαιτήσεων απογραφής • **Ordering and Replenishment**: Παραγγελία βέλτιστης ποσότητας για κάθε είδος. • **Purchasing**: Βοήθεια κέντρων διανομής για τη διαχείριση μεγάλου όγκου προϊόντων. • **Distribution and logistics**: Βοηθά στον προγραμματισμό αποστολής εξερχόμενων και εισερχόμενων εμπορευμάτων μειώνοντας το κόστος. • **Transportation Management**: Ανάπτυξη βέλτιστων load consolidation πλάνων και χρονοδιαγραμμάτων. • **Inventory Planning**: Βοηθά στον εντοπισμό του σωστού επιπέδου απογραφής, εξασφαλίζοντας ένα ορισμένο βαθμό εξυπηρέτησης.

2.1.2. Στοιχεία ενός Συστήματος BI

OLTP Συστήματα

Οι περισσότερες εταιρίες αποθηκεύουν στις βάσεις δεδομένων τους δεδομένα που σχετίζονται με τις συναλλαγές της εταιρίας όπως παραγγελίες, συναλλαγές, υπηρεσίες κ.α. Τα δεδομένα αυτά ονομάζονται Δεδομένα Συναλλαγών (Transactional Data) και υπάρχουν υπολογιστικά

συστήματα καταγραφής των transactional data που ονομάζονται OLTP Συστήματα (Online Transactional Processing). (3)

Τα συστήματα OLTP θα μπορούσαν να αποτελούν την άμεση πηγή δεδομένων των συστημάτων BI, όμως μια σειρά περιορισμών μας απομακρύνουν από την χρήση τους. Αρχικά, τα OLTP έχουν σχεδιαστεί για την αποθήκευση και διαχείριση μεγάλου όγκου δεδομένων, τα οποία θα πρέπει να είναι προσπελάσιμα από τα BI συστήματα γεγονός που θα επιβαρύνει σημαντικά τη λειτουργία των OLTP συστημάτων και θα δημιουργήσει εμπόδια στο βασικό τους ρόλο που είναι η αποθήκευση δεδομένων. (3) Οι παραδοσιακές βάσεις έχουν κανονικοποιημένα δεδομένα ώστε να αποφύγουν τα διπλότυπα και να εξοικονομήσουν χωρητικότητα. Αν τα δεδομένα που χρειάζεται ένα BI σύστημα αντληθούν από μια κανονικοποιημένη βάση δεδομένων, τότε χρειάζονται αρκετά joins πινάκων που περιέχουν την πληροφορία, γεγονός που οδηγεί σε μεγάλη χρονική καθυστέρηση και επιβάρυνση του συστήματος βάσης δεδομένων. Επιπλέον, σύμφωνα με το (3), στα συστήματα BI πολλές φορές χρειαζόμαστε δεδομένα τα οποία ονομάζονται Aggregates και τα οποία δεν είναι άμεσα αποθηκευμένα στις βάσεις δεδομένων, αλλά είναι αποτέλεσμα επεξεργασίας και εφαρμογής μαθηματικών υπολογισμών στα ήδη υπάρχοντα (ένα παράδειγμα aggregate είναι ο αριθμός των εσόδων που προήλθαν από τις πωλήσεις στο Internet το έτος 2011, έχοντας ως δεδομένο τις εγγραφές της κάθε πώλησης στο διάστημα αυτό). Η απευθείας άντληση τέτοιων δεδομένων απ' τα OLTP συστήματα θα είχε σαν αποτέλεσμα την επιβάρυνσή τους αλλά και το κλείδωμα της βάσης κατά τη διάρκεια του υπολογισμού (οι χρήστες των OLTP δε θα

είχαν πρόσβαση σε αυτά όσο θα γίνονταν οι υπολογισμοί). Επιπλέον οι υπολογισμοί θα ήταν σημαντικά αργοί γιατί τα OLTP συστήματα έχουν σχεδιαστεί με έμφαση στην αποθήκευση και όχι στην επεξεργασία των δεδομένων. Επιπλέον, επειδή τα OLTP αποθηκεύουν μεγάλο αριθμό δεδομένων, σε τακτά χρονικά διαστήματα τα παλαιότερα δεδομένα γίνονται Backup, επομένως δε θα μπορούσαμε να είχαμε άμεση πρόσβαση σε αυτά. Τέλος, τα OLTP συστήματα αποτελούνται από υποσυστήματα τα οποία μπορούν να έχουν διαφορετική δομή και τρόπο αναπαράστασης της ίδιας πληροφορίας (πχ excel, Access Db κτλ) ενώ στα συστήματα BI χρειαζόμαστε δεδομένα ενιαίας μορφής και δομής ώστε να έχουμε γρήγορη πρόσβαση σε αυτά.

Οδηγούμαστε λοιπόν στο συμπέρασμα ότι ο τρόπος που έχουν σχεδιαστεί οι παραδοσιακές OLTP βάσεις δεδομένων δε βοηθούν τους αναλυτές και τους Decision Makers οι οποίοι χρειάζονται aggregated και συγκεντρωτικές πληροφορίες, γρήγορες συγκρίσεις στον χώρο και τον χρόνο, και σύνθεση εκατομμυρίων δεδομένων, τάσεων και άλλων περίπλοκων διεργασιών που υποστηρίζουν τις τακτικές και στρατηγικές διεργασίες λήψεων αποφάσεων. (13)

Data Warehouse

Όλοι οι παραπάνω λόγοι μας οδηγούν στην ανάγκη ύπαρξης ενός ενδιάμεσου συστήματος το οποίο θα αντλεί τα επιθυμητά δεδομένα από τις OLTP βάσεις και έπειτα από επεξεργασία θα τα μετατρέπει σε μια μορφή που θα μπορούν να χρησιμοποιηθούν για τις ανάγκες του BI συστήματος. Αυτό το ενδιάμεσο σύστημα είναι μια βάση δεδομένων και

ονομάζεται Αποθήκη Δεδομένων (Data Warehouse) (3). Στο Data warehouse χρησιμοποιείται η λογική των πολλαπλών διαστάσεων (multidimensional) βάσει της οποίας συλλέγονται δεδομένα από πολλαπλές πηγές και δομούνται με τέτοιο τρόπο ώστε να παρέχεται προς ανάλυση μεγάλος όγκος δεδομένων με μεγάλη ταχύτητα. Σύμφωνα με το (3) ένα Data Warehouse μπορεί να διαιρεθεί σε Data Marts, τα οποία μας παρέχουν πληροφορία για ένα συγκεκριμένο πεδίο – τομέα της επιχείρησης (πχ Πωλήσεις, Παραγωγή κτλ). Στη παρούσα εργασία θα αναφερόμαστε στο όρο Data Warehouse γνωρίζοντας ότι οι ίδιες αρχές εφαρμόζονται και στα Data Marts (που αποτελούν μέρος των Data Warehouses).

Σύγκριση OLTP και DATA WAREHOUSE

Το βασικό χαρακτηριστικό των Data Warehouses είναι η ταχύτητα πρόσβασης, επεξεργασίας και ανάκτησης της πληροφορίας. Το Data Warehouse είναι μια σχεσιακή βάση δεδομένων, όπως και τα OLTP συστήματα, με τη διαφορά ότι χρειάζονται πολύ λιγότερα Joins στους πίνακές του για την λήψη των δεδομένων. Όπως προαναφέρθηκε, οι βάσεις δεδομένων των OLTP συστημάτων έχουν κανονικοποιημένα δεδομένα (normalized) ώστε να μην εμφανίζονται διπλότυπα (redundancy) και έτσι εξοικονομείται χώρος (σε βάρος του χρόνου προσπέλασης και ανάκτησης δεδομένων), ενώ στα συστήματα Data Warehouse μπορούμε να έχουμε επαναλαμβανόμενα δεδομένα (De-normalized) δομημένα σε πολλαπλές διαστάσεις, γεγονός που βοηθάει στην ταχύτερη ανάκτηση μεγάλου όγκου δεδομένων (σε βάρος της χωρητικότητας της βάσης δεδομένων). Στα Data Warehouses οι κανόνες

του Normalization αντικαθίστανται από νέους τρόπους οργάνωσης των δεδομένων και έτσι από τους σχεσιακούς πίνακες οδηγούμαστε σε Facts (Γεγονότα) και Dimensions (διαστάσεις). (3)

Data Load

Ένα σύστημα Data Warehouse, σύμφωνα με το (3) αντλεί τα δεδομένα του από τα OLTP συστήματα. Η διαδικασία της άντλησης δεδομένων από τα OLTP συστήματα ονομάζεται Φόρτωση Δεδομένων (Data Load) και είθισται να πραγματοποιείται σε προκαθορισμένες χρονικές στιγμές (πχ κάθε μήνα, ή κάθε βράδυ ή κάθε ώρα κτλ) ανάλογα με τις ανάγκες της εταιρίας. Με αυτό τον τρόπο το Data Warehouse ανεξαρτητοποιείται από το OLTP και το επιβαρύνει μόνο κατά τη διαδικασία του Data Load η οποία πραγματοποιείται σε χρονικές στιγμές που το OLTP βρίσκεται είναι λιγότερο απασχολημένο (πχ βραδινές ώρες, μη εργάσιμες ημέρες κτλ). Ένα μειονέκτημα του Data Warehouse έγκειται στο ότι ενώ στα OLPT συστήματα έχουμε τη δυνατότητα να έχουμε πρόσβαση σε δεδομένα τα οποία έχουν εισαχθεί στο σύστημα μέχρι τη χρονική στιγμή που τα αναζητούμε, στο Data Warehouse μπορούμε να αναλύσουμε δεδομένα μέχρι τη χρονική στιγμή που έγινε η ενημέρωση των δεδομένων από το OLTP σύστημα. Γι 'αυτό το λόγο γίνεται μια ανάλυση των αναγκών της εταιρίας και της φύσης των δεδομένων που πρέπει να αναλυθούν και καθορίζεται η συχνότητα ανανέωσης δεδομένων του Data Warehouse της. (3)

ETL (Extract – Transform – Load)

Ένα Data Warehouse μπορεί να αντλεί τα δεδομένα του από ένα ή περισσότερα OLTP συστήματα. Επιπλέον το κάθε OLTP σύστημα μπορεί να περιέχει ένα είδος πληροφορίας (πχ ημερομηνία) σε διάφορες ετερογενείς μορφές στους πίνακές του, οι οποίες θα πρέπει να μετασχηματισθούν σε μια μορφή που θα γίνεται «αποδεκτή» από το Data Warehouse. Έτσι μπορεί για το ίδιο πεδίο η κάθε βάση του OLTP να έχει διαφορετικό τύπο, (π.χ. να χρησιμοποιείται διαφορετικός τρόπος αναπαράστασης των ημερομηνιών, ή κάποια πεδία που είναι υποχρεωτικά στο Data Warehouse, σε κάποιο OLTP σύστημα να είναι προαιρετικά και να χρειάζεται η συμπλήρωση δεδομένων κ.α.) Επομένως πριν τα δεδομένα εισαχθούν στο Data Warehouse πρέπει να γίνει Καθαρισμός (Cleansing) και επεξεργασία τους ώστε να είναι στην κατάλληλη μορφή και να μπορέσουν να εισαχθούν στο Data Warehouse. Αυτή η διαδικασία ονομάζεται **ETL** (Extract – Transform – Load). (3) Στα πλαίσια αυτής της διαδικασίας αρχικά αποφασίζουμε την τελική μορφή που θα πρέπει να έχουν τα δεδομένα που θα χρησιμοποιήσουμε για ανάλυση. Έπειτα τα συλλέγουμε και (συνήθως) τα αποθηκεύουμε σε μια ενδιάμεση βάση με ένα bulk load ώστε να μην επιβαρύνουμε την παραγωγική βάση κατά τη διάρκεια της επεξεργασίας τους. Επομένως, ουσιαστικά λαμβάνουμε ένα ακριβές αντίγραφο των εγγραφών, των πεδίων και των πινάκων που χρειαζόμαστε από τις παραγωγικές βάσεις δεδομένων, μια διαδικασία που είναι σχετικά γρήγορη και δεν τις επιβαρύνει. Το επόμενο στάδιο περιλαμβάνει τον «καθαρισμό» και τον μετασχηματισμό των δεδομένων. Σε αυτό το στάδιο αλλάζουμε το μορφή

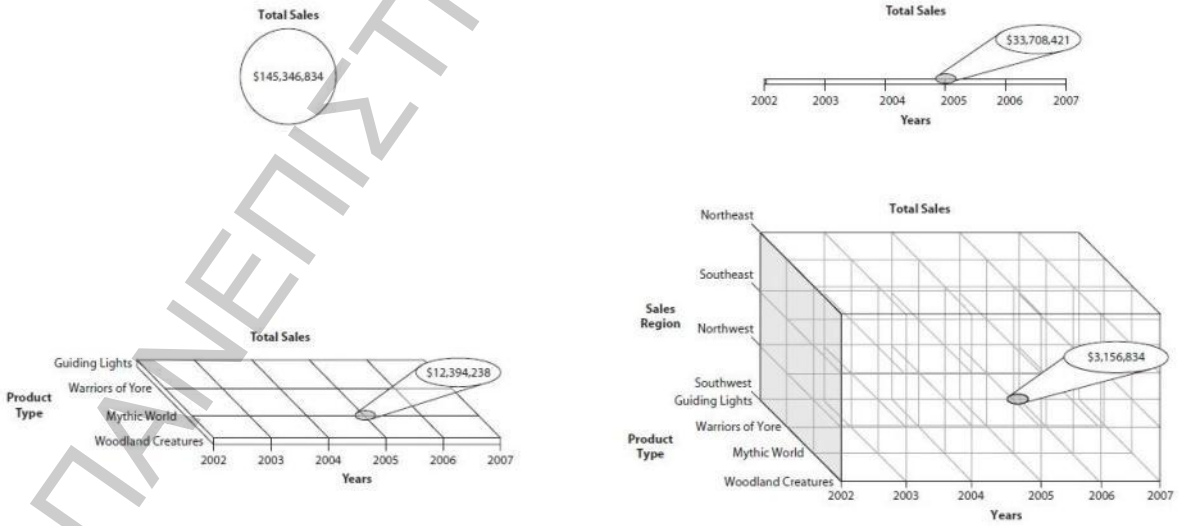
των δεδομένων όπου χρειάζεται, κάνουμε συγχώνευση καθώς και καθαρισμό των δεδομένων ώστε να συμφωνούν με την επιθυμητή τελική μορφή. Ένα παράδειγμα «βρώμικων δεδομένων» είναι πεδία ημερομηνίας τα οποία έχουν διαφορετικό format μεταξύ τους. Άλλο ένα τέτοιο παράδειγμα είναι ένα πεδίο location να βρίσκεται σε παραπάνω από 1 tables και να χρειάζεται συγχώνευση ή για παράδειγμα η χώρα να αναγράφεται με διαφορετικό τρόπο (σε άλλο πίνακα London και σε άλλο Lon-UK κτλ). Η διαδικασία του Transform είναι η πιο χρονοβόρα διαδικασία του ETL. Όταν ολοκληρωθεί, γίνεται Load των επεξεργασμένων και μετασχηματισμένων δεδομένων στο Data Warehouse όπου είναι και ο τελικός τους προορισμός που αποτελεί την πηγή δεδομένων του BI συστήματος. (14) (15)

Measures – Dimensions – Attributes - Hierarchies

Τα δεδομένα που χρησιμοποιούμε σε ένα σύστημα Business Intelligence χωρίζονται κυρίως σε 4 κατηγορίες, τα measures (μέτρα/μετρήσεις), τα dimensions (διαστάσεις), τα attributes (χαρακτηριστικά) και οι hierarchies (ιεραρχίες).

Τα Measures είναι πεδία τα οποία περιέχουν τις αριθμητικές τιμές που θέλουμε να μελετήσουμε. Είναι δηλαδή τα αριθμητικά δεδομένα τα οποία θα αναλυθούν για να βοηθήσουν στην λήψη αποφάσεων. Παραδείγματα measures πεδίων είναι το Amount, Tax Amount, Revenue, Total Sales κτλ. Οι πίνακες του Data Warehouse στους αποθηκεύονται τα Measures ονομάζονται Fact πίνακες. (3)

Τα Dimensions είναι πεδία τα οποία μας βοηθούν να κατηγοριοποιήσουμε τα measures. Είναι μια κατηγοριοποίηση που χρησιμοποιείται για να διαχωρίσει ένα Measure στα μικρότερα μέρη τα οποία το συνθέτουν. Όπως φαίνεται στην Εικόνα 2.2, μπορούμε να αναπαραστήσουμε ένα measure (πχ Total Sales) με ένα σημείο. Αν το κατηγοριοποιήσουμε ως προς το χρόνο μπορούμε να δούμε τις τιμές που συνέθεσαν το τελικό Total Sales. Επίσης μπορούμε επιπλέον να το διαχωρίσουμε σε τμήματα ανά product Type. Και εκεί βλέπουμε σε μεγαλύτερη λεπτομέρεια τις τιμές ανά χρόνο και προϊόν που συνθέτουν το τελικό Total Sales. Επιπλέον το χωρίζουμε σε περισσότερα κομμάτια, ως προς το Sales Region και μπορούμε να συνεχίσουμε να το αναλύουμε με τον ίδιο τρόπο. Τα Years, Product Type και Sales Region αποτελούν τα Dimensions μέσω των οποίων κατηγοριοποιήσαμε το measure Total Sales στις τιμές που το συνθέτουν. (3)



Εικόνα 2.2: Ανάλυση του Measures Total Sales ανα διάφορα Dimensions (3)

Τα Attributes είναι οι επιπρόσθετες πληροφορίες οι οποίες σχετίζονται με ένα dimension εκτός του id και της περιγραφής του. (3)

Τα Hierarchies είναι δομές από σχετικά dimensions και απαρτίζονται από 2 ή περισσότερα επίπεδα. Το Dimension του πιο υψηλού επιπέδου περιέχει ένα ή περισσότερα dimensions του επόμενου χαμηλότερου επιπέδου κ.ο.κ. Όπως τα dimensions κατηγοριοποιούν τα measures, έτσι και τα Hierarchies ομαδοποιούν τα Dimensions. Ένα παράδειγμα hierarchy είναι η χρονική ιεραρχία, το κάθε έτος περιέχει τετράμηνα, το κάθε τετράμηνο περιέχει μήνες κ.ο.κ. (3)

Σχήματα του Data Warehouse

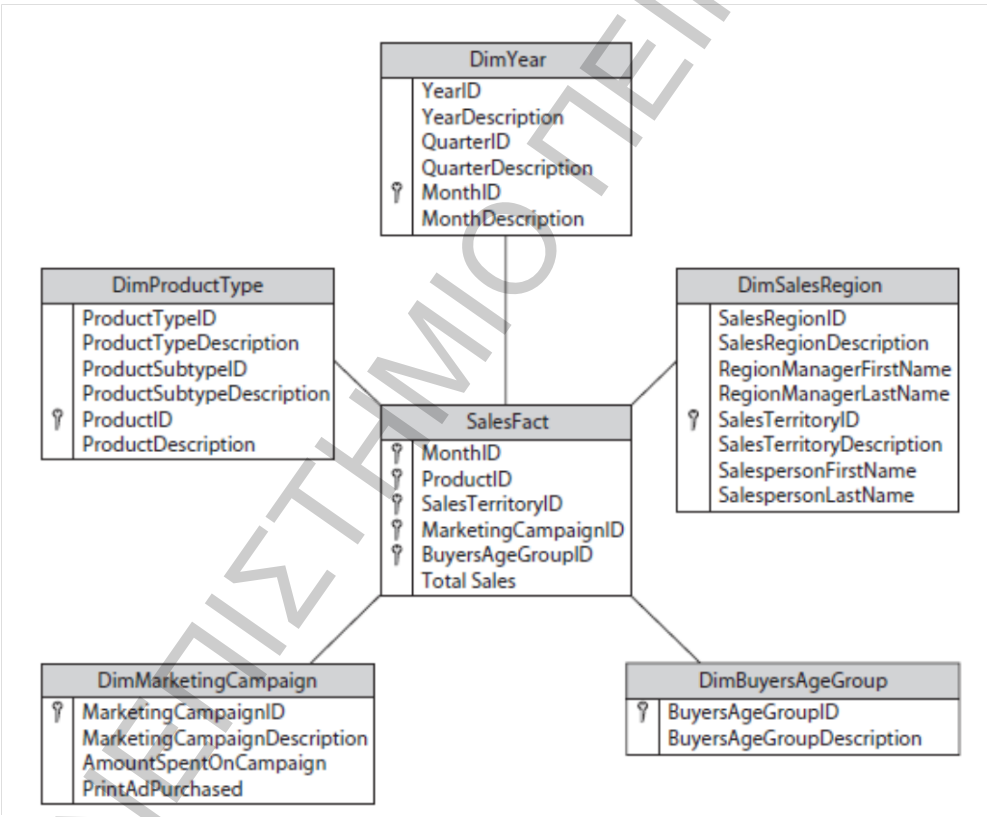
Τα δεδομένα ενός Data Warehouse αποθηκεύονται στη βάση με συγκεκριμένο τρόπο και δομή που ονομάζεται σχήμα (schema). Τα κυριότερα σχήματα βάσει των οποίων μπορεί να δομηθεί ένα data Warehouse είναι δύο, το Star και το Snowflake Schema.

Star Schema

Το Star Schema είναι ένα σχεσιακό σχήμα βάσης δεδομένων το οποίο διατηρεί τα Measures και Dimensions σε ένα Data Warehouse. Τα Measures αποθηκεύονται στο Fact Table και τα Dimension στα Dimension Tables. Το όνομά του προέρχεται από το σχήμα που παίρνει το διάγραμμα της βάσης δεδομένων που δημιουργείται. Στο κέντρο υπάρχει το Fact Table και γύρω του συνδέονται τα Dimension Tables. (3)

Το Fact Table περιέχει ένα πεδίο για κάθε measure και ένα πεδίο για κάθε Dimension το οποίο αποτελεί και δευτερεύον κλειδί του

αντίστοιχου Dimension πίνακα. Ο fact table είναι συνδεδεμένος με τους dimension tables με σχέση 1 προς 1 . Το πρωτεύον κλειδί του Fact Table είτε αποτελείται από όλα τα δευτερεύοντα κλειδιά προς τους Dimensions Tables με τους οποίους συνδέεται είτε από ένα πεδίο ενός μοναδικού ακεραίου αριθμού. Το όνομά του κάθε Fact πίνακα συντίθεται από το λεκτικό "Fact." και το όνομα του measure που περιγράφει (Εικόνα 2.3). (3) (15) (16)



Εικόνα 2.3: Star Schema (3)

Τα Dimensions αποθηκεύονται στα Dimension Tables. Τα Dimension Tables περιέχουν ένα μοναδικό id (συνήθως ακέραιο αριθμό) το οποίο αποτελεί το πρωτεύον κλειδί του και ένα πεδίο περιγραφής του dimension. Επίσης τα attributes αποθηκεύονται ως επιπλέον πεδία στον

πίνακα του αντίστοιχου dimension και αναπαριστούν επιπλέον πληροφορίες που σχετίζονται με αυτό. Τα πεδία που συνθέτουν μια ιεραρχία αποθηκεύονται στον πίνακα Dimension, κάθε επίπεδο της ιεραρχίας πρέπει να έχει το αντίστοιχο μοναδικό id του και το πρωτεύον κλειδί του πίνακα Dimension είναι το αντίστοιχο πεδίο του τελευταίου επιπέδου της ιεραρχίας. Το όνομά του αποτελείται από το λεκτικό "Dim." και το όνομα του dimension που περιγράφεται. (3) (15) (16)

Στο Star Schema υπάρχει μια εγγραφή στον Fact Table για κάθε μοναδικό συνδυασμό των Dimensions που σχετίζονται με αυτόν. Επιπλέον το Fact Table μπορεί να έχει παραπάνω από ένα measures, αρκεί αυτά να σχετίζονται με τα ίδια Dimensions. Στην περίπτωση που ο Fact συνδέεται με ένα πίνακα Dimension που έχει ιεραρχία, τότε το αντίστοιχο δευτερεύον κλειδί του Fact συνδέεται με το ID του τελευταίου επιπέδου της ιεραρχίας και έτσι έχει μια εγγραφή για κάθε συνδυασμό των μελών του επιπέδου αυτού. Επομένως στον Fact πίνακα αποθηκεύονται μόνο τα measures που αντιστοιχούν στο τελευταίο επίπεδο της ιεραρχίας. Για τον υπολογισμό των measures των μεγαλύτερων επιπέδων γίνονται μαθηματικοί υπολογισμοί (συνήθως πρόσθεση) των τιμών των measure των χαμηλότερων επιπέδων (πχ ο υπολογισμός του Quantity ενός μήνα υπολογίζεται από το άθροισμα των Quantities των ημερών που απαρτίζουν αυτό το μήνα). (3)

Το πλεονέκτημα αυτού του σχήματος είναι ότι χρειάζονται πολύ λίγα joins των πινάκων του Data Warehouse για να ανακτηθεί η επιθυμητή πληροφορία, άρα έχουμε και μεγαλύτερη ταχύτητα

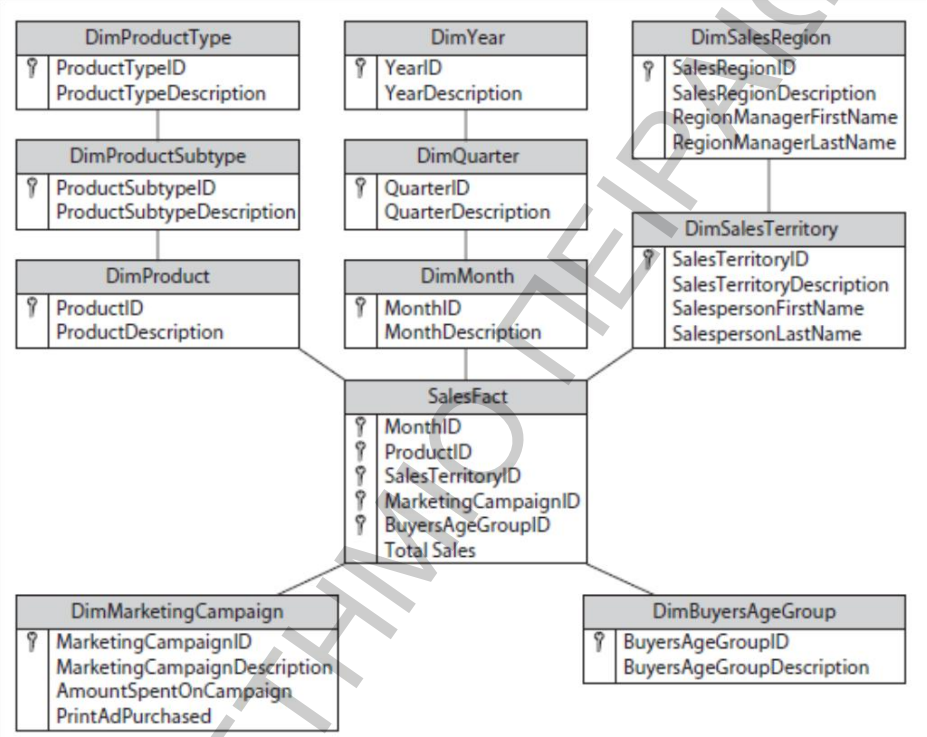
ανάκτησης. Βασικό μειονέκτημα είναι ότι υπάρχουν αρκετά διπλότυπα (redundancy). (15) (16)

Snowflake Schema

Στο Star Schema ένας πίνακας Dimension μπορεί να συνδέεται μόνο με τον Fact Table και πρέπει να περιέχει όλη την απαραίτητη πληροφορία. Το **Snowflake σχήμα** αποτελεί μια διαφοροποίηση του Star σχήματος με βασική διαφορά ότι τα Dimensions μπορούν να συνδέονται και με άλλους πίνακες (Sub-dimensions) (Εικόνα 2.4). Έτσι, το Snowflake Schema έχει πολλές ομοιότητες με το σχήμα των σχεσιακών βάσεων δεδομένων αφού ένα Dimension Table μπορεί να συνδέεται (μέσω δευτερευόντων κλειδιών) με άλλους δευτερεύοντες πίνακες επιτυγχάνοντας normalized δεδομένα και αποφεύγοντας τα διπλότυπα (Redundancy). Όσον αφορά τις ιεραρχίες, κάθε επίπεδο της ιεραρχίας έχει τον δικό του πίνακα ενώ ο Dimension πίνακας διατηρεί μόνο το κλειδί του τελευταίου επιπέδου. (3) (15) (16)

Το χαρακτηριστικό του Snowflake σχήματος είναι ότι έχουμε κανονικοποιημένα δεδομένα (Normalized Data) με αποτέλεσμα το μέγεθος των Dimensions είναι πολύ μικρότερο από το αντίστοιχο μέγεθος στο Star Schema. Επομένως το πλεονέκτημα είναι ότι εξοικονομούμε χωρητικότητα (αποφυγή Redundancy) και ταχύτητα κατά τη διάρκεια του Data Load από το OLTP σύστημα, και το μειονέκτημα ότι όσο αυξάνεται ο αριθμός των πινάκων, χρειάζονται και περισσότερα JOIN για να έχουμε πρόσβαση σε όλες τα πεδία των εγγραφές που σχετίζονται τα Dimensions με αποτέλεσμα να μειώνεται σημαντικά η απόδοση του συστήματος.

Στον αντίποδα, με το Star Schema έχουμε πολλές διπλοεγγραφές (redundancy – de-normalized data) που έχουν ως αποτέλεσμα μεγαλύτερη χρονική καθυστέρηση κατά το Data Load αλλά πολύ πιο μεγάλη ταχύτητα ανάκτησης των δεδομένων (3) (15) (16)



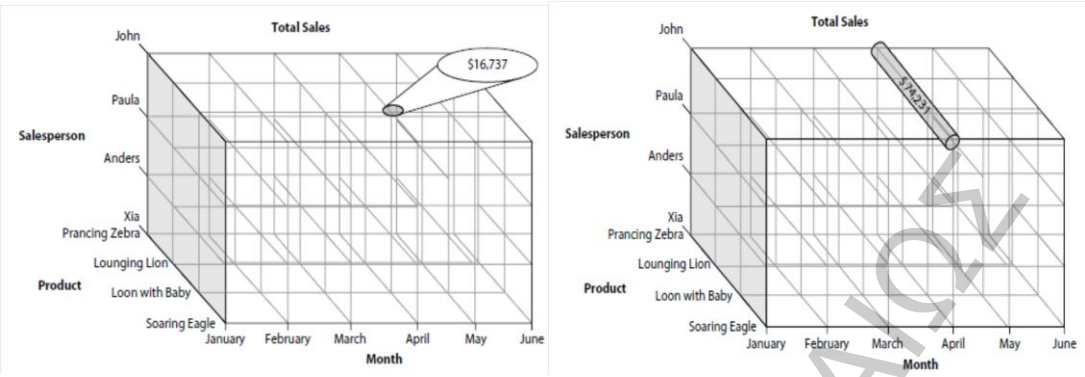
Εικόνα 2.4 Snowflake Schema (3)

OLAP

Το **OLAP** αποτελεί μια συλλογή από τεχνολογίες που έχουν σχεδιαστεί για ad-hoc πρόσβαση των δεδομένων που αποτελούν αντικείμενο ανάλυσης Business Intelligence. Το Data warehouse αποτελεί την πηγή των δεδομένων για τα OLAP συστήματα. Το μοντέλο που χρησιμοποιείται είναι η χρήση πολλαπλών διαστάσεων (Multidimensional) και τα δεδομένα οργανώνονται σε κύβους. Κύβος είναι η δομή που περιέχει τις τιμές ενός ή περισσοτέρων Measures για κάθε

συνδυασμό όλων των dimensions του (leaf - level values). Επιπλέον ο κύβος περιέχει aggregates τα οποία είναι τιμές που προκύπτουν από συνδυασμό ή μαθηματική επεξεργασία (πχ άθροισμα) των τιμών των measures για ένα μεμονωμένο Dimension ή για ένα συνδυασμό Dimensions. Αυτά τα pre-aggregations πραγματοποιούνται σε τακτά χρονικά διαστήματα με μια συγκεκριμένη διαδικασία (process) με αποτέλεσμα να βελτιώνεται κατά πολύ η απόδοση του συστήματος και η ταχύτητα ανάκτησης δεδομένων. Έτσι, σε ένα κύβο μπορούμε να μάθουμε μια τιμή που χρειαζόμαστε ανά πολλαπλά Dimensions (πχ κόστος ανά χώρα, χρόνο και product line). Μπορούν να δημιουργηθούν πολλοί και διαφορετικοί κύβοι από την ίδια πηγή δεδομένων. (3) (13) (15) (16)

Για παράδειγμα στην Εικόνα 2.5, η τιμή του John για το προϊόν Loon with Baby τον Απρίλιο είναι leaf level value, ενώ το Total Sales του John για τον Απρίλιο είναι Aggregate γιατί η τιμή αυτή προέρχεται από το άθροισμα των Total Sales για κάθε προϊόν. Επίσης επειδή πολλά από τα aggregates χρειάζονται χρόνο να υπολογιστούν σε πραγματικό χρόνο (on the fly) όταν τα ζητάει ένας χρήστης, ο υπολογισμός των τιμών τους γίνεται κατά τη διάρκεια του Process του κύβου και γι' αυτό και ονομάζονται preprocessed aggregates.



Εικόνα 2.5: Αναπαράσταση Leaf Level Value και Aggregated Value (3)

Επίσης στον κύβο ορίζονται τα KPI τα οποία είναι τα αρχικά του όρου Key Performance Indicator. Το KPI ορίζεται ως μια ποσοτική μέτρηση που χρησιμοποιείται για τον καθορισμό και τη μέτρηση της προόδου ενός οργανισμού ή μιας επιχείρησης με στόχο την επίτευξη των επιχειρηματικών της στόχων.

Τέλος, εφόσον έχουμε δημιουργήσει και τα OLAP συστήματα χρειάζεται μια εφαρμογή η οποία να διαβάζει τα δεδομένα του BI και να τα εμφανίζει στην κατάλληλη μορφή. Η πιο συνηθισμένη μορφή αναπαράστασης των BI δεδομένων είναι τα Pivot Tables και τα Pivot Charts. Τα Pivot Tables είναι δυναμικοί πίνακες στους οποίους μπορούμε να αναλύσουμε πολυδιάστατα δεδομένα και μεταβλητές τα οποία τα προσθέτουμε και τα αφαιρούμε σε πραγματικό χρόνο. Τα Pivot Charts έχουν την ίδια λογική με τα Pivot Tables με τη διαφορά ότι τα αποτελέσματα οπτικοποιούνται σχηματικά σε γραφήματα και πίτες. Επιπλέον μπορούν να χρησιμοποιηθούν KPI charts καθώς και χάρτες.

Συνοψίζοντας, σύμφωνα με το (17) τα βασικά στοιχεία ενός BI συστήματος είναι τα εξής:

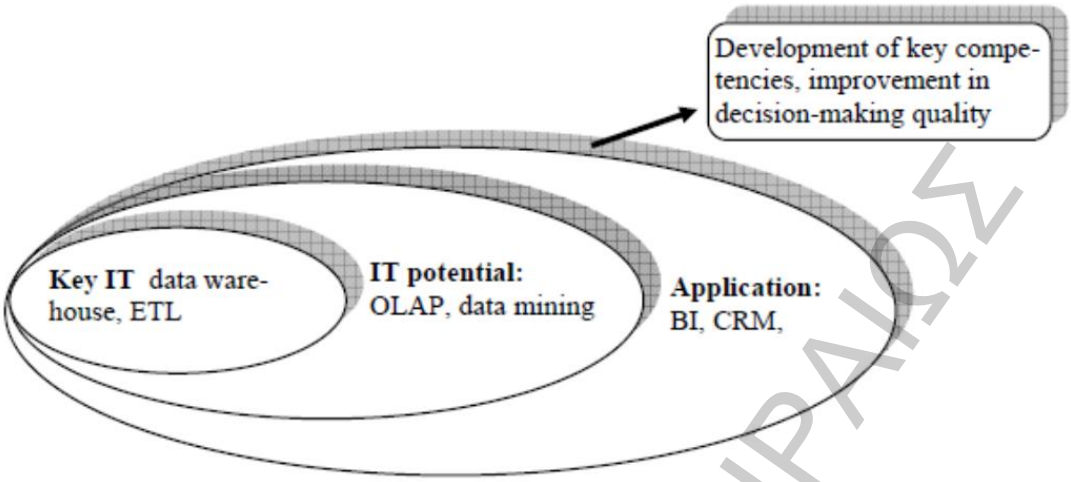
- **Operational Source Systems**: Στην πραγματικότητα δεν αποτελούν μέρος της λύσης αλλά πηγή δεδομένων για το σύστημα BI. Παραδείγματα τέτοιων συστημάτων είναι ERP systems, CRM systems, weather information systems, spatial information systems, οικονομικά στοιχεία κτλ.
- **Data Staging Area**: Λαμβάνουν την πληροφορία από τα Operational Source Systems (Extract) και πραγματοποιούν μετασχηματισμούς και καθαρισμό των δεδομένων αυτών ώστε να λάβουν μια μορφή η οποία έχει προκαθοριστεί ως η κατάλληλη για το BI σύστημά μας (Transform). Έπειτα τα δεδομένα είναι κατάλληλα να φορτωθούν στο Data Warehouse (Load).
- **Data Presentation Area**: Αυτή η περιοχή αποτελείται από το Data Warehouse το οποίο είναι το συνονθύλευμα πολλών Data Mart. Σύμφωνα με τους Kimball & Ross το Data Warehouse είναι «*το σύνολο των δεδομένων των περιοχών Staging και Presentation μιας εταιρίας, όπου τα λειτουργικά δεδομένα είναι δομημένα με τον κατάλληλο τρόπο για επερωτήσεις, βέλτιστη ανάλυση και ευκολία χρήσης*». Τα Data Mart κατά τους ίδιους είναι ένα μέρος του Data Warehouse που περιέχει πληροφορίες για ένα μέρος της επιχείρησης (πχ Sales). Τα δεδομένα σε ένα Data Warehouse είναι δομημένα σε διαστάσεις και όχι σχεσιακά όπως στα operational systems. Αυτό σημαίνει ότι υπάρχει ένας πίνακας με facts και διάφοροι άλλοι πίνακες με dimensions που όλα μαζί συνδέονται

στον Fact Πίνακα. Όσο μειώνεται ο αριθμός των Joins τόσο αυξάνεται το Performance (καθώς και το Redundancy).

- **Data Access Tools**: Αποτελούν τα εργαλεία μέσω των οποίων οι τελικοί χρήστες αντλούν την πληροφορία από το Data Warehouse. Υπάρχουν πολλές εταιρίες που έχουν δημιουργήσει τέτοια συστήματα (Cognos, Excel, SAP BW, MicroStrategy) καθώς και Custom λύσεις.

Άλλος ένας τρόπος προσέγγισης (Εικόνα 2.6) γίνεται από το (11) βάσει του οποίου τα βασικά μέρη μιας υποδομής BI είναι τα εξής:

- **Key Information Technologies** οι οποίες σχετίζονται με την εξόρυξη των δεδομένων και την αποθήκευσή τους, όπως για παράδειγμα τα ETL εργαλεία και τα data Warehouses.
- **Information Technologies Potential** τα οποία κατά βάση ασχολούνται με την ανάλυση και την παρουσίαση των δεδομένων (πχ τεχνικές OLAP και data mining)
- **BI Applications** τα οποία υποστηρίζουν την λήψη αποφάσεων σε διάφορους κλάδους όπως η παραγωγή, οι πωλήσεις, η διαχείριση ανταγωνισμού, τα οικονομικά κτλ.

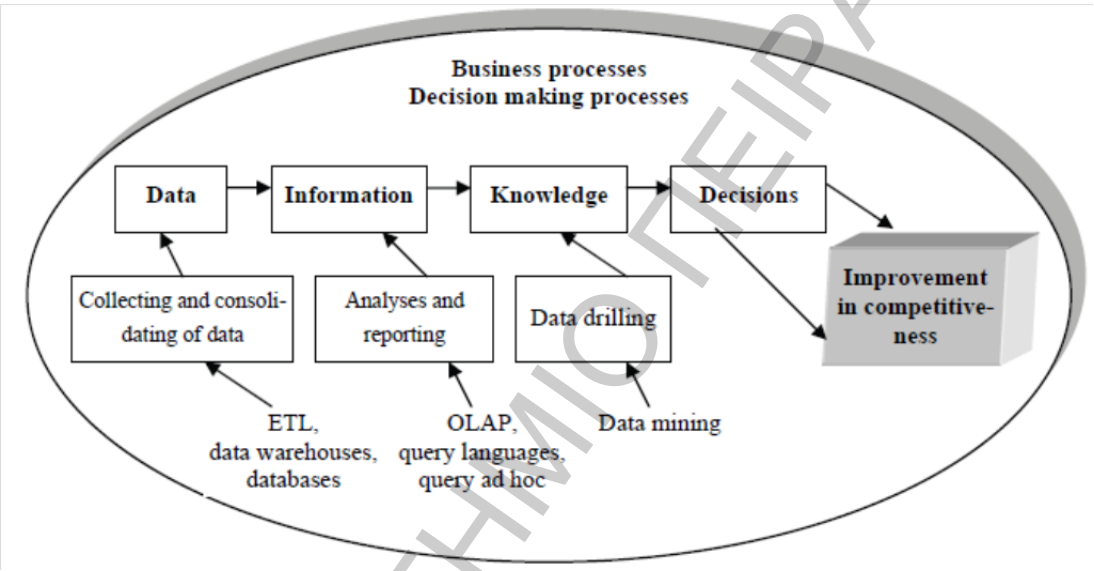


Εικόνα 2.6: Βασικά Μέρη μιας λύσης Business Intelligence (11)

Συμπερασματικά, αρχικά πρέπει να δημιουργηθεί το Data Warehouse από το οποίο θα αντληθούν τα δεδομένα. Λαμβάνουμε λοιπόν τα δεδομένα από τις OLTP βάσεις δεδομένων και τα μετασχηματίζουμε/τροποποιούμε με τέτοιο τρόπο ώστε να οργανωθούν σε πίνακες Fact και Dimension και να έχουν την επιθυμητή μορφή. Έπειτα, όταν η δημιουργία του Data Warehouse έχει ολοκληρωθεί πρέπει να δημιουργηθούν OLAP συστήματα τα οποία θα περιέχουν κύβους με τα δεδομένα του Data Warehouse, οργανωμένες διαστάσεις και ιεραρχίες, KPIs, υπολογιστικά πεδία κ.α. Στα OLAP συστήματα τα δεδομένα, έπειτα από pre-aggregations, βρίσκονται στην τελική τους μορφή. Τέλος πρέπει να δημιουργηθεί μια εφαρμογή φιλική στον χρήστη, μέσω της οποίας ο τελικός χρήστης θα έχει πρόσβαση στα δεδομένα ώστε να προβεί στην ανάλυσή τους.

2.1.3. Σχεδιασμός και Ανάλυση BI Συστημάτων

Τα συστήματα BI, σύμφωνα με το (18), αποτελούν λύσεις που μετασχηματίζουν τα δεδομένα σε πληροφορία, τη πληροφορία σε γνώση και δημιουργούν το κατάλληλο περιβάλλον για στρατηγικό τρόπο σκέψης και δράσης και λήψη βέλτιστων αποφάσεων (Εικόνα 2.7).



Εικόνα 2.7: Ανάλυση Διαδικασίας Λήψεως Αποφάσεων (18)

Θα πρέπει λοιπόν να οριστούν τα στοιχεία και οι πληροφορίες που απαιτούνται ώστε η εφαρμογή να είναι χρήσιμη και αποδοτική. Σύμφωνα με το (3) οι ερωτήσεις που πρέπει να απαντηθούν κατά τη διάρκεια της ανάλυσης απαιτήσεων είναι:

- Ποια δεδομένα, στατιστικά και γραφήματα χρειάζονται για να οδηγηθούμε σε αποτελεσματικό Decision Making? (measures)

- Ως προς ποια στοιχεία πρέπει να μπορεί να χωριστεί αυτή η πληροφορία ώστε να μπορεί να αναλυθεί σωστά? (dimensions και hierarchies)
- Ποια επιπλέον πληροφορία θα μπορούσε να συσχετιστεί με αυτά τα δεδομένα ? (attributes).

Στη συνέχεια θα πρέπει να αντιστοιχήσουμε αυτές τις πληροφορίες με ήδη υπάρχοντα δεδομένα, εντοπίζοντας τα πεδία που αντιστοιχούν σε κάθε πληροφορία στο OLTP σύστημά. Αν δεν υπάρχουν αυτά τα πεδία αυτούσια, τότε αναζητούνται τρόποι να υπολογιστούν από σύνθεση άλλων υπαρχόντων πεδίων. Επιπλέον, μπορεί τα στοιχεία που χρειαζόμαστε να βρίσκονται σε διαφορετικά OLTP συστήματα, είτε ακόμη και σε αρχεία κειμένου ή λογιστικά φύλλα. Σε αυτές τις περιπτώσεις πρέπει να συγχωνεύσουμε τα δεδομένα από όλες αυτές τις ετερογενείς πηγές δεδομένων ώστε να αντλήσουμε την επιθυμητή πληροφορία. (3) Αν παρ' όλα αυτά δεν υπάρχει τρόπος να αντλήσουμε την πληροφορία που χρειάζεται τότε είτε ζητούμε την αλλαγή του OLTP συστήματος ώστε να τα συμπεριλάβει είτε αναπροσαρμόζουμε τις απαιτήσεις της λύσης Business Intelligence

Κατά τη διαδικασία της προετοιμασίας δημιουργίας ενός Data Warehouse, εκτός των άλλων, πρέπει να πραγματοποιηθούν οι εξής ενέργειες (18):

- Να κατηγοριοποιηθούν τα δεδομένα των πηγών δεδομένων με βάση τη χρησιμότητά τους (απαραίτητα, χρήσιμα, μη χρήσιμα).

- Να βρεθεί και να εντοπιστεί η θέση των δεδομένων στις βάσεις αυτές.
- Να βρεθούν οι σχέσεις μεταξύ των δεδομένων που μπορεί να βρίσκονται σε διαφορετικές πηγές.
- Να βρεθεί η μορφή και η δομή αυτών των δεδομένων καθώς και η σχέση που έχουν με την τελική πληροφορία που χρειαζόμαστε.
- Να βρεθούν πιθανές πηγές λαθών καθώς και οι πιθανοί περιορισμοί στην λήψη των δεδομένων.

Γενικότερα, μια μεθοδολογία που μπορεί να ακολουθηθεί είναι η εξής:

1. Να καταγραφούν οι πληροφορίες που χρειάζονται για να μπορέσει να γίνει ανάλυση.
2. Να οριστούν οι κύβοι που πρέπει να δημιουργηθούν, τα Fact και τα Dimensions, και τα πεδία-πληροφορία που θα περιέχει το καθένα. (τι measures θα υπάρχουν στα Facts, τι attributes και Hierarchies στα Dimensions, να οριστούν τα υπολογιστικά πεδία κτλ).
3. Να γίνει μια έρευνα ώστε να βρεθούν αν υπάρχουν τα απαραίτητα δεδομένα που θα οδηγήσουν στις πληροφορίες αυτές. Αν δεν υπάρχουν να γίνει μια προσπάθεια να δημιουργηθούν (Calculated Fields) είτε να γίνει μια αναθεώρηση των πληροφοριών που μπορούμε να διαθέσουμε.
4. Να βρεθούν οι πηγές που περιέχουν τα δεδομένα και να πραγματοποιηθεί μια καταγραφή της μορφής και της θέσης στην οποία βρίσκονται.
5. Να γίνει μια ανάλυση της δομής του Data Warehouse που θα δημιουργηθεί: ποιο θα είναι το σχήμα του, ποιο θα είναι το mapping

των πεδίων του με τα πεδία των Πηγών δεδομένων, ποια είναι η αρχική μορφή, ποια η τελική των δεδομένων και πως θα οδηγηθούμε σε αυτή.

6. Έπειτα πρέπει να οριστούν και να περιγραφούν οι μετασχηματισμοί που θα γίνουν και ο καθαρισμός των δεδομένων ώστε να αποθηκευτούν στην επιθυμητή μορφή και θέση.
7. Να οριστεί το Security του Analysis και το ποιοι θα έχουν πρόσβαση σε ποια δεδομένα.
8. Επιπλέον πρέπει να οριστεί κάθε πότε θα γίνεται Update των δεδομένων, με ποιο τρόπο (incremental-full) κτλ.

Εφόσον έχει ολοκληρωθεί η ανάλυση και γνωρίζουμε ποια δεδομένα θα αντλήσουμε από ποιες πηγές και σε ποια μορφή μπορούμε να ξεκινήσουμε την υλοποίηση του συστήματος.

2.2. Πλατφόρμες Υλοποίησης Συστημάτων BI

Στην παγκόσμια αγορά υπάρχουν ολοένα και περισσότερες εταιρίες οι οποίες διαθέτουν πακέτα που σχετίζονται με τα BI συστήματα. Οι σημαντικότερες εξ' αυτών καθώς και τα αντίστοιχα προϊόντα τους είναι τα εξής:

Microsoft

Η λύση της Microsoft (19) που αφορά το BI είναι η λύση που χρησιμοποιείται στο πρακτικό μέρος αυτής της εργασίας και αποτελείται από 3 διαφορετικά συστήματα.

- SQL Server 2008 R2: Περιλαμβάνει τα εξής:
    - Integration Services: Είναι το βασικό εργαλείο για να μπορέσει να γίνει η διαδικασία του ETL και μέσω αυτού δημιουργείται η βάση Data Warehouse.
    - Analysis Services: Τα δεδομένα του Data Warehouse οργανώνονται σε κύβους με pre-aggregated δεδομένα που είναι οργανωμένα σε measures, dimensions, hierarchies, actions, KPIs, Calculated Members και μέσω ενός OLAP Service διατίθενται στις εφαρμογές BI.
    - Reporting Services: Παρέχονται εργαλεία για τη δημιουργία Report.
- SharePoint και Excel: Είναι εφαρμογές τις οποίες χρησιμοποιούν οι τελικοί χρήστες για να έχουν πρόσβαση στα δεδομένα BI και

παρέχουν οπτικοποίηση των δεδομένων με διάφορους τρόπους (pivot Tables, pivot chart κ.α.)

SAP

Η πλατφόρμα BI της SAP (20) είναι το SAP Netweaver Business Warehouse. Τα εργαλεία Data Warehousing και BI Platform χρησιμοποιούνται για το ETL, τη δημιουργία του Data Warehouse και των κύβων και το BI Suite: Business Explorer δίνει τη δυνατότητα πρόσβασης σε αυτά τα δεδομένα από τους χρήστες των συστημάτων SAP.

IBM

Η εταιρία IBM έχει δημιουργήσει μια Web-Based λύση BI που ονομάζεται IBM Cognos BI (21). Τα βασικά εργαλεία του Cognos BI είναι τα: Connection, Administration, Configuration, Business Insight – Advanced, Report Studio, Query Studio, Analysis Studio, Event Studio, Metric Studio & Designer, Framework Manager, Transformer, Map Manager.

3. Υλοποίηση Συστήματος Επιχειρηματικής Ευφυΐας

3.1. Περιγραφή

Στα πλαίσια της διπλωματικής εργασίας έχει υλοποιηθεί ένα ολοκληρωμένο σύστημα επιχειρηματικής ευφυΐας το οποίο περιλαμβάνει την ανάλυση των απαιτήσεων της εταιρίας Adventure Works Cycles, τη δημιουργία Data Warehouse και OLAP κύβων, καθώς και την δημιουργία ενός Web Site μέσω του οποίου θα υπάρχει πρόσβαση στα δεδομένα για ανάλυση.

Adventure Works Cycles

Η βάση δεδομένων Adventure Works είναι μια Demo βάση που παρέχεται από την Microsoft για σκοπούς evaluation και testing. Η βάση Adventure Works είναι η βάση δεδομένων της εταιρίας Adventure Works Cycles η οποία είναι μια πολυεθνική εταιρία κατασκευής ποδηλάτων. Η εταιρία κατασκευάζει και εμπορεύεται μεταλλικά και σύνθετα ποδήλατα στις εμπορικές αγορές της Βορείου Αμερική, της Ευρώπης και της Ασίας. Η λειτουργική της βάση βρίσκεται στο Bothell της Washington (290 υπάλληλοι) ενώ διάφορες ομάδες της εταιρίας δραστηριοποιούνται σε διάφορα μέρη του κόσμου. Η θυγατρική εταιρία Importadores Neptuno δραστηριοποιείται στο Μεξικό και αποτελεί από το 2001 το βασικό κατασκευαστή και διανομέα ποδηλάτων της Adventure Works. Βασικοί στόχοι της εταιρίας είναι η διεύρυνση του μεριδίου αγοράς μέσω της στοχευόμενης πώλησης στους καλούς πελάτες, της διεύρυνσης της

διάθεσης των προϊόντων μέσω του διαδικτύου και της μείωσης του κόστους πωλήσεων μέσω της μείωσης του παραγωγικού κόστους. (22)

Οι πληροφορίες που σχετίζονται με τους πελάτες και τις πωλήσεις είναι ένα σημαντικό κομμάτι της βάσης δεδομένων της Adventure Works. Υπάρχουν δύο τύποι πελατών, οι μεμονωμένοι πελάτες που αγοράζουν τα προϊόντα μέσω του εταιρικού Web Site και τα καταστήματα χονδρικής ή λιανικής που αγοράζουν τα προϊόντα από τους εταιρικούς αντιπροσώπους. (23) Οι κατηγορίες προϊόντων που εμπορεύεται η εταιρία είναι ποδήλατα, ανταλλακτικά εξαρτήματα ποδηλάτων (πχ ρόδες, πεντάλ, φρένα κ.α.), προϊόντα ένδυσης που σχετίζονται με ποδηλάτες και αξεσουάρ ποδηλάτων (πχ κράνη, επιγονατίδες κ.α.). (24) Όσον αφορά τις προμήθειες, το τμήμα προμηθειών αγοράζει τις πρώτες ύλες και τα εξαρτήματα που χρειάζονται για την κατασκευή των ποδηλάτων. Επιπλέον η Adventure Works αγοράζει εξοπλισμό ποδηλάτου, όπως μπουκαλάκια νερού, κράνη κ.α. και τα μεταπουλάει. (25) Τέλος, η βάση δεδομένων της εταιρίας περιέχει αρκετές πληροφορίες που σχετίζονται με την παραγωγή των προϊόντων όπως τα υλικά του κάθε προϊόντος, η τοποθεσία κατασκευής, η θέση του στις αποθήκες, τεχνικά εγχειρίδια και άλλα. (26)

Στα πλαίσια αυτής της εργασίας δημιουργείται ένα Data Warehouse βασισμένο σε ανάγκες της Adventure Works χρησιμοποιώντας ως πηγή δεδομένων την βάση δεδομένων AdventureWorks2008R2.

Λογισμικό και Πλατφόρμες ανάπτυξης

Για τη δημιουργία του Data Warehouse και των OLAP κύβων χρησιμοποιήθηκε το SQL Server Business Intelligence Studio της Microsoft και συγκεκριμένα ένα SSIS Integration Services Project για τη δημιουργία του Data Warehouse και ένα SSAS Analysis Services Project για τη δημιουργία των κύβων. Για τις ανάγκες της δημιουργίας του Web Site χρησιμοποιήθηκε το Microsoft Visual Studio 2010 και συγκεκριμένα η τεχνολογία ASP.net. Το Pivot Table και το Pivot Chart είναι Controls που παρέχονται από την εταιρία Radar-Soft και επιτρέπεται η χρήση τους για Evaluation Use. Επιπλέον χρησιμοποιήθηκε ο χάρτης της Google Maps καθώς και οι Open Source βιβλιοθήκες Artem.Google και Artem.GeoCoding για την οπτικοποίηση των πληροφοριών πάνω στο χάρτη. Τέλος χρησιμοποιήθηκαν εικονίδια και θέματα CSS που λήφθηκαν από το (27) και το (28). Τα εικονίδια του χάρτη έχουν ληφθεί από το (29). Η εφαρμογή έχει ελεγχθεί και λειτουργεί ορθά σε Internet Explorer 9 και σε ανάλυση οθόνης 1366*768.

Επισημαίνεται ότι επειδή δεν χρησιμοποιείται η εταιρική άδεια της εταιρίας Radar Soft τα Controls Pivot Table και Pivot Chart, σύμφωνα με την τρέχουσα πολιτική, ανανεώνονται σε τακτά χρονικά διαστήματα (συνήθως κάθε 2 μήνες) και οι νέες βιβλιοθήκες πρέπει να εγκατασταθούν στο ASP Project για να μπορέσει να εκτελεστεί το Web Site. Η λύση που χρησιμοποιείται είναι είτε η εγκατάσταση των νέων βιβλιοθηκών στο Project είτε η αλλαγή του ρολογιού του λειτουργικού συστήματος στο οποίο εκτελείται το Site στην ημερομηνία παράδοσης της εργασίας αυτής (../../12 έως ../../12) που είναι και η ημερομηνία λειτουργίας

της τελευταίας βιβλιοθήκης που χρησιμοποιήθηκε στο παρόν παραδοθέν Project.

3.2. Ανάλυση Απαιτήσεων

Η βασική ανάλυση των απαιτήσεων επικεντρώνεται σε δυο βασικά σημεία: ποιες πληροφορίες θέλουν να αντλήσουν οι Decision Makers της εταιρίας Adventure Works και σε τι μορφή θέλουν να δουν τη συγκεκριμένη πληροφορία. Κατά κύριο λόγο οι απαντήσεις του πρώτου ερωτήματος θα μας οδηγήσουν στη δημιουργία του Data Warehouse και των OLTP κύβων ενώ το δεύτερο ερώτημα θα μας οδηγήσει στη δημιουργία του Web User Interface το οποίο θα οπτικοποιεί αυτά τα δεδομένα.

Ανάλυση Data Warehouse και OLAP κύβων.

Οι βασικές απαιτήσεις των Decision Makers συνοψίζονται σε 3 ομάδες:

- Order Sales: Περιλαμβάνει στοιχεία που σχετίζονται με παραγγελίες και όχι με τα επιμέρους προϊόντα που έχει η κάθε παραγγελία (ανάλυση πωλήσεων ως προς τις παραγγελίες). Χρειάζονται οι πληροφορίες Freight, Number of Order Sales, Tax Amount και Total Due. Αυτές οι πληροφορίες θα πρέπει να μπορούν να αναλυθούν – ομαδοποιηθούν ανά Address of Bill, Address of Shipment, Customer, Due Date, Order Date, Sales Person, Card Type και Shipment Method.

- Line Sales: Έχει στοιχεία για τα προϊόντα της κάθε παραγγελίας (ανάλυση πωλήσεων ως προς το προϊόν). Χρειάζονται οι πληροφορίες Number of Line Sales, Line Total, Order Quantity, Product Cost, Unit Price και Unit Price Discount. Τα χαρακτηριστικά βάσει των οποίων θα μπορούν να αναλυθούν οι πληροφορίες αυτές είναι Address of Bill, Address of Shipment, Currency, Customer, Due Date, Line Sales Details, Order Date. Product, Sales Person, Special Offer και Card Type.
- Geo Sales: Εμφανίζει πληροφορίες των προϊόντων της κάθε παραγγελίας με geospatial δεδομένα (ανάλυση πωλήσεων προϊόντων ως προς τη περιοχή πώλησης). Οι απαραίτητες πληροφορίες είναι Number of Geo Sales, Line Total, Order Quantity, Unit Price, Unit Price Discount, Longitude, Latitude (απεικόνιση σε χάρτη). Η ανάλυση θα μπορεί να γίνει είτε ως προς το Bill Address είτε ως προς το Shipment Address.

Βάση των παραπάνω πληροφοριών οδηγούμαστε στην ανάγκη δημιουργίας 3 OLAP κύβων, του Order Sales, του Line Sales και του Geo Sales. Οι κύβοι αυτοί θα περιέχουν τα measures και τα Dimensions (με hierarchies και attributes) που εμφανίζονται στα παρακάτω σχήματα.

Order Sales Cube			
Measures	**Dimensions**	**Attributes**	**Hierarchies**
• Freight • Order Sales Count • Tax Amt • Total Due	Bill To Address	• Address • Address ID • City • Country • Country Group • Country Region	Bill To Address Hierarchy

		• State Province	
	Currency	• Currency • Currency Code	
	Customer	• Customer • Customer ID • Person Type • Store	
	Due Date	• Year • Year Quarter • Year Month	Due Date Hierarchy
	Order Date	• Year • Year Quarter • Year Month	Order Date Hierarchy
	Sales Person	• Business Entity ID • Gender • Job Title • Sales Person	Sales Person Hierarchy
	Ship To Address	• Address • Address ID • City • Country • Country Group • Country Region • State Province	
	Card Type	Card Type	
	Shipment Method	Shipment Method	

Line Sales Cube			
Measures	**Dimensions**	**Attributes**	**Hierarchies**
• Line Sales Count • Line Total • Order Qty • Product Cost • Unit Price • Unit Price Discount	Bill To Address	• Address • Address ID • City • Country • Country Group • Country Region • State Province	Bill To Address Hierarchy
	Currency	• Currency • Currency Code	
	Customer	• Customer • Customer ID • Person Type • Store	
	Due Date	• Year • Year Quarter • Year Month	Due Date Hierarchy
	Line Sales Details	• Online Order • Order Status	

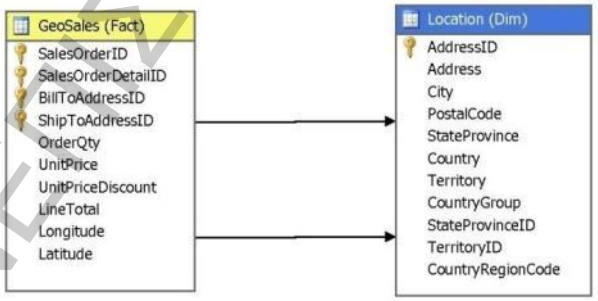
		• Sales Order Detail ID • Sales Order ID	
	Order Date	• Year • Year Quarter • Year Month	Order Date Hierarchy
	Product	• Color • Finished Goods • Make • Product Category • Product Class • Product Line • Product Model • Product Name • Product Number • Product Style • Product Subcategory	• Category • Finished Goods • Make • Product Class • Product Color • Product Line • Product Style
	Sales Person	• Business Entity ID • Gender • Job Title • Sales Person	Sales Person Hierarchy
	Ship To Address	• Address • Address ID • City • Country • Country Group • Country Region • State Province	
	Card Type	Card Type	
	Shipment Method	Shipment Method	

Geo Sales Cube

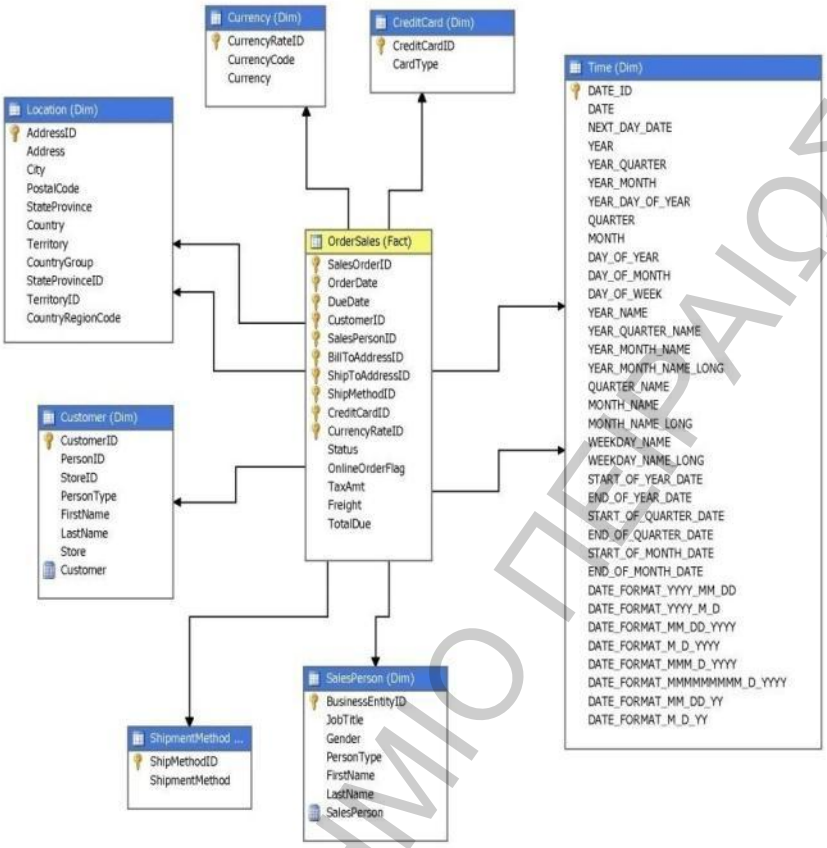
Measures	Dimensions	Attributes	Hierarchies
• Geo Sales Count • Latitude • Line Total • Longitude • Order Qty • Unit Price • Unit Price Discount	Bill To Address	• Address • Address ID • City • Country • Country Group • Country Region • State Province	Bill To Address Hierarchy
	Ship To Address	• Address • Address ID • City • Country • Country Group • Country Region • State Province	

Πριν ξεκινήσουμε τη διαδικασία της διερεύνησης και της άντλησης των δεδομένων από τις πηγές δεδομένων, θα πρέπει να ορίσουμε το σχήμα του Data Warehouse που θα δημιουργήσουμε. Στη συγκεκριμένη περίπτωση το σχήμα που θα χρησιμοποιήσουμε είναι το σχήμα STAR. Οι λόγοι που οδηγούμαστε σε αυτή την επιλογή είναι αρχικά γιατί μας ενδιαφέρει η ταχύτητα ανάκτησης της πληροφορίας κατά τη διάρκεια της ανάλυσης, μιας και ο χρόνος άντλησης των δεδομένων από την παραγωγική βάση είναι δευτερεύουσας σημασίας. Επιπλέον έχουμε δυνατότητα αποθήκευσης μεγάλου όγκου δεδομένων στον Server. Ας σημειωθεί ότι το σχήμα Star χρησιμοποιείται κυρίως σε μικρά Data Warehouses λόγω της απλότητας σχεδιασμού του και της ταχύτητας ανάκτησης της πληροφορίας από τους τελικούς χρήστες.

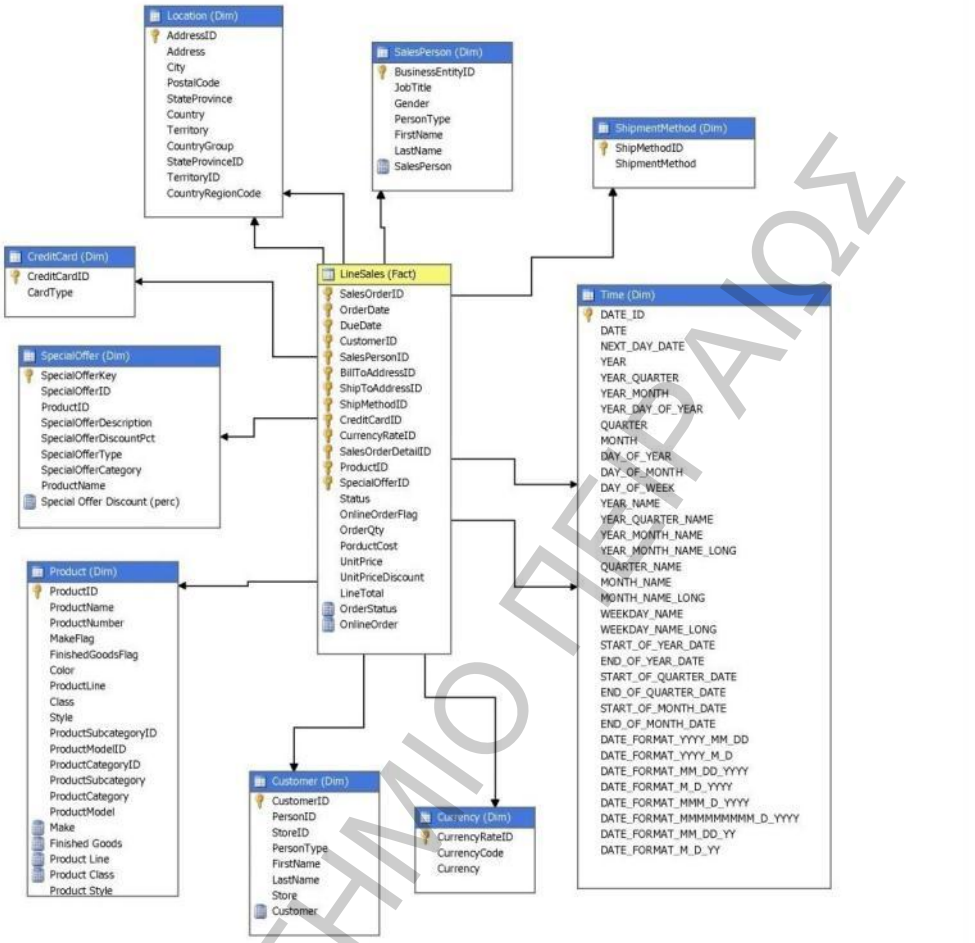
Ο κάθε κύβος έχει το δικό του Fact πίνακα, ο οποίος συνδέεται με τα Dimensions όπως φαίνεται στις Εικόνες 3.1, 3.2 και 3.3 :



Εικόνα 3.1: Κύβος Geo Sales

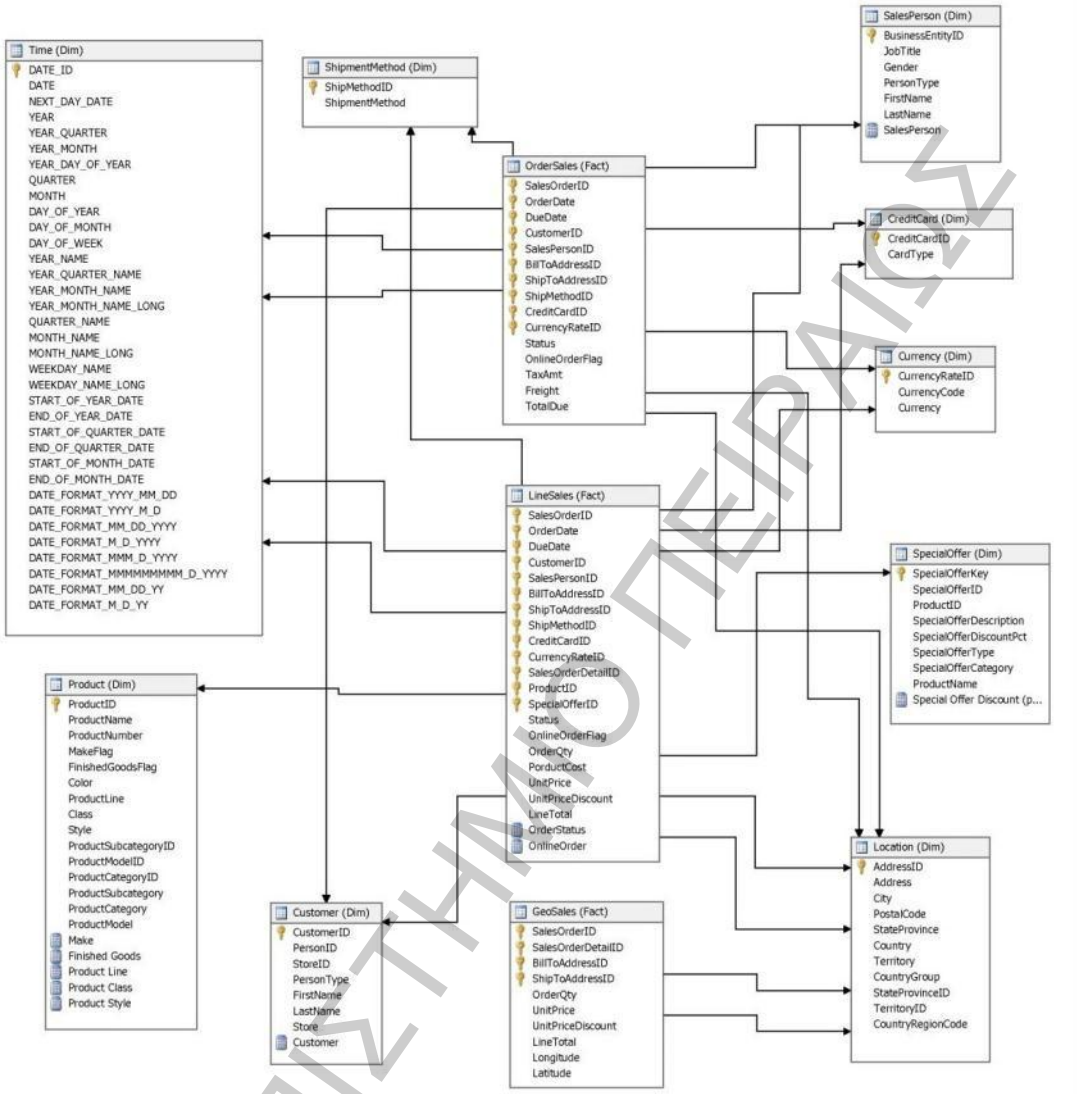


Εικόνα 3.2: Κύβος Order Sales



Εικόνα 3.3: Κύβος Line Sales

Ενώ ο κάθε κύβος έχει τα δικά του Measures και Dimensions, όλα αυτά προέρχονται από τους πίνακες μιας ενιαίας βάσης Data Warehouse. Η βάση αυτή περιέχει 3 Fact πίνακες, (Fact.OrderSales, Fact.LineSales και Fact.GeoSales) οι οποίοι είναι συνδεδεμένοι με τους αντίστοιχους Dimension πίνακες οι οποίοι είναι οι εξής: Dim.CreditCard, Dim.Currency, Dim.Customer, Dim.Location, Dim.Product, Dim.SalesPerson, Dim.SalesReason, Dim.ShipmentMethod, Dim.SpecialOffer, Dim.Time, τα περιεχόμενα των οποίων φαίνονται στην Εικόνα 3.4.



Εικόνα 3.4: Πίνακες Data Warehouse "Adventure Works DW"

Εφόσον έχουμε την ανάλυση των δεδομένων που χρειάζονται, θα πρέπει να βρούμε ποια είναι η πηγή τους. Δηλαδή θα πρέπει να εντοπίσουμε που βρίσκονται τα δεδομένα αυτά στις παραγωγικές βάσεις και έπειτα να ορίσουμε τη μορφή που τα χρειαζόμαστε και να ορίσουμε τι ενέργειες χρειάζονται στη φάση του ETL ώστε τα δεδομένα να ληφθούν στην επιθυμητή μορφή. Η πηγή δεδομένων μας είναι η παραγωγική βάση

της Adventure Works που ονομάζεται AdventureWorks2008R2. Τα πεδία που χρειαζόμαστε με τα αντίστοιχα πεδία της παραγωγικής βάσης βρίσκονται στον παρακάτω πίνακα:

Fact.OrderSales	
[Data Warehouse Field]	**[Table].[Field]**
[SalesOrderID]	[Sales.SalesOrderHeader].[SalesOrderID]
[OrderDate]	[Sales.SalesOrderHeader].[OrderDate]
[DueDate]	[Sales.SalesOrderHeader].[DueDate]
[Status]	[Sales.SalesOrderHeader].[Status]
[OnlineOrderFlag]	[Sales.SalesOrderHeader].[OnlineOrderFlag]
[CustomerID]	[Sales.SalesOrderHeader].[CustomerID]
[SalesPersonID]	[Sales.SalesOrderHeader].[SalesPersonID]
[BillToAddressID]	[Sales.SalesOrderHeader].[BillToAddressID]
[ShipToAddressID]	[Sales.SalesOrderHeader].[ShipToAddressID]
[ShipMethodID]	[Sales.SalesOrderHeader].[ShipMethodID]
[CreditCardID]	[Sales.SalesOrderHeader].[CreditCardID]
[CurrencyRateID]	[Sales.SalesOrderHeader].[CurrencyRateID]
[TaxAmt]	[Sales.SalesOrderHeader].[TaxAmt]
[Freight]	[Sales.SalesOrderHeader].[Freight]
[TotalDue]	[Sales.SalesOrderHeader].[TotalDue]
Fact.LineSales	
[Data Warehouse Field]	**[Table].[Field]**
[SalesOrderID]	[Sales.SalesOrderHeader].[SalesOrderID]
[OrderDate]	[Sales.SalesOrderHeader].[OrderDate]
[DueDate]	[Sales.SalesOrderHeader].[DueDate]
[Status]	[Sales.SalesOrderHeader].[Status]
[OnlineOrderFlag]	[Sales.SalesOrderHeader].[OnlineOrderFlag]
[CustomerID]	[Sales.SalesOrderHeader].[CustomerID]
[SalesPersonID]	[Sales.SalesOrderHeader].[SalesPersonID]
[BillToAddressID]	[Sales.SalesOrderHeader].[BillToAddressID]
[ShipToAddressID]	[Sales.SalesOrderHeader].[ShipToAddressID]
[ShipMethodID]	[Sales.SalesOrderHeader].[ShipMethodID]
[CreditCardID]	[Sales.SalesOrderHeader].[CreditCardID]
[CurrencyRateID]	[Sales.SalesOrderHeader].[CurrencyRateID]
[SalesOrderDetailID]	[Sales.SalesOrderDetail].[SalesOrderDetailID]
[OrderQty]	[Sales.SalesOrderDetail].[OrderQty]
[ProductID]	[Sales.SalesOrderDetail].[ProductID]
[PorductCost]	[Production.Product].[StandardCost]
[SpecialOfferID]	[Sales.SalesOrderDetail].[SpecialOfferID]
[UnitPrice]	[Sales.SalesOrderDetail].[UnitPrice]
[UnitPriceDiscount]	[Sales.SalesOrderDetail].[UnitPriceDiscount]
[LineTotal]	[Sales.SalesOrderDetail].[UnitPriceDiscount]
Fact.GeoSales	
[Data Warehouse Field]	**[Table].[Field]**
[SalesOrderID]	[Sales.SalesOrderHeader].[SalesOrderID]

[SalesOrderDetailID]	[Sales.SalesOrderDetail].[SalesOrderDetailID]
[BillToAddressID]	[Sales.SalesOrderHeader].[BillToAddressID]
[ShipToAddressID]	[Sales.SalesOrderHeader].[ShipToAddressID]
[OrderQty]	[Sales.SalesOrderDetail].[OrderQty]
[UnitPrice]	[Sales.SalesOrderDetail].[UnitPrice]
[UnitPriceDiscount]	[Sales.SalesOrderDetail].[UnitPriceDiscount]
[LineTotal]	[Sales.SalesOrderDetail].[UnitPriceDiscount]
[Longitude]	[Person.Address].[SpatialLocation]
[Latitude]	[Person.Address].[SpatialLocation]
Dim.Time	
[Data Warehouse Field]	**[Table].[Field]**
[DATE_ID] [DATE] [NEXT_DAY_DATE] [YEAR] [YEAR_QUARTER] [YEAR_MONTH] [YEAR_DAY_OF_YEAR] [QUARTER] [MONTH] [DAY_OF_YEAR] [DAY_OF_MONTH] [DAY_OF_WEEK] [YEAR_NAME] [YEAR_QUARTER_NAME] [YEAR_MONTH_NAME] [YEAR_MONTH_NAME_LONG] [QUARTER_NAME] [MONTH_NAME] [MONTH_NAME_LONG] [WEEKDAY_NAME] [WEEKDAY_NAME_LONG] [START_OF_YEAR_DATE] [END_OF_YEAR_DATE] [START_OF_QUARTER_DATE] [END_OF_QUARTER_DATE] [START_OF_MONTH_DATE] [END_OF_MONTH_DATE] [DATE_FORMAT_YYYY_MM_DD] [DATE_FORMAT_YYYY_M_D] [DATE_FORMAT_MM_DD_YYYY] [DATE_FORMAT_M_D_YYYY] [DATE_FORMAT_MMM_D_YYYY] [DATE_FORMAT_MMMMMMMMM_D_YYYY] [DATE_FORMAT_MM_DD_YY] [DATE_FORMAT_M_D_YY]	Τα πεδία αυτού του πίνακα θα δημιουργηθούν με ένα SQL Script το οποίο θα δημιουργεί τα πεδία και θα συμπληρώνει τις αντίστοιχες τιμές για μια χρονική περίοδο που θα ορίζεται στο Script.
Dim.SpecialOffer	
[Data Warehouse Field]	**[Table].[Field]**
[SpecialOfferKey]	Primary Key Autonumber
[SpecialOfferID]	[Sales.SpecialOfferProduct].[SpecialOfferID]

[ProductID]	[Sales.SpecialOfferProduct].[ProductID]
[SpecialOfferDescription]	[Sales.SpecialOffer].[Description]
[SpecialOfferDiscountPct]	[Sales.SpecialOffer].[DiscountPct]
[SpecialOfferType]	[Sales.SpecialOffer].[Type]
[SpecialOfferCategory]	[Sales.SpecialOffer].[Category]
[ProductName]	[Production.Product].[Name]
Dim.ShipmentMethod	
[Data Warehouse Field]	**[Table].[Field]**
[ShipMethodID]	[Purchasing.ShipMethod].[ShipMethodID]
[ShipmentMethod]	[Purchasing.ShipMethod].[Name]
Dim.SalesPerson	
[Data Warehouse Field]	**[Table].[Field]**
[BusinessEntityID]	[Sales.SalesPerson].[BusinessEntityID]
[JobTitle]	[HumanResources.Empleyee].[JobTitle]
[Gender]	[HumanResources.Empleyee].[Gender]
[PersonType]	[Person.Person].[PersonType]
[FirstName]	[Person.Person].[FirstName]
[LastName]	[Person.Person].[LastName]
Dim.Product	
[Data Warehouse Field]	**[Table].[Field]**
[ProductID]	[Production.Product].[ProductID]
[ProductName]	[Production.Product].[Name]
[ProductNumber]	[Production.Product].[ProductNumber]
[MakeFlag]	[Production.Product].[MakeFlag]
[FinishedGoodsFlag]	[Production.Product].[FinishedGoodsFlag]
[Color]	[Production.Product].[Color]
[ProductLine]	[Production.Product].[ProductLine]
[Class]	[Production.Product].[Class]
[Style]	[Production.Product].[Style]
[ProductSubcategoryID]	[Production.Product].[ProductSubcategoryID]
[ProductModelID]	[Production.Product].[ProductModelID]
[ProductCategoryID]	[Production.ProductCategory].[ProductCategoryID]
[ProductSubcategory]	[Production.ProductSubcategory].[Name]
[ProductCategory]	[Production.ProductCategory].[Name]
[ProductModel]	[Production.ProductModel].[Name]
Dim.Location	
[Data Warehouse Field]	**[Table].[Field]**
[AddressID]	[Person.Address].[AddressID]
[Address]	[Person.Address].[AddressLine]
[City]	[Person.Address].[City]
[PostalCode]	[Person.Address].[PostalCode]
[StateProvince]	[Person.StateProvince].[Name]
[Country]	[Person.CountryRegion].[CountryRegionName]
[Territory]	[Sales.SalesTerritory].[Name]
[CountryGroup]	[Sales.SalesTerritory].[Group]
[StateProvinceID]	[Person.Address].[StateProvinceID]
[TerritoryID]	[Person.StateProvince].[TterritoryID]
[CountryRegionCode]	[Person.StateProvince].[CountryRegionCode]

Dim.Customer	
[Data Warehouse Field]	**[Table].[Field]**
[CustomerID]	[Sales.Customer].[CustmerID]
[PersonID]	[Sales.Customer].[PersonID]
[StoreID]	[Sales.Customer].[StoreID]
[PersonType]	[Person.Person].[PersonType]
[FirstName]	[Person.Person].[FirstName]
[LastName]	[Person.Person].[LastName]
[Store]	[Sales.Store].[Name]
Dim.Currency	
[Data Warehouse Field]	**[Table].[Field]**
[CurrencyRateID]	[Sales.CurrencyRate].[CurrencyRateID]
[CurrencyCode]	[Sales.Currency].[CurrencyCode]
[Currency]	[Sales.Currency].[Name]
Dim.CreditCard	
[Data Warehouse Field]	**[Table].[Field]**
[CreditCardID]	[Sales.CreditCard].[CreditCartID]
[CardType]	[Sales.CreditCard].[CardType]

Εφόσον έχουν εντοπιστεί οι συσχετισμοί των επιθυμητών πεδίων με τα αντίστοιχα πεδία στη παραγωγική OLTP βάση της Adventure Works, θα πρέπει να οριστεί η επεξεργασία και ο καθαρισμός που θα πραγματοποιηθεί ώστε τα δεδομένα να έχουν της κατάλληλη μορφή. Αυτή η διαδικασία θα παρουσιαστεί διεξοδικά στην επόμενη ενότητα. Αναφέρεται ότι παρεμβάλλουμε ένα στάδιο μεταξύ της μεταφοράς των δεδομένων από την OLTP βάση στη βάση Data Warehouse. Αυτό το στάδιο ονομάζεται Staging και είναι ουσιαστικά η δημιουργία μιας ενδιάμεσης βάσης δεδομένων που περιέχει μόνο τους πίνακες και τα πεδία τα οποία χρειαζόμαστε από την παραγωγική βάση. Ουσιαστικά είναι ένα bulk load των δεδομένων από την παραγωγική βάση σε μια ενδιάμεση και γίνεται κυρίως γιατί η αντιγραφή αυτούσιων των δεδομένων της παραγωγικής βάσης είναι μια σχετικά γρήγορη διαδικασία και έτσι δεν απασχολείται η κεντρική βάση δεδομένων για αρκετό διάστημα. Έπειτα, αποδεσμεύουμε και δεν επιβαρύνουμε την παραγωγική βάση μιας και η διαδικασία

επεξεργασίας και μετασχηματισμού των δεδομένων γίνεται πλέον στην staging database. Άλλο ένα πλεονέκτημα είναι ότι υπάρχει ένα αντίγραφο ασφαλείας των δεδομένων που χρειαζόμαστε το οποίο είναι διαθέσιμο όποτε χρειαστεί και επιπλέον η staging βάση δεδομένων είναι μια βάση που συγκεντρώνει τα δεδομένα που χρειαζόμαστε από πολλές και διαφορετικές βάσεις δεδομένων.

Ανάλυση Web Site Business Intelligence Application.

Το BI Web Site της εταιρίας Adventure Works είναι το μέσο για να έχουν οι εργαζόμενοι της εταιρίας πρόσβαση στα δεδομένα του Data Warehouse. Οι απαιτήσεις είναι οι εξής:

- Pivot Table μέσω του οποίου θα υπάρχει πρόσβαση στα δεδομένα του Data Warehouse μέσω ενός δυναμικού πίνακα.
- Pivot Chart όπου η οπτικοποίηση των δεδομένων γίνεται μέσω διαγραμμάτων.
- Χάρτης στο οποίο θα εμφανίζονται τα αριθμητικά δεδομένα σε σχέση με χωρικές τοποθεσίες (κύβος GeoSales).
- Μια σελίδα ρυθμίσεων όπου θα ορίζονται κάποιες λεπτομέρειες όπως η διεύθυνση σύνδεσης με τον Analysis Server κ.α.

3.3. Δημιουργία Data Warehouse (SSIS Project)

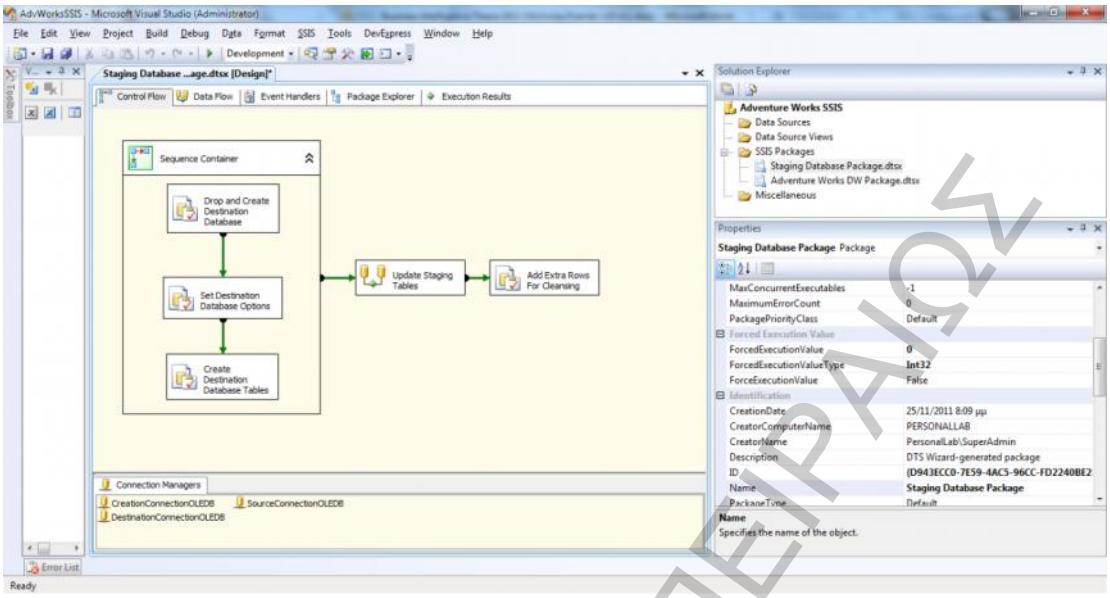
Εφόσον έχει ολοκληρωθεί η ανάλυση των απαιτήσεων και έχουν εντοπιστεί οι αντιστοιχήσεις των επιθυμητών πεδίων με τα πεδία της παραγωγικής βάσης δεδομένων η διαδικασία της δημιουργίας του Data Warehouse μπορεί να ξεκινήσει.

Staging Database

Το πρώτο βήμα, όπως αναφέρθηκε και στην προηγούμενη ενότητα, είναι η δημιουργία μιας ενδιάμεσης βάσης δεδομένων η οποία ονομάζεται Staging και η οποία θα περιέχει τους πίνακες και τα πεδία που χρειαζόμαστε για τη δημιουργία του Data Warehouse. Ουσιαστικά υλοποιείται το μέρος Extract της διαδικασίας Extract-Transform- Load (ETL) που απαιτείται για τη δημιουργία ενός Data Warehouse.

Πρέπει να αναφερθεί ότι προαπαιτούμενο είναι η εγκατάσταση της παραγωγικής OLTP βάσης δεδομένων AdventureWorks2008R2 η οποία βρίσκεται στο (30) και παρέχεται στα παραδοτέα αρχεία της εργασίας.

Για την δημιουργία του Data Warehouse χρησιμοποιούμε το Microsoft SQL Server Business Intelligence Studio δημιουργώντας ένα Integration Services Project με το όνομα "Adventure Works SSIS". Έπειτα δημιουργούμε ένα SSIS Package με το όνομα "Staging Database Package.dtsx". Σε αυτό το Package θα οριστούν οι ενέργειες για τη δημιουργία της Staging βάσης δεδομένων.



Εικόνα 3.5: SSIS Project Adventure Works

Τα βήματα που ακολουθήθηκαν είναι τα εξής:

Βήμα 1: Δημιουργήθηκαν 3 Connection στον Connection Manager:

1. Το "CreationConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στον SQL Server που θα αποθηκευθεί η staging βάση δεδομένων. Δεν συνδεόμαστε σε κάποια συγκεκριμένη βάση δεδομένων (χρησιμοποιείται για να γίνουν κάποιες αρχικές διεργασίες στον SQL Server). Για το συγκεκριμένο Project ο SQL Server βρίσκεται στο localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.
2. Το "SourceConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στην παραγωγική βάση AdventureWorks2008R2 για να αντλήσουμε τα δεδομένα. Για το συγκεκριμένο Project η βάση

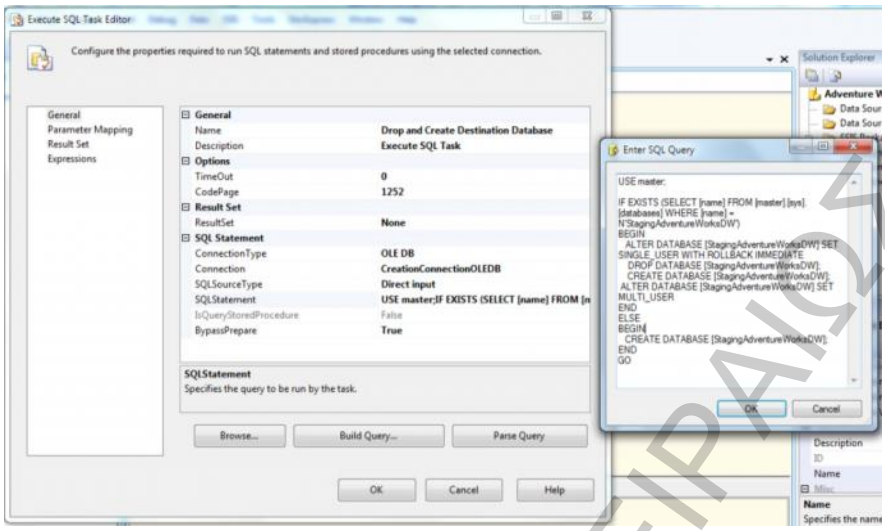
βρίσκεται στο Localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.

3. Το "DestinationConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στη staging βάση StagingAdventureWorksDW στην οποία θα μεταφερθούν τα δεδομένα. Για το συγκεκριμένο Project η βάση βρίσκεται στο localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.

Βήμα 2: Δημιουργήθηκε ένας Sequence Container στον οποίο τοποθετήθηκαν 3 SQL Tasks τα οποία συνδέονται μεταξύ τους ώστε να εκτελεστούν σειριακά το ένα μετά το άλλο:

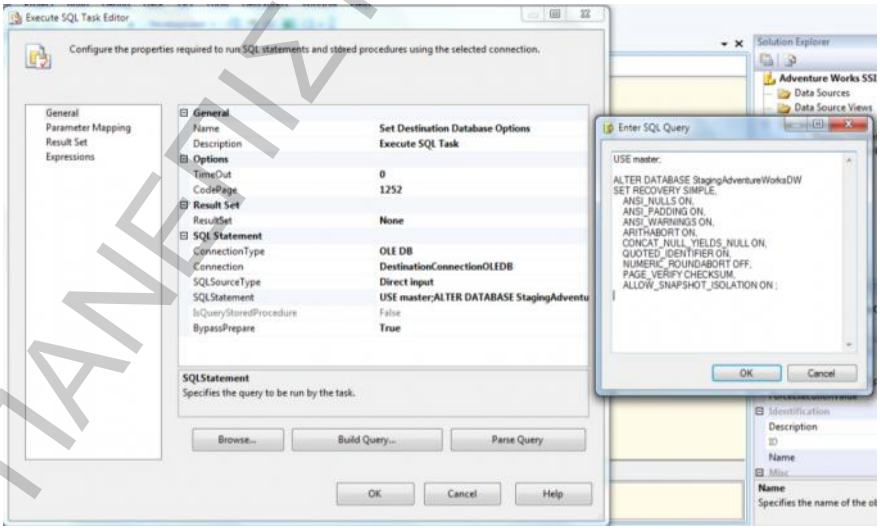
- "Drop and Create Destination Database".
- "Set Destination Database Options".
- "Create Destination Database Tables".

Στο "Drop and Create Destination Database" Task συνδεόμαστε μέσω του CreationConnectionOLEDB στον SQL Server που θα τοποθετηθεί η Staging βάση και μέσω ενός SQL Task " (Εικόνα 3.6) βλέπουμε αν υπάρχει η βάση StagingAdventureWorksDW και αν ναι τη διαγράφουμε (drop) και τη ξαναδημιουργούμε, έτσι ώστε να συμπεριλάβει τα νέα στοιχεία.



Εικόνα 3.6: Εντολές SQL Drop and Create Destination Database

Εφόσον ολοκληρώθηκε η δημιουργία της βάσης StagingAdventureWorksDW, εκτελείται το Task "Set Destination Database Options" το οποίο, μέσω του DestinationConnectionOLDB πλέον αφού υπάρχει η βάση, εκτελεί κάποιες απαραίτητες εντολές ώστε να αρχικοποιηθεί η βάση δεδομένων. " (Εικόνα 3.7)



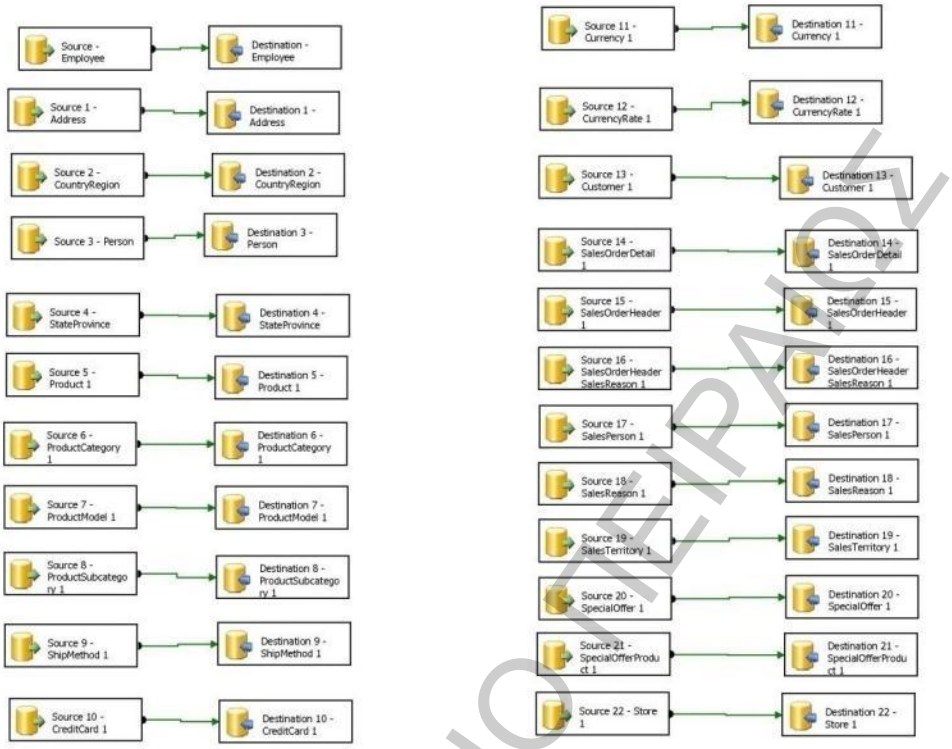
Εικόνα 3.7: Εντολές SQL Set Destination Database Options

Τέλος, εκτελείται το τελευταίο Task " Create Destination Database Tables" το οποίο, μέσω του DestinationConnectionOLDB, εκτελεί το SQL Script που δημιουργεί τους πίνακες και τα πεδία της βάσης StagingAdventureWorksDW. Ο κώδικας που εκτελείται είναι ο εξής:

```
IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'HumanResources')
BEGIN
EXEC(N'CREATE SCHEMA [HumanResources]')
END
CREATE TABLE [HumanResources].[Employee] ( [BusinessEntityID] int NOT NULL, [JobTitle] nvarchar(50) NOT NULL, [Gender]
nchar(1) NOT NULL, PRIMARY KEY (BusinessEntityID) ) GO


IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Person')
BEGIN
EXEC(N'CREATE SCHEMA [Person]')
END
CREATE TABLE [Person].[Address] ( [AddressID] int NOT NULL, [AddressLine1] nvarchar(60) NOT NULL, [City] nvarchar(30)
NOT NULL, [StateProvinceID] int NOT NULL, [PostalCode] nvarchar(15) NOT NULL, [SpatialLocation] geography, [Longitude]
float, [Latitude] float, PRIMARY KEY (AddressID) ) GO

CREATE TABLE [Person].[CountryRegion] ( [CountryRegionCode] nvarchar(3) NOT NULL, [Name] nvarchar(50) NOT NULL,
PRIMARY KEY (CountryRegionCode) )GO

CREATE TABLE [Person].[Person] ([BusinessEntityID] int NOT NULL,[PersonType] nchar(2) NOT NULL,[FirstName] nvarchar(50)
NOT NULL,[LastName] nvarchar(50) NOT NULL,PRIMARY KEY (BusinessEntityID))GO

CREATE TABLE [Person].[StateProvince] ([StateProvinceID] int NOT NULL,[CountryRegionCode] nvarchar(3) NOT NULL,[Name]
nvarchar(50) NOT NULL,[TerritoryID] int NOT NULL,PRIMARY KEY (StateProvinceID))GO

IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Production')
BEGIN
EXEC(N'CREATE SCHEMA [Production]')
END

CREATE TABLE [Production].[Product] ([ProductID] int NOT NULL,[Name] nvarchar(50) NOT NULL,[ProductNumber]
nvarchar(25) NOT NULL,[MakeFlag] bit NOT NULL,[FinishedGoodsFlag] bit NOT NULL,[StandardCost] money NOT
NULL,[Color] nvarchar(15),[ProductLine] nchar(2),[Class] nchar(2),[Style] nchar(2),[ProductSubcategoryID] int,[ProductModelID]
int,PRIMARY KEY (ProductID))GO

CREATE TABLE [Production].[ProductCategory] ([ProductCategoryID] int NOT NULL,[Name] nvarchar(50) NOT NULL,PRIMARY
KEY (ProductCategoryID))GO

CREATE TABLE [Production].[ProductModel] ([ProductModelID] int NOT NULL,[Name] nvarchar(50) NOT NULL,PRIMARY KEY
(ProductModelID))GO

CREATE TABLE [Production].[ProductSubcategory] ([ProductSubcategoryID] int NOT NULL,[ProductCategoryID] int NOT
NULL,[Name] nvarchar(50) NOT NULL,PRIMARY KEY (ProductSubcategoryID))GO

IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Purchasing')
BEGIN
EXEC(N'CREATE SCHEMA [Purchasing]')
END
```

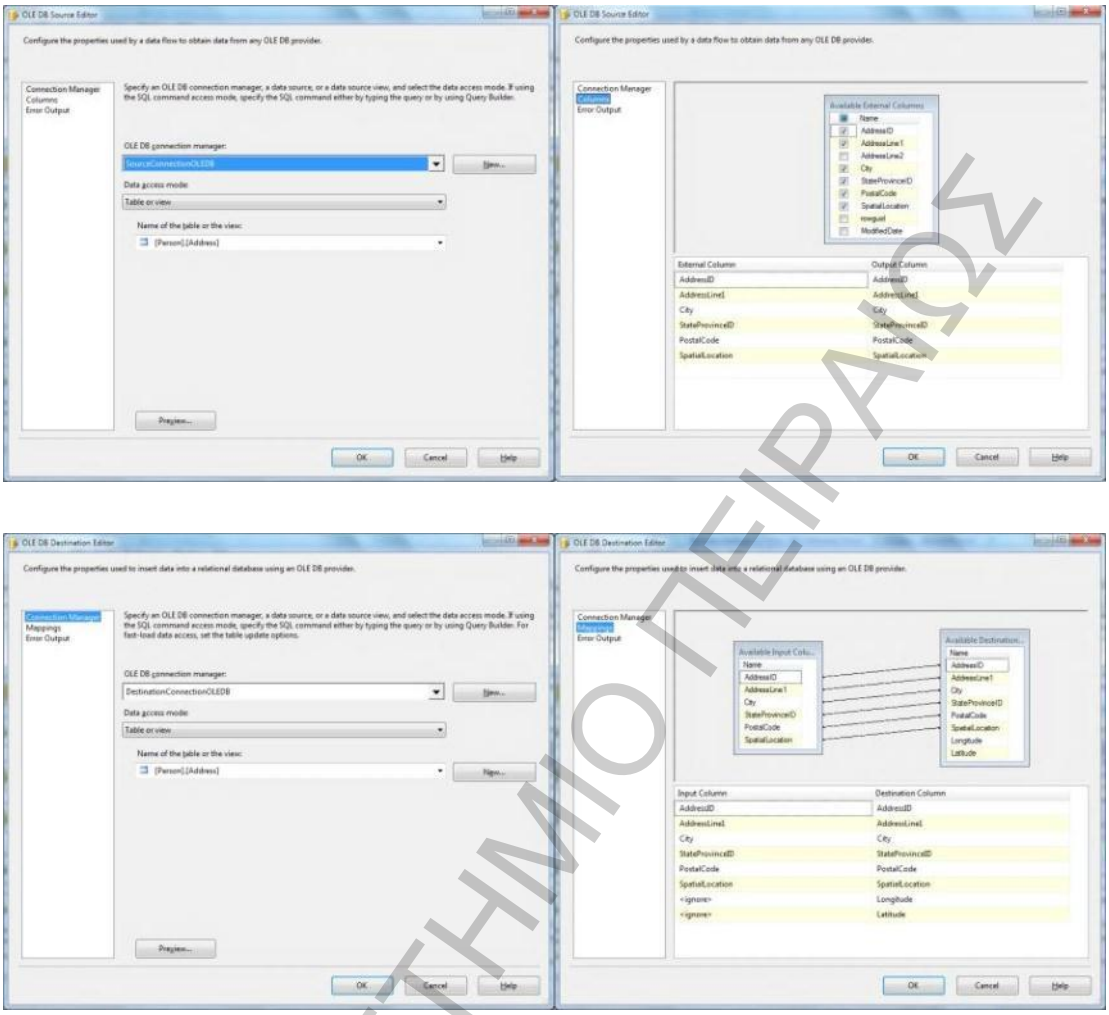
```
CREATE TABLE [Purchasing].[ShipMethod] ([ShipMethodID] int NOT NULL,[Name] nvarchar(50) NOT NULL,PRIMARY KEY
(ShipMethodID))GO


IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Sales')
BEGIN
EXEC(N'CREATE SCHEMA [Sales]')
END

CREATE TABLE [Sales].[CreditCard] ([CreditCardID] int NOT NULL,[CardType] nvarchar(50) NOT NULL,PRIMARY KEY
(CreditCardID))GO

CREATE TABLE [Sales].[Currency] ([CurrencyCode] nchar(3) NOT NULL,[Name] nvarchar(50) NOT NULL,PRIMARY KEY
(CurrencyCode))GO

CREATE TABLE [Sales].[CurrencyRate] ([CurrencyRateID] int NOT NULL,[ToCurrencyCode] nchar(3) NOT NULL,
PRIMARY KEY (CurrencyRateID))GO

CREATE TABLE [Sales].[Customer] ([CustomerID] int NOT NULL,[PersonID] int,[StoreID] int,PRIMARY KEY (CustomerID))GO

CREATE TABLE [Sales].[SalesOrderDetail] ([SalesOrderDetailID] int NOT NULL,[SalesOrderID] int NOT NULL,[OrderQty] smallint
NOT NULL,[ProductID] int NOT NULL,[SpecialOfferID] int NOT NULL,[UnitPrice] money NOT NULL,[UnitPriceDiscount] money
NOT NULL,[LineTotal] numeric(38,6) NOT NULL,PRIMARY KEY (SalesOrderDetailID,SalesOrderID))GO

CREATE TABLE [Sales].[SalesOrderHeader] ([SalesOrderID] int NOT NULL,[OrderDate] datetime NOT NULL,
[DueDate] datetime NOT NULL,[ShipDate] datetime,[Status] tinyint NOT NULL,[OnlineOrderFlag] bit NOT NULL,
[AccountNumber] nvarchar(15),[CustomerID] int NOT NULL,[SalesPersonID] int,[TerritoryID] int,[BillToAddressID] int NOT
NULL,[ShipToAddressID] int NOT NULL,[ShipMethodID] int NOT NULL,[CreditCardID] int,[CurrencyRateID] int,[TaxAmt]
money NOT NULL,[Freight] money NOT NULL,[TotalDue] money NOT NULL,PRIMARY KEY (SalesOrderID))GO

CREATE TABLE [Sales].[SalesOrderHeaderSalesReason] ([SalesOrderID] int NOT NULL,[SalesReasonID] int NOT NULL,PRIMARY
KEY (SalesReasonID,SalesOrderID))GO

CREATE TABLE [Sales].[SalesPerson] ([BusinessEntityID] int NOT NULL,PRIMARY KEY (BusinessEntityID))GO

CREATE TABLE [Sales].[SalesReason] ([SalesReasonID] int NOT NULL,[Name] nvarchar(50) NOT NULL,[ReasonType] nvarchar(50)
NOT NULL,PRIMARY KEY (SalesReasonID))GO

CREATE TABLE [Sales].[SalesTerritory] ([TerritoryID] int NOT NULL,[Name] nvarchar(50) NOT NULL,[Group] nvarchar(50) NOT
NULL,PRIMARY KEY (TerritoryID))GO

CREATE TABLE [Sales].[SpecialOffer] ([SpecialOfferID] int NOT NULL,[Description] nvarchar(255) NOT NULL,[DiscountPct]
smallmoney NOT NULL,[Type] nvarchar(50) NOT NULL,[Category] nvarchar(50) NOT NULL,PRIMARY KEY (SpecialOfferID))GO

CREATE TABLE [Sales].[SpecialOfferProduct] ([SpecialOfferID] int NOT NULL,[ProductID] int NOT NULL,PRIMARY KEY
(ProductID,SpecialOfferID))GO

CREATE TABLE [Sales].[Store] ([BusinessEntityID] int NOT NULL,[Name] nvarchar(50) NOT NULL,PRIMARY KEY
(BusinessEntityID))GO
```


Βήμα 3: Ο Sequence Container συνδέεται σειριακά με ένα Data Flow Task το οποίο ονομάζεται " Update Staging Tables" και το οποίο έχει Data Flow Components που μεταφέρουν τις εγγραφές των επιθυμητών πινάκων με τα επιθυμητά πεδία από την AdventureWorks2008R2 (source database) στην StagingAdventureWorksDW (destination database) (σχήμα).



Εικόνα 3.8: Update Staging Tables Data Flow Task

Ένα παράδειγμα είναι η μεταφορά του πίνακα Person.Address. Στην Εικόνα 3.9 φαίνεται ο πίνακας και τα πεδία που επιλέχθηκαν από την Source Database καθώς και ο πίνακας προορισμού και η αντιστοίχηση των πεδίων.



Εικόνα 3.9: Δημιουργία Πίνακα Person.Address

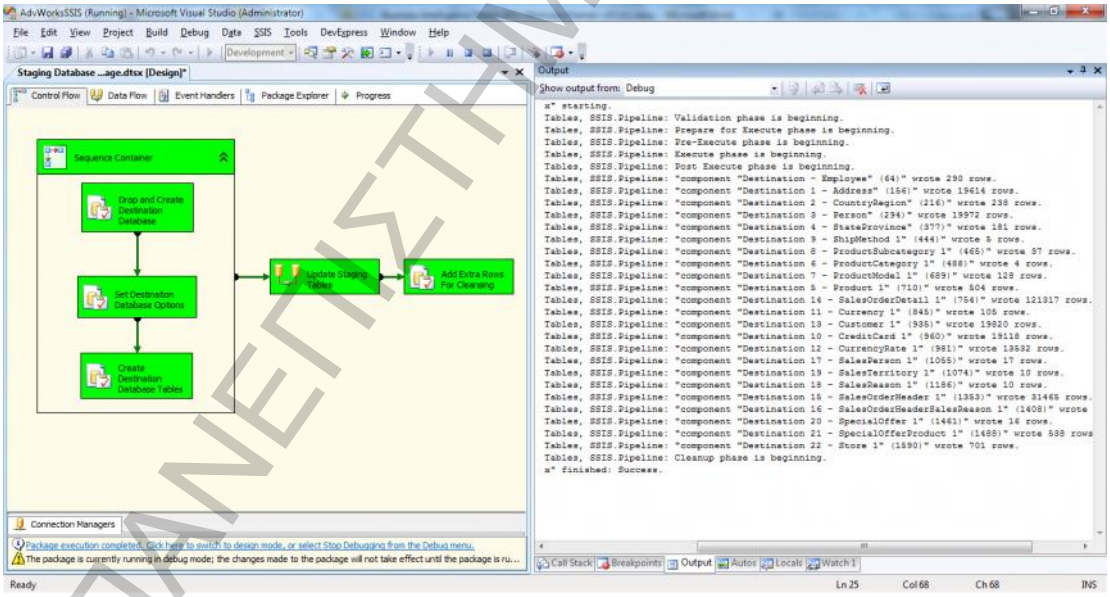
Βήμα 4: Το βήμα αυτό εκτελείται τελευταίο και περιλαμβάνει το SQL Task "Add Extra Rows For Cleansing" και εκτελεί το ακόλουθο SQL Script στη Staging βάση δεδομένων:

```
UPDATE [Person].[Address]
SET Longitude = SpatialLocation.Long, Latitude = SpatialLocation.Lat
GO
INSERT INTO [Sales].[CurrencyRate]
VALUES (0,'NoC')
GO
INSERT INTO [Sales].[Currency]
VALUES ('NoC', 'No Currency')
GO
INSERT INTO [Sales].[SalesReason]
VALUES (0,'No Reason', 'Other')
```

```
GO
INSERT INTO [Sales].[SalesPerson]
VALUES (0)
GO
INSERT INTO [Person].[Person]
VALUES (0,'NO','NoPerson','NoPerson')
GO
INSERT INTO StagingAdventureWorksDW.HumanResources.Employee
VALUES (0,'No Job Title','M')
GO
INSERT INTO [Sales].[CreditCard]
VALUES (0,'NoCreditCard')
GO
```

Ουσιαστικά ορίζει δυο πεδία στον πίνακα [Person.Address] που είναι το Longitude και Latitude και ορίζει τις τιμές τους από το ειδικό spatial πεδίο SpatialLocation. Οι υπόλοιπες εντολές αντικαθιστούν τις κενές εγγραφές με μια καθορισμένη εγγραφή ώστε να μην εμφανίζονται κενές εγγραφές στο Data Warehouse.

Με την εκτέλεση του συγκεκριμένου Package ολοκληρώνεται η δημιουργία της ενδιάμεσης βάσης δεδομένων (Εικόνα 3.10).



Εικόνα 3.10: Ολοκλήρωση Δημιουργίας Staging Database

Data Warehouse Database

Η ενδιάμεση (staging) βάση δεδομένων που περιέχει όλα τα δεδομένα που χρειαζόμαστε για τη δημιουργία του Data Warehouse έχει δημιουργηθεί. Για τη δημιουργία της Data Warehouse βάσης δεδομένων δημιουργούμε ένα επιπλέον SSIS Package το οποίο ονομάζεται "Adventure Works DW Package.dtsx". Σε αυτό το package υλοποιείται το μέρος Load-Transform και έτσι ολοκληρώνεται η διαδικασία Extract -Transform-Load που απαιτείται για τη δημιουργία ενός Data Warehouse.

Τα βήματα που ακολουθήθηκαν για την δημιουργία της Data Warehouse βάσης δεδομένων είναι τα εξής:

Βήμα 1: Δημιουργήθηκαν 3 Connection στον Connection Managers:

1. "CreationConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στον SQL Server που θα αποθηκευθεί η data warehouse βάση δεδομένων και χρησιμοποιείται για να γίνουν κάποιες αρχικές διεργασίες στον SQL Server. Για το συγκεκριμένο Project ο SQL Server βρίσκεται στο localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.
2. "SourceConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στην staging βάση StagingAdventureWorksDW για να αντλήσουμε τα δεδομένα. Για το συγκεκριμένο Project η βάση βρίσκεται στο Localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.

3. "DestinationConnectionOLEDB": Είναι ο Manager μέσω του οποίου συνδεόμαστε στη Data Warehouse βάση AdventureWorksDW. Για το συγκεκριμένο Project η βάση βρίσκεται στο localhost και τα στοιχεία σύνδεσης είναι Username: sa και Password:123.

Βήμα 2: Δημιουργήθηκε ένας Sequence Container στον οποίο τοποθετήθηκαν 3 SQL Tasks τα οποία συνδέονται μεταξύ τους ώστε να εκτελεστούν σειριακά το ένα μετά το άλλο:

- "Drop and Create DW Database".
- "Set Destination Database Options".
- "Create Destination Database Tables".

Και τα τρία SQL Tasks έχουν ακριβώς την ίδια λειτουργικότητα με τα αντίστοιχά τους στο Staging Package με τη διαφορά ότι εφαρμόζονται στην Data Warehouse βάση δεδομένων AdventureWorksDW και όχι στην Staging. Ο SQL κώδικας που εκτελείται στην "Create Destination Database Tables" είναι αυτός που δημιουργεί τους πίνακες Dim και Fact καθώς και τα πεδία και τα κλειδιά τους και είναι ο εξής:

```
IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Dim')
BEGIN
EXEC(N'CREATE SCHEMA [Dim]')
END

CREATE TABLE [Dim].[Location] ([AddressID] int NOT NULL,[Address] nvarchar(60) NOT NULL,[City] nvarchar(30) NOT
NULL,[PostalCode] nvarchar(15) NOT NULL,[StateProvince] nvarchar(50) NOT NULL,[Country] nvarchar(50) NOT
NULL,[Territory] nvarchar(50) NOT NULL,[CountryGroup] nvarchar(50) NOT NULL,[StateProvinceID] int NOT
NULL,[TerritoryID] int NOT NULL,[CountryRegionCode] nvarchar(3) NOT NULL,PRIMARY KEY (AddressID))GO

CREATE TABLE [Dim].[Product] ([ProductID] int NOT NULL,[ProductName] nvarchar(50) NOT NULL,[ProductNumber]
varchar(25) NOT NULL,[MakeFlag] bit NOT NULL,[FinishedGoodsFlag] bit NOT NULL,[Color] nvarchar(15),[ProductLine]
nchar(2),[Class] nchar(2),[Style] nchar(2),[ProductSubcategoryID] int,[ProductModelID] int,[ProductCategoryID]
int,[ProductSubcategory] nvarchar(50),[ProductCategory] nvarchar(50),[ProductModel] nvarchar(50),PRIMARY KEY
(ProductID))GO

CREATE TABLE [Dim].[ShipmentMethod] ([ShipMethodID] int NOT NULL,[ShipmentMethod] nvarchar(50) NOT NULL,PRIMARY
KEY (ShipMethodID))GO
```

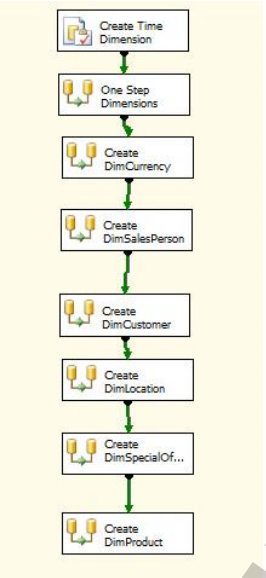
```
CREATE TABLE [Dim].[CreditCard] ([CreditCardID] int NOT NULL,[CardType] nvarchar(50) NOT NULL,PRIMARY KEY (CreditCardID))GO

CREATE TABLE [Dim].[Currency] ([CurrencyRateID] int NOT NULL,[CurrencyCode] nchar(3) NOT NULL,[Currency] nvarchar(50) NOT NULL,PRIMARY KEY (CurrencyRateID))GO

CREATE TABLE [Dim].[Customer] ([CustomerID] int NOT NULL,[PersonID] int,[StoreID] int,[PersonType] nchar(2),[FirstName] nvarchar(50),[LastName] nvarchar(50),[Store] nvarchar(50),PRIMARY KEY (CustomerID))GO

CREATE TABLE [Dim].[SalesPerson] ([BusinessEntityID] int NOT NULL,[JobTitle] nvarchar(50) NOT NULL,[Gender] nchar(1) NOT NULL,[PersonType] nchar(2) NOT NULL,[FirstName] nvarchar(50) NOT NULL,[LastName] nvarchar(50) NOT NULL,PRIMARY KEY (BusinessEntityID))GO

CREATE TABLE [Dim].[SpecialOffer] ([SpecialOfferKey] int NOT NULL IDENTITY,[SpecialOfferID] int NOT NULL,[ProductID] int NOT NULL,[SpecialOfferDescription] nvarchar(255) NOT NULL,[SpecialOfferDiscountPct] smallmoney NOT NULL,[SpecialOfferType] nvarchar(50) NOT NULL,[SpecialOfferCategory] nvarchar(50) NOT NULL,[ProductName] nvarchar(50) NOT NULL,PRIMARY KEY (SpecialOfferKey))GO

IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name = N'Fact')
BEGIN
EXEC(N'CREATE SCHEMA [Fact]')
END

CREATE TABLE [Fact].[LineSales] ([SalesOrderID] int NOT NULL,[OrderDate] datetime NOT NULL,[DueDate] datetime NOT NULL,[Status] tinyint NOT NULL,[OnlineOrderFlag] bit NOT NULL,[CustomerID] int NOT NULL,[SalesPersonID] int NOT NULL,[BillToAddressID] int NOT NULL,[ShipToAddressID] int NOT NULL,[ShipMethodID] int NOT NULL,[CreditCardID] int NOT NULL,[CurrencyRateID] int NOT NULL,[SalesOrderDetailID] int NOT NULL,[OrderQty] smallint NOT NULL,[ProductID] int NOT NULL,[PorductCost] money NOT NULL,[SpecialOfferID] int NOT NULL,[UnitPrice] money NOT NULL,[UnitPriceDiscount] money NOT NULL,[LineTotal] numeric(38,6) NOT NULL,PRIMARY KEY ([SalesOrderID],[OrderDate],[DueDate],[CustomerID],[SalesPersonID],[CreditCardID],[CurrencyRateID],[BillToAddressID],[ShipToAddressID],[ShipMethodID],[SalesOrderDetailID],[ProductID],[SpecialOfferID]))GO

CREATE TABLE [Fact].[OrderSales] ([SalesOrderID] int NOT NULL,[OrderDate] datetime NOT NULL,[DueDate] datetime NOT NULL,[Status] tinyint NOT NULL,[OnlineOrderFlag] bit NOT NULL,[CustomerID] int NOT NULL,[SalesPersonID] int NOT NULL,[BillToAddressID] int NOT NULL,[ShipToAddressID] int NOT NULL,[ShipMethodID] int NOT NULL,[CreditCardID] int NOT NULL,[CurrencyRateID] int NOT NULL,[TaxAmt] money NOT NULL,[Freight] money NOT NULL,[TotalDue] money NOT NULL,PRIMARY KEY ([SalesOrderID],[SalesPersonID],[CreditCardID],[CurrencyRateID],[OrderDate],[DueDate],[CustomerID],[BillToAddressID],[ShipToAddressID],[ShipMethodID]))GO

CREATE TABLE [Fact].[GeoSales] ([SalesOrderID] int NOT NULL,[SalesOrderDetailID] int NOT NULL,[BillToAddressID] int NOT NULL,[ShipToAddressID] int NOT NULL,[OrderQty] smallint NOT NULL,[UnitPrice] money NOT NULL,[UnitPriceDiscount] money NOT NULL,[LineTotal] numeric(38,6) NOT NULL,[Longitude] float,[Latitude] float,PRIMARY KEY ([SalesOrderID],[BillToAddressID],[ShipToAddressID],[SalesOrderDetailID]))GO
```

Βήμα 3: Δημιουργήθηκε ένας Sequence Container ο οποίος συνδέεται σειριακά με τον αντίστοιχο Container του βήματος 2 και ο οποίος περιέχει τα Data Flow Tasks που δημιουργούν όλους τους Dimension πίνακες του Data Warehouse.

Στο σημείο αυτό καλό είναι να γίνει μια μνεία στην διαφορά του SnowFlake και του Star σχήματος, βλέποντας το από μια τεχνική οπτική. Το σχήμα του Data Warehouse επιλέχθηκε να είναι Star και αυτό σημαίνει

ότι το κάθε Dimension έχει ένα μοναδικό πίνακα στη βάση. Αν τα πεδία του κάθε Dimension πίνακα προέρχονται από περισσότερους του ενός πίνακες της staging βάσης, τότε πρέπει να γίνει συγχώνευση(join) των πινάκων αυτών για να συγκεντρωθούν τα πεδία (και οι αντίστοιχες εγγραφές) στον τελικό Dimension πίνακα. Τα Joins αυτά δημιουργούν πολλές διπλότυπες εγγραφές που καταλαμβάνουν μεγαλύτερο χώρο και καθυστερούν τη διαδικασία της άντλησης των δεδομένων. Το πλεονέκτημα όμως είναι ότι κατά τη διάρκεια της ανάλυσης, οι εγγραφές αυτές υπάρχουν ήδη στην DW βάση, επομένως η ταχύτητα ανάκτησής τους από τον τελικό χρήστη είναι πολύ μεγάλη. Αν είχε επιλεγεί το σχήμα Snowflake, τότε δε θα χρειάζονταν αυτά τα Joins στη διάρκεια του Integration, γιατί θα επιτρέπονταν το κάθε Dimension να έχει πολλούς συνδεδεμένους πίνακες (σχεσιακές βάσεις). Αυτό έχει ως πλεονέκτημα την επιτάχυνση της διαδικασίας του Integration καθώς και την εξοικονόμηση χώρου στη βάση (επιτυγχάνεται από την αποφυγή διπλότυπων). Το μειονέκτημα είναι ότι τα Joins των πινάκων που αποτελούν ένα Dimensions ή ένα Fact θα πραγματοποιούνταν κατά την διάρκεια της ανάλυσης των δεδομένων του Data Warehouse, επιβαρύνοντας το σύστημα και καθυστερώντας τον χρήστη κατά την διάρκεια της ανάλυσης.

Τα στοιχεία του Sequence Container που δημιουργήθηκε φαίνονται στην Εικόνα 3.11. Θα γίνει μια ανάλυση των μερών που απαρτίζουν το συγκεκριμένο Sequence γιατί με αυτό τον τρόπο θα περιγραφούν οι Μετασχηματισμοί που γίνονται για να δημιουργηθούν τα Dimensions του Data Warehouse.



Εικόνα 3.11: Sequence Container δημιουργίας των Dimensions Tables.

Create Time Dimension: Ο πίνακας της διάστασης Time θα πρέπει να περιέχει όλες τις χρονικές τιμές που χρειαζόμαστε και διαφορετικές μορφές. Για την ανάγκη της δημιουργίας του συγκεκριμένου πίνακα χρησιμοποιήθηκε ένα SQL Script (31) το οποίο δέχεται σαν παραμέτρους την αρχική ημερομηνία και την τελική ημερομηνία (για τις ανάγκες του Project έχουν δοθεί οι ημερομηνίες από 01/01/1990 έως 01/01/2013) και δημιουργεί μια εγγραφή για κάθε ημέρα του συγκεκριμένης χρονικής περιόδου. Το SQL Script που εκτελείται είναι το εξής:

```
IF not EXISTS(SELECT name FROM sys.tables WHERE name = 'Dim.Time')
CREATE TABLE [Dim].[Time]([DATE_ID] [datetime] not null primary key clustered, [DATE] [date] not null, [NEXT_DAY_DATE]
[datetime]  not null, [YEAR] [smallint]  not null, [YEAR_QUARTER] [int] not null, [YEAR_MONTH] [int] not null,
[YEAR_DAY_OF_YEAR] [int] not null, [QUARTER] [tinyint] not null, [MONTH] [tinyint] not null, [DAY_OF_YEAR] [smallint] not
null, [DAY_OF_MONTH] [smallint] not null, [DAY_OF_WEEK] [tinyint] not null, [YEAR_NAME] [varchar] (4) not null,
[YEAR_QUARTER_NAME] [varchar] (7) not null, [YEAR_MONTH_NAME] [varchar] (8) not null, [YEAR_MONTH_NAME_LONG]
[varchar] (14) not null, [QUARTER_NAME] [varchar] (2) not null, [MONTH_NAME] [varchar] (3) not null,
[MONTH_NAME_LONG] [varchar] (9) not null,[WEEKDAY_NAME] [varchar] (3) not null,[WEEKDAY_NAME_LONG] [varchar]
(9) not null,[START_OF_YEAR_DATE] [datetime] not null,[END_OF_YEAR_DATE] [datetime] not
null,[START_OF_QUARTER_DATE] [datetime] not null,[END_OF_QUARTER_DATE] [datetime] not
null,[START_OF_MONTH_DATE] [datetime] not null,[END_OF_MONTH_DATE] [datetime] not
null,[DATE_FORMAT_YYYY_MM_DD] [varchar](10) not null,[DATE_FORMAT_YYYY_M_D] [varchar](10) not
null,[DATE_FORMAT_MM_DD_YYYY] [varchar](10) not null, [DATE_FORMAT_M_D_YYYY] [varchar](10) not
null,[DATE_FORMAT_MMM_D_YYYY] [varchar](12) not null,[DATE_FORMAT_MMMMMMMMM_D_YYYY] [varchar](18) not
null,[DATE_FORMAT_MM_DD_YY] [varchar](8) not null,[DATE_FORMAT_M_D_YY] [varchar](8) not null)
```

```
declare @cr varchar(2)
select @cr = char(13)+Char(10)
declare @ErrorMessage varchar(400)
declare @START_DATE datetime
declare @END_DATE datetime
declare @LOW_DATE datetime
declare @FIRST_DATE AS DATETIME = '01/01/1990'
declare @LAST_DATE AS DATETIME = '01/01/2013'
declare    @start_no   int
declare    @end_no    int
-- Verify @FIRST_DATE is not null
if @FIRST_DATE is null
    begin select @ErrorMessage =
        '@FIRST_DATE cannot be null'
    goto Error_Exit
    end
-- Verify @LAST_DATE is not null
if @LAST_DATE is null
    begin
    select @ErrorMessage =
        '@LAST_DATE cannot be null'
    goto Error_Exit
    end
-- Verify @FIRST_DATE is not before 1754-01-01
IF  @FIRST_DATE < '17540101'  begin
    select @ErrorMessage =
        '@FIRST_DATE cannot be before 1754-01-01'+
        ', @FIRST_DATE = '+
        isnull(convert(varchar(40),@FIRST_DATE,121),'NULL')
    goto Error_Exit
    end
-- Verify @LAST_DATE is not after 9997-12-31
IF  @LAST_DATE > '99971231'   begin
    select @ErrorMessage =
        '@LAST_DATE cannot be after 9997-12-31'+
        ', @LAST_DATE = '+
        isnull(convert(varchar(40),@LAST_DATE,121),'NULL')
    goto Error_Exit
    end
-- Verify @FIRST_DATE is not after @LAST_DATE
if @FIRST_DATE > @LAST_DATE
    begin
    select @ErrorMessage =
        '@FIRST_DATE cannot be after @LAST_DATE'+
        ', @FIRST_DATE = '+
        isnull(convert(varchar(40),@FIRST_DATE,121),'NULL')+
        ', @LAST_DATE = '+
        isnull(convert(varchar(40),@LAST_DATE,121),'NULL')
    goto Error_Exit
    end
-- Set @START_DATE = @FIRST_DATE at midnight
select @START_DATE    = dateadd(dd,datediff(dd,0,@FIRST_DATE),0)
-- Set @END_DATE = @LAST_DATE at midnight
select @END_DATE  = dateadd(dd,datediff(dd,0,@LAST_DATE),0)
-- Set @LOW_DATE = earliest possible SQL Server datetime
select @LOW_DATE  = convert(datetime,'17530101')
-- Find the number of day from 1753-01-01 to @START_DATE and @END_DATE
select    @start_no = datediff(dd,@LOW_DATE,@START_DATE) , @end_no = datediff(dd,@LOW_DATE,@END_DATE)
-- Declare number tables
declare @num1 table (NUMBER int not null primary key clustered)
declare @num2 table (NUMBER int not null primary key clustered)
declare @num3 table (NUMBER int not null primary key clustered)
-- Declare table of ISO Week ranges
declare @ISO_WEEK table([ISO_WEEK_YEAR] int not null primary key clustered, [ISO_WEEK_YEAR_START_DATE] datetime not
null, [ISO_WEEK_YEAR_END_DATE] Datetime not null)
```

```
-- Find rows needed in number tables
declare @rows_needed int
declare @rows_needed_root int
select  @rows_needed = @end_no - @start_no + 1
select  @rows_needed = case when @rows_needed < 10  then 10  else @rows_needed  end
select @rows_needed_root = convert(int,ceiling(sqrt(@rows_needed)))
-- Load number 0 to 16
insert into @num1 (NUMBER)
select NUMBER = 0 union all select  1 union all select  2 union all
select 3 union all select  4 union all select  5 union all
select 6 union all select  7 union all select  8 union all
select 9 union all select 10 union all select 11 union all
select 12 union all select 13 union all select 14 union all
select 15 order by  1
-- Load table with numbers zero thru square root of the number of rows needed +1
insert into @num2 (NUMBER)
select NUMBER = a.NUMBER+(16*b.NUMBER)+(256*c.NUMBER)
from @num1 a cross join @num1 b cross join @num1 c
where a.NUMBER+(16*b.NUMBER)+(256*c.NUMBER) < @rows_needed_root order by 1
-- Load table with the number of rows needed for the date range
insert into @num3 (NUMBER)
select NUMBER = a.NUMBER+(@rows_needed_root*b.NUMBER)
from @num2 a cross join @num2 b
where a.NUMBER+(@rows_needed_root*b.NUMBER) < @rows_needed order by 1
declare @iso_start_year int
declare @iso_end_year int
select @iso_start_year = datepart(year,dateadd(year,-1,@start_date))
select  @iso_end_year = datepart(year,dateadd(year,1,@end_date))
-- Load table with start and end dates for ISO week years
insert into @ISO_WEEK( [ISO_WEEK_YEAR], [ISO_WEEK_YEAR_START_DATE], [ISO_WEEK_YEAR_END_DATE])
select [ISO_WEEK_YEAR] = a.NUMBER, [0ISO_WEEK_YEAR_START_DATE] = dateadd(dd,(datediff(dd,@LOW_DATE,
dateadd(day,3,dateadd(year,a.[NUMBER]-1900,0)))/7)*7,@LOW_DATE), [ISO_WEEK_YEAR_END_DATE] = dateadd(dd,-
1,dateadd(dd,(datediff(dd,@LOW_DATE, dateadd(day,3,dateadd(year,a.[NUMBER]+1-1900,0)))/7)*7,@LOW_DATE))
from( select NUMBER = NUMBER+@iso_start_year from @num3 where NUMBER+@iso_start_year <= @iso_end_year) a order by
a.NUMBER
-- Load Date table
insert into [Dim].[Time]
select [DATE_ID] = convert (int, a.[DATE]), [DATE] = a.[DATE] , [NEXT_DAY_DATE] = dateadd(day,1,a.[DATE]), [YEAR] =
datepart(year,a.[DATE]), [YEAR_QUARTER] = (10*datepart(year,a.[DATE]))+datepart(quarter,a.[DATE]), [YEAR_MONTH] =
(100*datepart(year,a.[DATE]))+datepart(month,a.[DATE]), [YEAR_DAY_OF_YEAR] =
(1000*datepart(year,a.[DATE]))+datediff(dd,dateadd(yy,datediff(yy,0,a.[DATE]),0),a.[DATE])+1 ,[QUARTER] =
datepart(quarter,a.[DATE]) , [MONTH] = datepart(month,a.[DATE]) , [DAY_OF_YEAR] =
datediff(dd,dateadd(yy,datediff(yy,0,a.[DATE]),0),a.[DATE])+1 , [DAY_OF_MONTH] = datepart(day,a.[DATE]) , [DAY_OF_WEEK]
= (datediff(dd,'17530107',a.[DATE])%7)+1 , [YEAR_NAME] = datename(year,a.[DATE]) , [YEAR_QUARTER_NAME] =
datename(year,a.[DATE])+' Q'+datename(quarter,a.[DATE]) , [YEAR_MONTH_NAME] = datename(year,a.[DATE])+'
'+left(datename(month,a.[DATE]),3) , [YEAR_MONTH_NAME_LONG]  = datename(year,a.[DATE])+' '+datename(month,a.[DATE]) ,
[QUARTER_NAME] = 'Q'+datename(quarter,a.[DATE]) , [MONTH_NAME]=left(datename(month,a.[DATE]),3),
[MONTH_NAME_LONG]=datename(month,a.[DATE]),[WEEKDAY_NAME]=left(datename(weekday,a.[DATE]),3),[WEEKDAY_N
AME_LONG]=datename(weekday,a.[DATE]),[START_OF_YEAR_DATE]=dateadd(year,datediff(year,0,a.[DATE]),0),[END_OF_YEA
R_DATE]=dateadd(day,-1,dateadd(year,datediff(year,0,a.[DATE])+1,0)), [START_OF_QUARTER_DATE]=
dateadd(quarter,datediff(quarter,0,a.[DATE]),0),[END_OF_QUARTER_DATE]=dateadd(day,-1 ,dateadd(quarter,datediff(
quarter,0,a.[DATE])+1,0)),[START_OF_MONTH_DATE]=dateadd(month,datediff(month,0,a.[DATE]),0),[END_OF_MONTH_DATE]
=dateadd(day,- 1,dateadd(month,datediff(month,0,a.[DATE])+1,0)), [DATE_FORMAT_YYYY_MM_DD]=
convert(char(10),a.[DATE],111), [DATE_FORMAT_YYYY_M_D]= convert(varchar(10),convert(varchar(4),year(a.[DATE]))+'/'+
convert(varchar(2),day(a.[DATE]))+'/'+convert(varchar(2),month(a.[DATE]))),[DATE_FORMAT_MM_DD_YYYY]=convert(char(10),a.
[DATE],101),[DATE_FORMAT_M_D_YYYY]=convert(varchar(10),convert(varchar(2),month(a.[DATE]))+'/'+convert(varchar(2),day(a.
[DATE]))+'/'+convert(varchar(4),year(a.[DATE]))),[DATE_FORMAT_MMM_D_YYYY]=convert(varchar(12),left(datename(month,a.[D
ATE]),3)+''+convert(varchar(2),day(a.[DATE]))+','+convert(varchar(4),year(a.[DATE]))),[DATE_FORMAT_MMMMMMMMM_D_YYY
Y]=convert(varchar(18),datename(month,a.[DATE])+''+convert(varchar(2),day(a.[DATE]))+','+convert(varchar(4),year(a.[DATE]))),[D
ATE_FORMAT_MM_DD_YY]=convert(char(8),a.[DATE],1),[DATE_FORMAT_M_D_YY]=convert(varchar(8),convert(varchar(2),mon
th(a.[DATE]))+'/'+convert(varchar(2),day(a.[DATE]))+'/'+right(convert(varchar(4),year(a.[DATE])),2))
from(
-- Derived table is all dates needed for date range
select top 100 percent [DATE_ID]=aa.[NUMBER],[DATE]=dateadd(dd,aa.[NUMBER],@LOW_DATE)
from
```

```
(select NUMBER=NUMBER+@start_no from @num3 where NUMBER+@start_no<=@end_no )aa order by aa.[NUMBER] )a join
    -- Match each date to the proper ISO week year
    @ISO_WEEK b on a.[DATE] between b.[ISO_WEEK_YEAR_START_DATE] and b.[ISO_WEEK_YEAR_END_DATE] order by
a.[DATE_ID]
return
Error_Exit:
-- Return a pesudo error message by trying to
-- convert an error message string to an int.
-- This method is used because the error displays
-- the string it was trying to convert, and so the
-- calling application sees a formatted error message.
declare @error int
set @error = convert(int,@cr+@cr+
'*************************************************************'+@cr+
'* Error in Procedure Dim.Time:'+@cr+'* '+
isnull(@ErrorMessage,'Unknown Error')+@cr+
'*************************************************************'+@cr+@cr)
Return
```

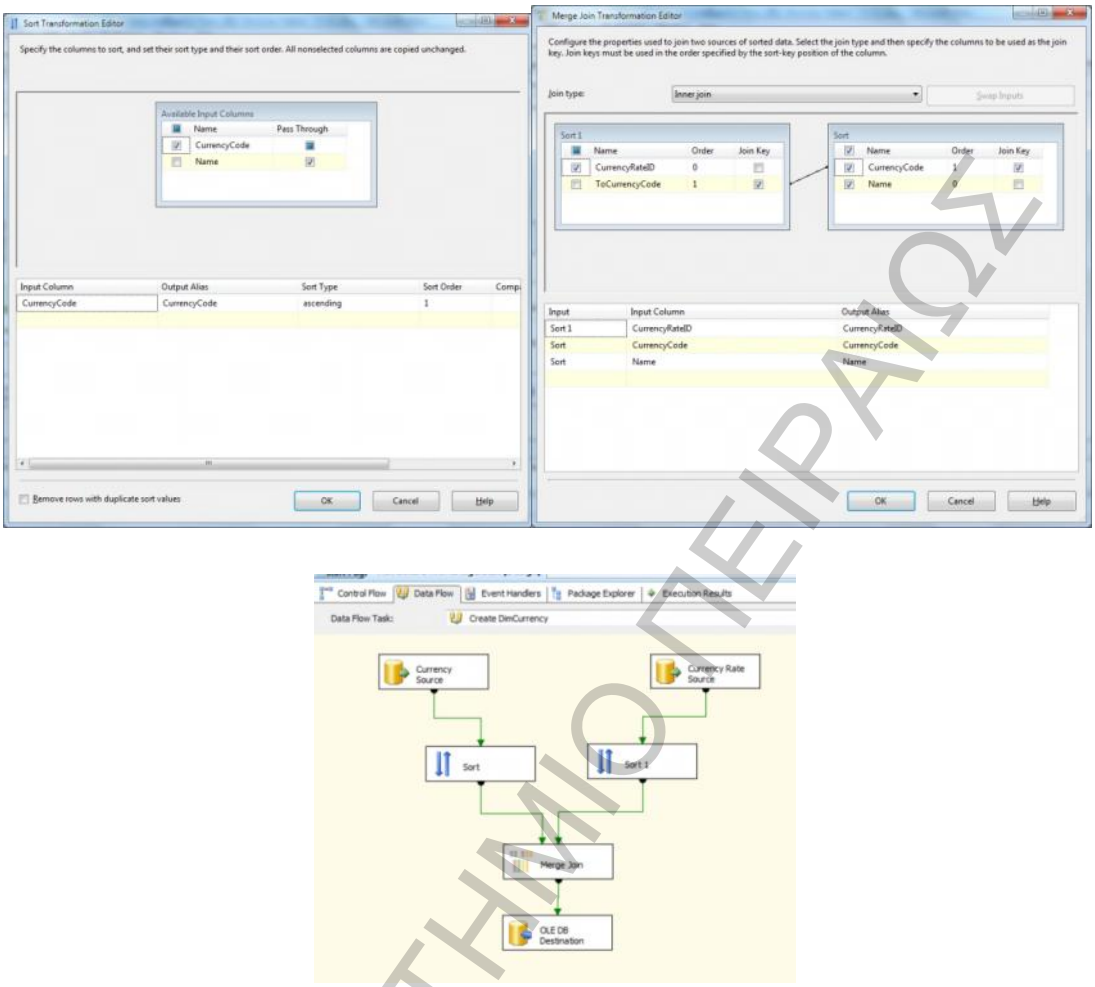
<u>One Step Dimensions</u>: Στο Data Flow Task αυτό περιέχονται οι ενέργειες που δημιουργούν dimensions των οποίων τα πεδία προέρχονται αποκλειστικά από έναν πίνακα της staging βάσης (για το καθένα αντίστοιχα). Έτσι λοιπόν έχουμε τη δημιουργία του πίνακα Dim.ShipmentMethod από τον πίνακα Purchasing.ShipMethod όπου το πεδίο ShipMethodID μεταφέρεται αυτούσιο ενώ το πεδίο Name μετονομάζεται σε ShipmentMethod. Επιπλέον έχουμε τη δημιουργία του πίνακα Dim.CreditCard από τον πίνακα Sales.CreditCard όπου μεταφέρονται αυτούσια τα πεδία CreditCardID και CardType. (Εικόνα 3.12)

Εικόνα 3.12: Data Flow Tasks Δημιουργίας One Step Dimensions.

Dim.ShipmentMethod	
[Data Warehouse Field]	**[Table].[Field]**
[ShipMethodID]	[Purchasing.ShipMethod].[ShipMethodID]
[ShipmentMethod]	[Purchasing.ShipMethod].[Name]
Dim.CreditCard	
[Data Warehouse Field]	**[Table].[Field]**
[CreditCardID]	[Sales.CreditCard].[CreditCartID]
[CardType]	[Sales.CreditCard].[CardType]

Create DimCurrency: Ο πίνακας αυτός χρειάζεται πεδία που περιέχονται σε δύο πίνακες της Staging βάσης. Η απεικόνιση βρίσκεται στον παρακάτω πίνακα. Για να μπορέσουμε να συγχωνεύσουμε τα πεδία αυτά σε ένα πίνακα χρησιμοποιούμε το Transformation Merge Join. Για να λειτουργήσει σωστά το Join θα πρέπει να υπάρχει ένα κοινό πεδίο που να συνδέει τους δυο πίνακες. Το πεδίο αυτό είναι το Currency Code που βρίσκεται και στους δυο πίνακες. Επιπλέον πρέπει τα πεδία, πριν γίνει το Merge Join, να έχουν ταξινομηθεί κατά το πεδίο αυτό και αυτό επιτυγχάνεται με το Sort Transformation. Το Join που γίνεται είναι Inner Join, όπου κατά την συγχώνευση των πινάκων Α και Β ψάχνει για κάθε εγγραφή του Α να βρει μια αντίστοιχη εγγραφή του πίνακα Β ως προς το πεδίο που έχει οριστεί να γίνει το Join και αν βρει δημιουργεί μια νέα εγγραφή στον τελικό πίνακα με τα επιθυμητά πεδία του Join. (Εικόνα 3.13)

Dim.Currency		
[Data Warehouse Field]	**[Sales.Currency].[Field]**	**[Sales.CurrencyRate].[Field]**
[Currency]	[Name]	
[CurrencyCode]	[CurrencyCode] →Join→	[ToCurrencyCode]
[CurrencyRateID]		[CurrencyRateID]



Εικόνα 3.13: Βήματα Δημιουργίας του Currency Dimension

Create DimSalesPerson: Ο πίνακας αυτός αποτελείται από πεδία που προέρχονται από τρεις πίνακες, όπως φαίνεται στον παρακάτω πίνακα. Αρχικά γίνεται Merge (Inner) Join του πίνακα Sales.SalesPerson με τον HumanResources.Employee στο πεδίο BussinessEntityID, αφού έχει προηγηθεί ταξινόμηση και των δυο πινάκων ως προς το πεδίο αυτό. Έπειτα, ο πίνακας που προκύπτει συγχωνεύεται με τον πίνακα Person.Person ως προς το ίδιο πεδίο (BusinessEntityID) αφού έχει προηγηθεί ταξινόμηση του Person.Person ως προς αυτό.

Dim.SalesPerson			
[Data Warehouse Field]	**[Sales.SalesPerson].[Field]**	**[HumanResources.Employee].[Field]**	**[Person.Person].[Field]**
[BusinessEntityID]	[BusinessEntityID] →Join→	[BusinessEntityID] →Join→	[BusinessEntityID]
[JobTitle]		[JobTitle]	
[Gender]		[Gender]	
[PersonType]			[PersonType]
[FirstName]			[FirstName]
[LastName]			[LastName]

Create DimCustomer: Τα πεδία του πίνακα προέρχονται από 3 διαφορετικούς πίνακες της Staging βάσης δεδομένων. Αρχικά οι πίνακες Sales.Customer και Person.Person ταξινομούνται ως προς το πεδίο PersonID και BusinessEntityID αντίστοιχα και έπειτα γίνεται Merge (Inner) Join σε αυτό το πεδίο. Ο πίνακας που προκύπτει ταξινομείται ως προς το πεδίο StoreID για να συγχωνευτεί με Merge (Inner) Join με τον πίνακα Sales.Store ως προς το πεδίο BusinessEntityID (βάσει του οποίου είναι ταξινομημένος με Sort Transformation.

Dim.Customer			
[Data Warehouse Field]	**[Sales.Customer].[Field]**	**[Person.Person].[Field]**	**[Sales.Store].[Field]**
[CustomerID]	[CustmerID]		
[PersonID]	[PersonID] →Join→	[BusinessEntityID]	
[PersonType]		[PersonType]	
[FirstName]		[FirstName]	
[LastName]		[LastName]	
[StoreID]	[StoreID] →Join→	→Join→	[BusinessEntityID]
[Store]			[Name]

Create DimLocation: Στον πίνακα φαίνονται οι 4 πίνακες της Staging βάσης δεδομένων από τους οποίους αντλούνται τα πεδία του Dim.Location. Το πρώτο βήμα είναι η ταξινόμηση των πινάκων Person.Address και Person.StateProvince και το Merge (Inner) Join των

πινάκων αυτών ως προς το πεδίο StateProvinceID. Έπειτα, ο συγχωνευμένος πίνακας καθώς και ο πίνακας Person.CountryRegion ταξινομούνται και συγχωνεύονται (Merge Inner Join) ως προς το πεδίο CountryRegionCode. Τέλος, ο πίνακας που προκύπτει από αυτή τη συγχώνευση αυτή και ο πίνακας Sales.SalesTerritory ταξινομούνται και συγχωνεύονται με Merge (Inner) Join ως προς το πεδίο TerritoryID και προκύπτει ο πίνακας Dim.Location.

Dim.Location				
[Data Warehouse Field]	**[Person.Address].[Field]**	**[Person.StateProvince].[Field]**	**[Person.CountryRegion].[Field]**	**[Sales.SalesTerritory].[Field]**
[AddressID]	[AddressID]			
[Address]	[AddressLine]			
[City]	[City]			
[PostalCode]	[PostalCode]			
[StateProvinceID]	[StateProvinceID] →Join→	[StateProvinceID]		
[StateProvince]		[Name]		
[CountryRegionCode]		[CountryRegionCode] →Join→	[CountryRegionCode]	
[Country]			[CountryRegionName]	
[TerritoryID]		[TerritoryID] →Join→	→Join→	[TerritoryID]
[Territory]				[Name]
[CountryGroup]				[Group]

Create DimSpecialOffer: Ο πίνακας αυτός αποτελείται από πεδία που προέρχονται από τρεις πίνακες της Staging βάσης δεδομένων, όπως φαίνεται στον παρακάτω πίνακα. Αρχικά ταξινομούμε και συγχωνεύουμε τους πίνακες Sales.SpecialOfferProduct και Sales.SpecialOffer ως προς το πεδίο SpecialOfferID. Έπειτα ο πίνακας που προκύπτει καθώς και ο πίνακας Production.Product ταξινομούνται και συγχωνεύονται (Merge (Inner) Join) ως προς το πεδίο ProductID και έτσι προκύπτει ο πίνακας Dim.SpecialOffer.

Dim.SpecialOffer			
[Data Warehouse Field]	**[Sales.SpecialOfferProduct].[Field]**	**[Sales.SpecialOffer].[Field]**	**[Production.Product].[Field]**
[SpecialOfferKey] – Primary Autonumber Key			
[SpecialOfferID]	[SpecialOfferID] →Join→	[SpecialOfferID]	
[SpecialOfferDescription]		[Description]	
[SpecialOfferDiscountPct]		[DiscountPct]	
[SpecialOfferType]		[Type]	
[SpecialOfferCategory]		[Category]	
[ProductID]	[ProductID] →Join→	→Join→	[ProductID]
[ProductName]			[Name]

Create DimProduct: Ο τελευταίος πίνακας των Dimensions προκύπτει από την επεξεργασία και συγχώνευση 4 πινάκων της Staging Βάσης. Αρχικά οι πίνακες Production.ProductSubcategory και Production.ProductCategory ταξινομούνται και συγχωνεύονται ως προς το πεδίο CategoryID. Ο πίνακας που προκύπτει ταξινομείται ως προς [ProductSubcategoryID] και συγχωνεύεται (Merge (Inner) Join) ως προς αυτό το πεδίο με τον πίνακα Production.Product ο οποίος έχει ταξινομηθεί ως προς το ίδιο πεδίο. Τέλος ο πίνακας που προκύπτει καθώς και ο πίνακας Production.ProductModel ταξινομούνται και συγχωνεύονται ως προς το πεδίο ProductModelID και προκύπτει ο πίνακας Dim.Product.

Dim.Product				
[Data Warehouse Field]	**[Production.ProductSubcategory].[Field]**	**[Production.ProductCategory].[Field]**	**[Production.Product].[Field]**	**[Production.ProductModel].[Field]**
[ProductCategoryID]	[ProductCategoryID] →Join→	[ProductCategoryID]		
[ProductSubcategory]	[Name]			
[ProductCategory]		[Name]		
[ProductSubcategoryID]	[ProductSubcategoryID] →Join →	→Join →	[ProductSubcategoryID] →Join →	
[ProductID]			[ProductID]	
[ProductName]			[Name]	
[ProductNumber]			[ProductNumber]	
[MakeFlag]			[MakeFlag]	
[FinishedGoodsFlag]			[FinishedGoodsFlag]	
[Color]			[Color]	

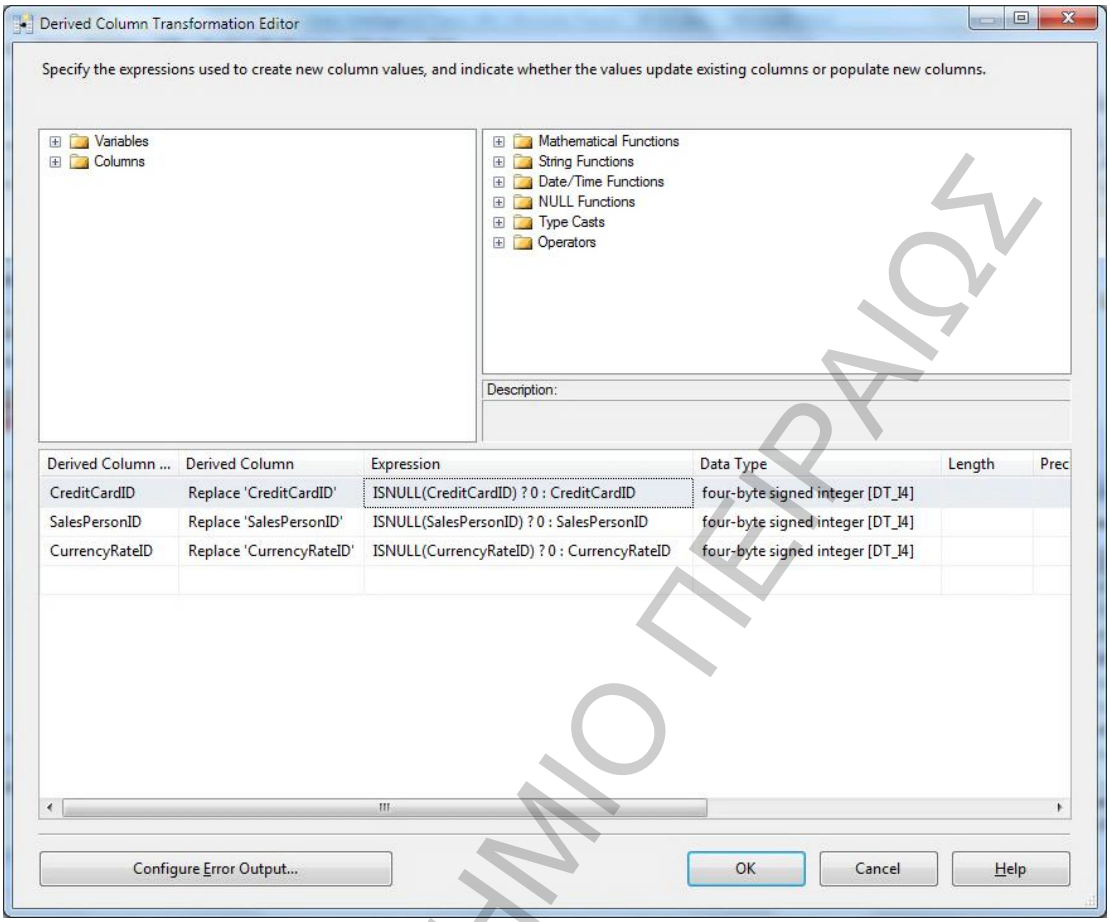
[ProductLine]			[ProductLine]	
[Class]			[Class]	
[Style]			[Style]	
[ProductModelID]			[ProductModelID] →Join→	[ProductModelID]
[ProductModel]				[Name]

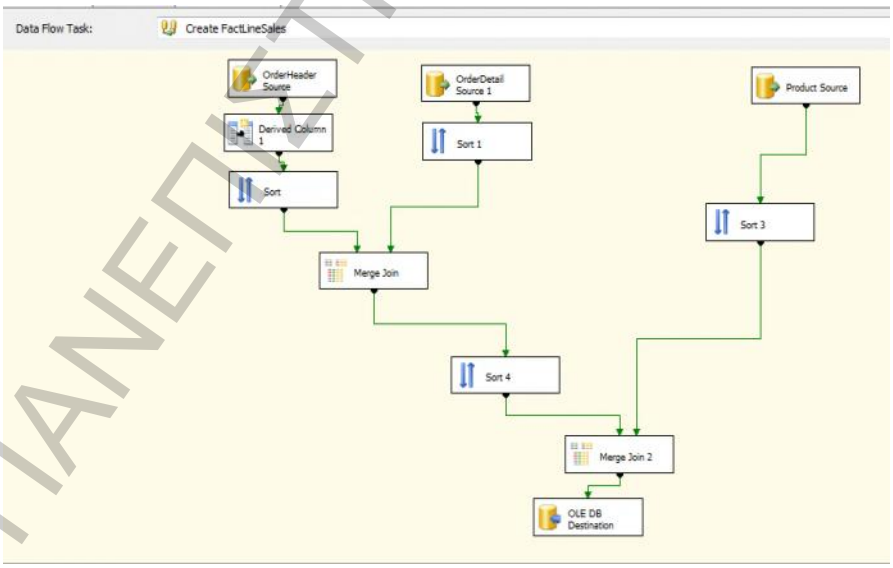
Βήμα 4: Το τελευταίο στάδιο είναι η δημιουργία των 3 Fact πινάκων του Data Warehouse. Στο Star Schema οι πίνακες Fact περιέχουν τα πεδία μέσω των οποίων συνδέονται με τους Dimension πίνακες (Secondary Keys) και τα measures που μας ενδιαφέρουν. Τα Data Flow Tasks που δημιουργούν αυτούς τους πίνακες βρίσκονται και αυτά σε ένα Sequence Container το οποίο εκτελείται τελευταίο, αμέσως μετά το Sequence Container δημιουργίας των Dimensions. Τα μέρη που απαρτίζουν αυτό το Sequence είναι τα εξής:

Create FactLineSales: Ο πίνακας Fact.LineSales αντλεί τα πεδία του από 3 πίνακες, όπως φαίνεται και στον πίνακα. Αρχικά εφαρμόζουμε ένα μετασχηματισμό που ονομάζεται Derived Column Transformation στα πεδία CreditCardID, SalesPersonID και CurrencyID του πίνακα Sales.SalesOrderHeader (Εικόνα 3.14). Σύμφωνα με αυτό το μετασχηματισμό, εάν η τιμή κάποιου από αυτά τα πεδία είναι κενό, τότε στο εγγράφεται η τιμή 0. Στο τελευταίο βήμα της δημιουργίας της Staging Database που ονομάζεται "Add Extra Rows For Cleansing" έχουμε προσθέσει στους αντίστοιχους πίνακες [Sales].[CreditCard], [Sales].[SalesPerson] και [Sales].[CurrencyRate] τις τιμές 0, ώστε να διαχειριστούμε και στα Dimensions την τιμή 0 και επομένως στον τελικό χρήστη να εμφανιστεί με user friendly τρόπο ότι δεν υπάρχει η συγκεκριμένη αντιστοιχία. Το επόμενο βήμα είναι η ταξινόμηση και

συγχώνευση των πινάκων Sales.SalesOrderHeader και Sales.SalesOrderDetail ως προς το πεδίο SalesOrderID. Ο πίνακας που προκύπτει ταξονομείται ως προς ProductID και συγχωνεύεται με τον πίνακα Production.Product ο οποίος έχει ταξινομηθεί ως προς το ίδιο πεδίο. Έτσι προκύπτει ο πίνακας Fact.LineSales (Εικόνα 3.14)

Fact.LineSales			
[Data Warehouse Field]	**[Sales.SalesOrderHeader].[Field]**	**[Sales.SalesOrderDetail].[Field]**	**[Production.Product].[Field]**
[OrderDate]	[OrderDate]		
[DueDate]	[DueDate]		
[Status]	[Status]		
[OnlineOrderFlag]	[OnlineOrderFlag]		
[CustomerID]	[CustomerID]		
[SalesPersonID]	[SalesPersonID]		
[BillToAddressID]	[BillToAddressID]		
[ShipToAddressID]	[ShipToAddressID]		
[ShipMethodID]	[ShipMethodID]		
[CreditCardID]	[CreditCardID]		
[CurrencyRateID]	[CurrencyRateID]		
[SalesOrderID]	[SalesOrderID] →Join→	[SalesOrderID]	
[SalesOrderDetailID]		[SalesOrderDetailID]	
[OrderQty]		[OrderQty]	
[PorductCost]			
[SpecialOfferID]		[SpecialOfferID]	
[UnitPrice]		[UnitPrice]	
[UnitPriceDiscount]		[UnitPriceDiscount]	
[LineTotal]		[UnitPriceDiscount]	
[ProductID]		[ProductID] →Join→	[ProductID]
[PorductCost]			StandardCost]





Εικόνα 3.14: Δημιουργία του Fact Πίνακα LineSales

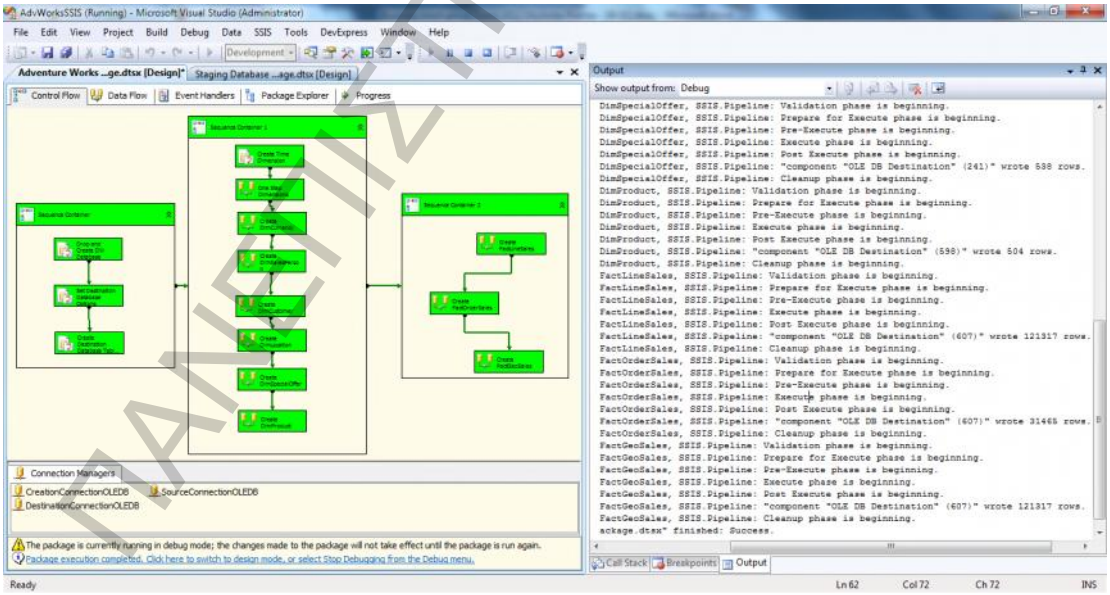
Create FactOrderSales: Ο πίνακας αυτός λαμβάνει τις τιμές του από ένα μόνο πίνακα της Staging βάσης δεδομένων, τον Sales.SalesOrderHeader. Οι μετασχηματισμοί που λαμβάνουν χώρα είναι ένα Derived Column Transformation μέσω του οποίου οι τιμές των πεδίων CreditCardID, SalesPersonID και CurrencyID μετατρέπονται σε 0, όπως και στη δημιουργία του Fact.LineSales καθώς και ένα Sort Transformation ως προς το SalesOrderID.

Fact.OrderSales	
[Data Warehouse Field]	**[Sales.SalesOrderHeader].[Field]**
[SalesOrderID]	[SalesOrderID]
[OrderDate]	[OrderDate]
[DueDate]	[DueDate]
[Status]	[Status]
[OnlineOrderFlag]	[OnlineOrderFlag]
[CustomerID]	[CustomerID]
[SalesPersonID]	[SalesPersonID]
[BillToAddressID]	[BillToAddressID]
[ShipToAddressID]	[ShipToAddressID]
[ShipMethodID]	[ShipMethodID]
[CreditCardID]	[CreditCardID]
[CurrencyRateID]	[CurrencyRateID]
[TaxAmt]	[TaxAmt]
[Freight]	[Freight]
[TotalDue]	[TotalDue]

Create FactGeoSales: Ο τελευταίος πίνακας που δημιουργείται είναι ο Fact.GeoSales ο οποίος αντλεί τις τιμές του από 3 πίνακες, όπως φαίνεται στον πίνακα. Αρχικά γίνεται ταξινόμηση και συγχώνευση των πινάκων Sales.SalesOrderHeader και Sales.SalesOrderDetail ως προς το πεδίο SalesOrderID και έπειτα ταξινόμηση του πίνακα που προκύπτει ως προς το πεδίο ShipToAddressID και συγχώνευση του με τον πίνακα Person.Address ο οποίος έχει ταξινομηθεί ως προς το πεδίο αυτό.

Fact.GeoSales			
[Data Warehouse Field]	**[Sales.SalesOrderHeader].[Field]**	**[Sales.SalesOrderDetail].[Field]**	**[Person.Address].[Field]**
[BillToAddressID]	[BillToAddressID]		
[SalesOrderID]	[SalesOrderID]→JOIN→	[SalesOrderID]	
[SalesOrderDetailID]		[SalesOrderDetailID]	
[OrderQty]		[OrderQty]	
[UnitPrice]		[UnitPrice]	
[UnitPriceDiscount]		[UnitPriceDiscount]	
[LineTotal]		[UnitPriceDiscount]	
[ShipToAddressID]	[ShipToAddressID] →JOIN→	→JOIN→	[AddressID]
[Longitude]			[Longitude]
[Latitude]			[Latitude]

Με την εκτέλεση του Package "Adventure Works DW Package" που μόλις περιγράφηκε, δημιουργείται η Data Warehouse βάση δεδομένων η οποία θα αποτελέσει πηγή δεδομένων για SSAS OLAP κύβους.(Εικόνα 3.15)



Εικόνα 3.15: Ολοκλήρωση δημιουργίας του Data Warehouse

3.4. Δημιουργία OLAP κύβων (Analysis Services Project)

Εφόσον έχει δημιουργηθεί η βάση δεδομένων Data Warehouse, πρέπει να δημιουργήσουμε τους κύβους οι οποίοι θα λαμβάνουν τα δεδομένα από αυτή, θα εκτελούν τα pre-aggregations, θα ορίζουν τις ιεραρχίες, τα measures, το analysis Service και γενικότερα θα δώσουν πρόσβαση στην επιθυμητή πληροφορία στους τελικούς χρήστες του BI system.

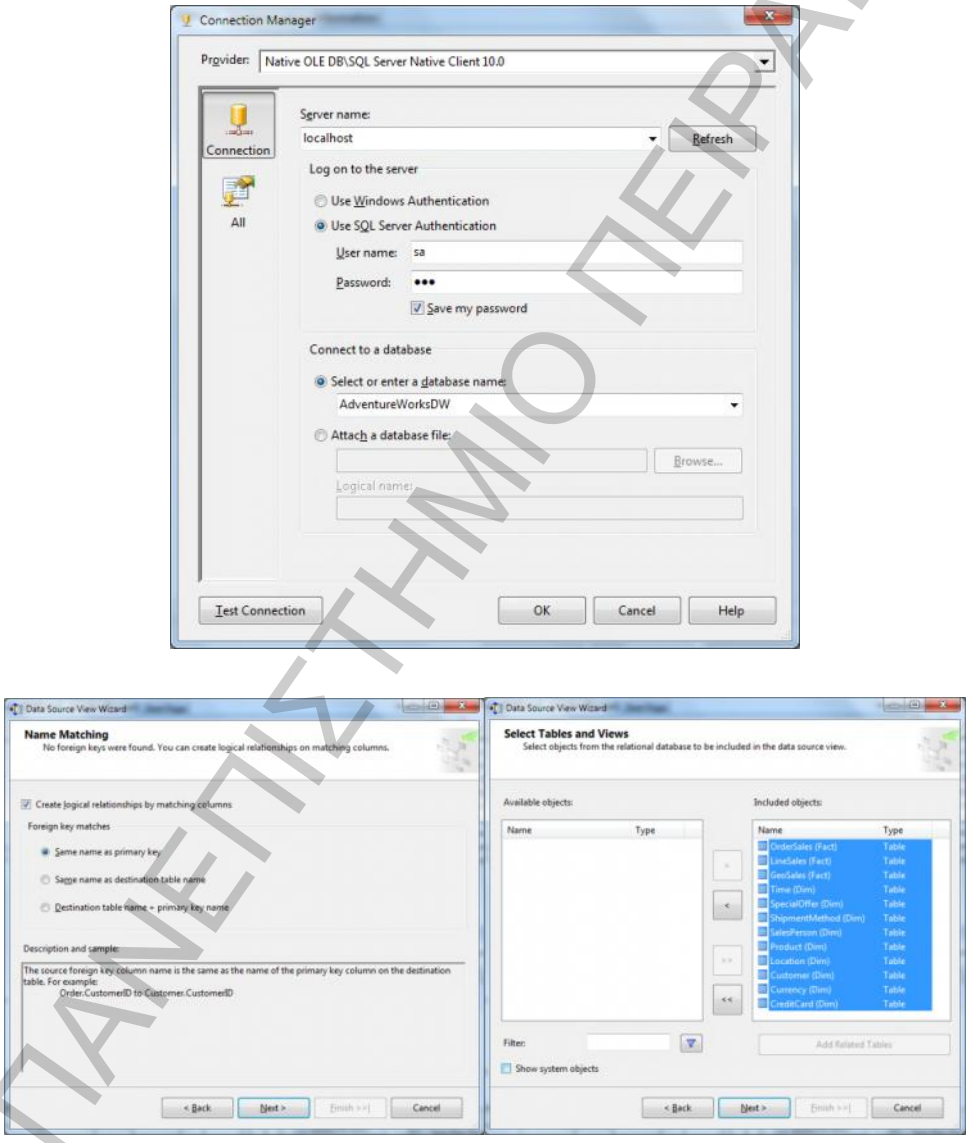
Τα βήματα που ακολουθήθηκαν για τη δημιουργία των κύβων και των Analysis Services είναι τα εξής:

Βήμα 1: Ανοίγουμε το Microsoft SQL Server Business Intelligence Studio Και δημιουργούμε ένα νέο Analysis Services Project με το όνομα AdvWorksSSAS.

Βήμα 2: Στην γραμμή εργαλείων που εμφανίζεται αριστερά δημιουργούμε ένα DataSource με το όνομα AdvWorksDWDataSource το οποίο, όπως φαίνεται στην εικόνα, συνδέεται με το localhost με UserID = sa και Password = 123 και βάση δεδομένων την Data Warehouse Βάση με όνομα AdventureWorksDW.

Βήμα 3: Δημιουργείται ένα Data Source View από μέσω του αντίστοιχου Wizard και ακλουθώντας τα βήματα (εικόνα) ορίζουμε το DataSource, επιτρέπουμε την αυτόματη δημιουργία Relationship πινάκων μέσω των κλειδιών ίδιου ονόματος και δίνουμε στο View το όνομα AdvWorksDWDataSourceView. Με την ολοκλήρωση του Wizard

δημιουργείται ένα View της DataWarehouse βάσης AdventureWorksDW και όπου είναι εφικτό έχουν οριστεί και τα Relationships μεταξύ των πινάκων.(Εικόνα 3.16) Ελέγχουμε τις σχέσεις των πινάκων και σε περίπτωση που έχει παραληφθεί κάποια τη δημιουργούμε εμείς έτσι ώστε οι σχέσεις να είναι αυτές οι οποίες προβλέπεται.

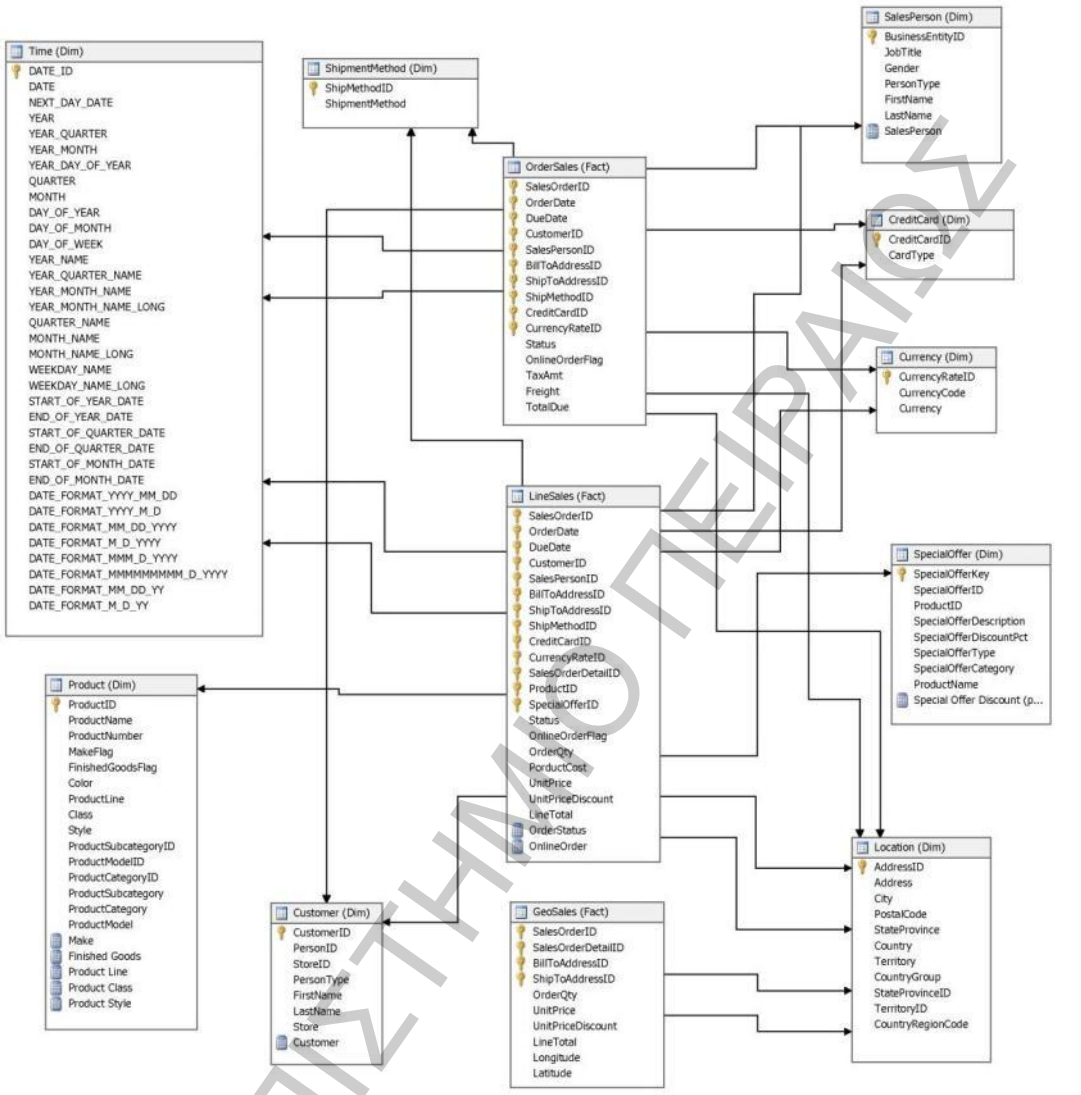


Εικόνα 3.16: Wizard Δημιουργίας SSAS DataSource View

Βήμα 4: Μας δίνεται η δυνατότητα να δημιουργήσουμε πεδία των οποίων οι τιμές θα προκύπτουν από την εφαρμογή υπολογισμών σε ήδη υπάρχοντα πεδία. Τα πεδία αυτά ονομάζονται Calculated Fields και ορίζονται στο Data Source View της βάσης δεδομένων. Για τις ανάγκες του συγκεκριμένου Project έχουν δημιουργηθεί τα εξής Calculated Fields.

Table	Calculated Field	Expression	Description
Dim.SalesPerson	SalesPerson	FirstName + LastName (εικόνα)	SalesPerson
Dim.SpecialOffer	Special Offer Discount (perc)	SpecialOfferDiscountPct*100	Special Offer Discount
Dim.Customer	Customer	FirstName + ' ' + LastName	Customer Full Name
Dim.Product	Make	CASE MakeFlag WHEN 0 THEN 'Purchased' WHEN 1 THEN 'Manufactured in-house' END	Make Flag 0 = Product is purchased, 1 = Product is manufactured in-house.
Dim.Product	Finished Goods	CASE FinishedGoodsFlag WHEN 0 THEN 'Not Salable' WHEN 1 THEN 'Salable' END	Finished Goods Flag 0 = Product is not a salable item. 1 = Product is salable
Dim.Product	Product Line	CASE ProductLine WHEN 'R' THEN 'Road' WHEN 'M' THEN 'Mountain' WHEN 'T' THEN 'Touring' WHEN 'S' THEN 'Standard' ELSE 'Other' END	Product Line R = Road, M = Mountain, T = Touring, S = Standard
Dim.Product	Product Class	CASE Class WHEN 'H' THEN 'High' WHEN 'M' THEN 'Medium' WHEN 'L' THEN 'Low' ELSE 'Other' END	Product Class H = High, M = Medium, L = Low
Dim.Product	Product Style	CASE Style WHEN 'W' THEN 'Womens' WHEN 'M' THEN 'Mens' WHEN 'U' THEN 'Universal' ELSE 'Other' END	Product Style W = Womens, M = Mens, U = Universal
Fact.LineSales	OrderStatus	CASE Status WHEN 1 THEN 'In process' WHEN 2 THEN 'Approved' WHEN 3 THEN 'Backordered' WHEN 4 THEN 'Rejected' WHEN 5 THEN 'Shipped' WHEN 6 THEN 'Cancelled' END	1 = In process; 2 = Approved; 3 = Backordered; 4 = Rejected; 5 = Shipped; 6 = Cancelled
Fact.LineSales	OnlineOrder	CASE OnlineOrderFlag WHEN 0 THEN 'Offline Order' WHEN 1 THEN 'Online Order' END	0 = Order placed by sales person. 1 = Order placed online by customer.

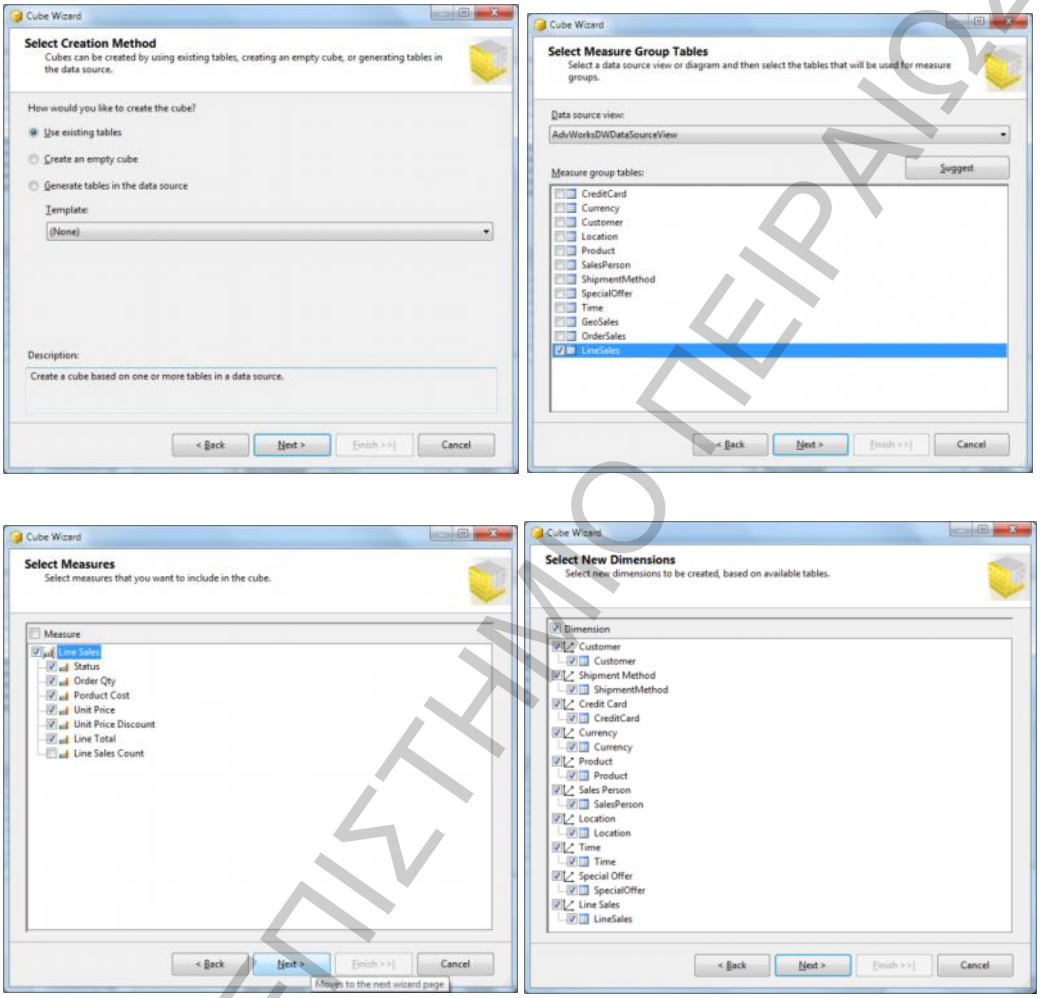
Με τη δημιουργία των Calculated Fields ολοκληρώνεται η δημιουργία του Data Source View, (Εικόνα 3.17)



Εικόνα 3.17: Οι Πινάκες και οι Σχέσεις του DW DataSource View

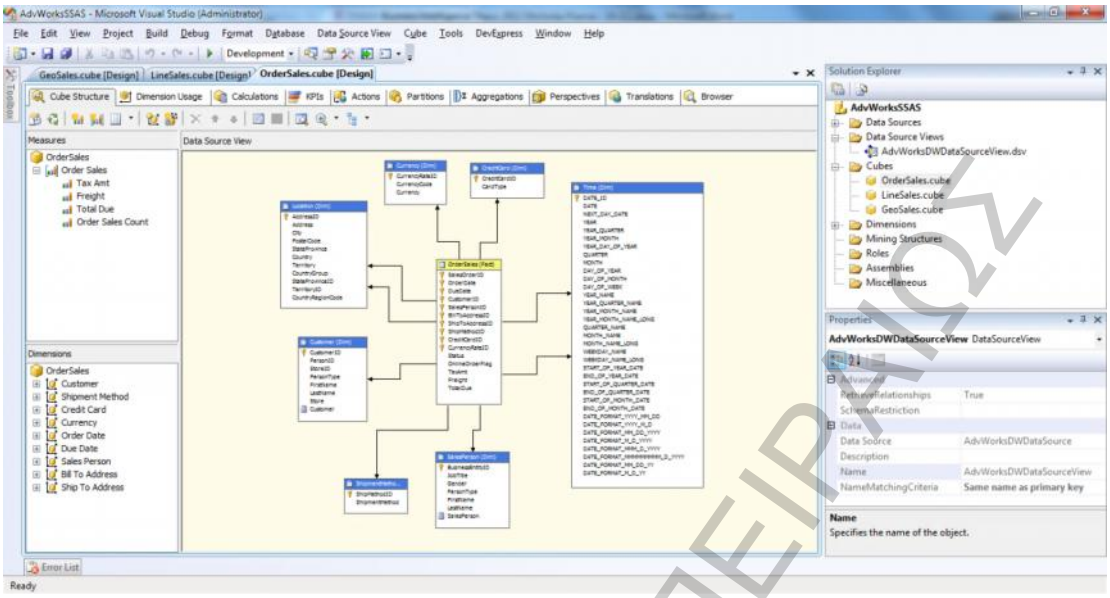
Βήμα 5: Το επόμενο βήμα είναι η δημιουργία των τριών κύβων που έχουν οριστεί στην ανάλυση. Οι κύβοι αυτοί είναι οι LineSales OrderSales και GeoSales. Για τη δημιουργία του κάθε κύβου τρέχουμε το αντίστοιχο Wizard μέσω του οποίου διαλέγουμε τον Fact πίνακα με τα measures, έπειτα τα measures που χρειαζόμαστε από τον πίνακα αυτό, καθώς και τα dimensions. Τέλος δίνουμε το όνομα του κύβου και με την ολοκλήρωση

του Wizard έχει δημιουργηθεί ο κύβος με τα measures αλλά και τα Dimensions που περιέχονται σε αυτόν. Με αυτή τη διαδικασία λοιπόν δημιουργήθηκαν και οι τρεις κύβοι και τα αντίστοιχα Dimensions.

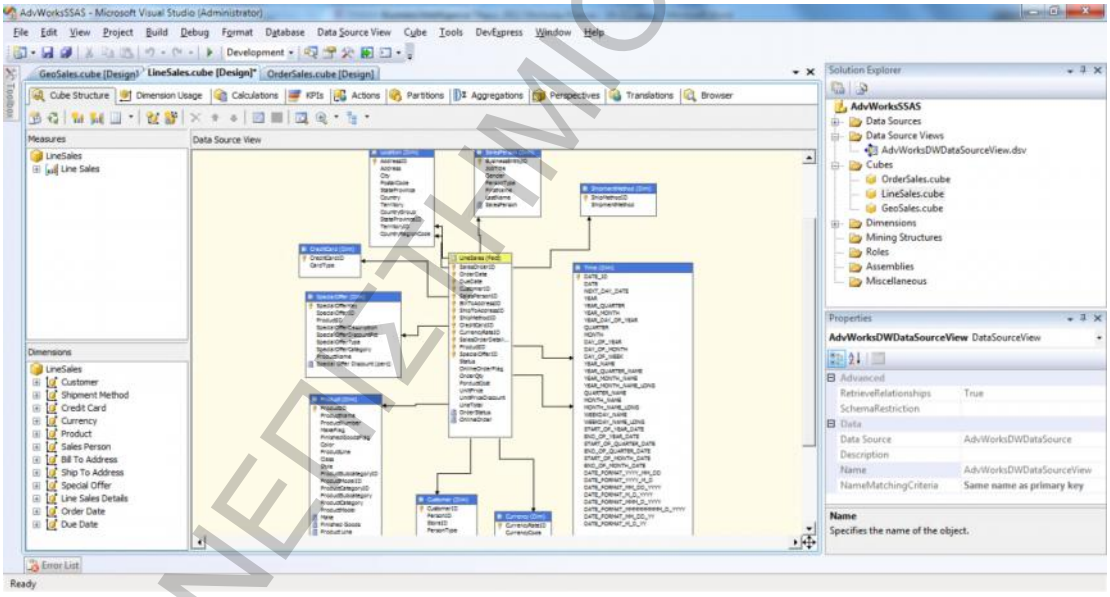


Εικόνα 3.18: Wizard κατασκευής του κύβου Line Sales

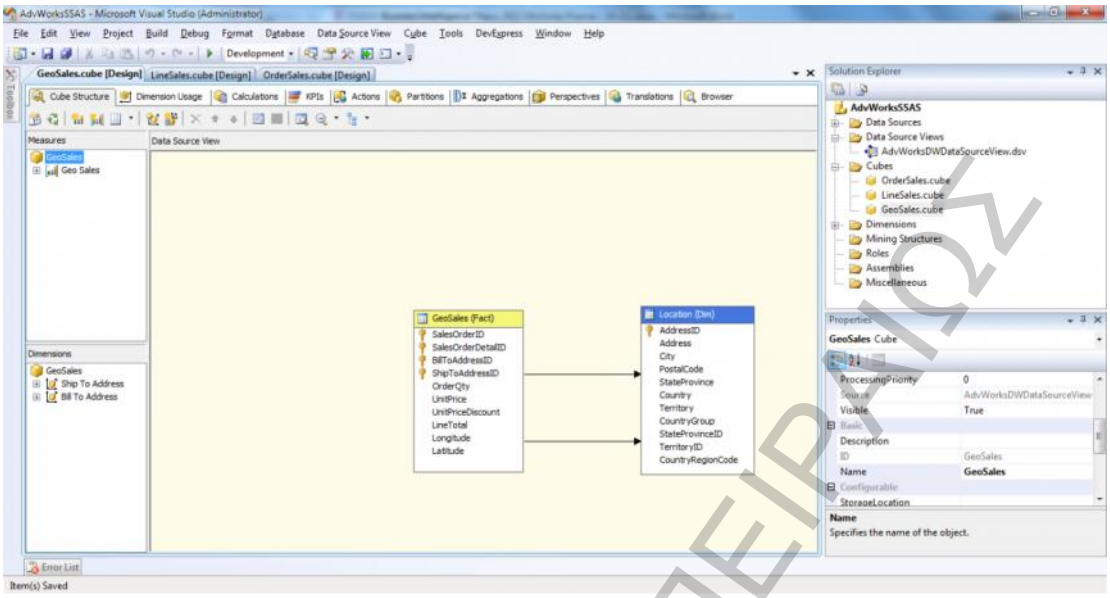
Οι 3 κύβοι φαίνονται στις παρακάτω εικόνες (3.19, 3.20, 3.21) :



Εικόνα 3.19: Κύβος Order Sales



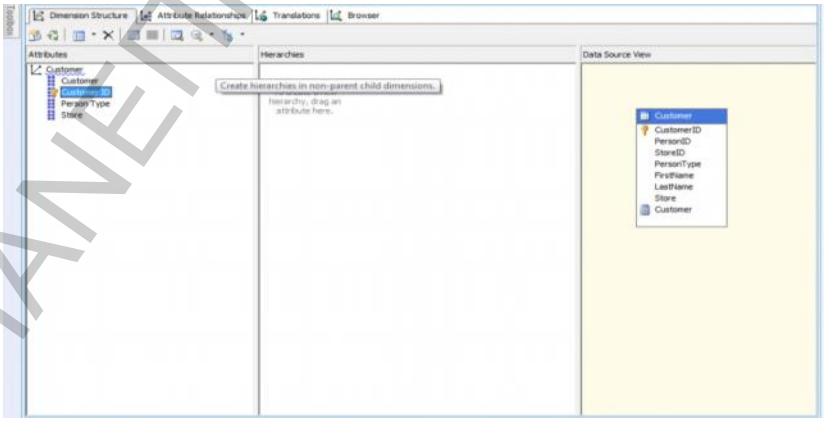
Εικόνα 3.20: Κύβος Line Sales



Εικόνα 3.21: Κύβος Geo Sales

Βήμα 6: Το τελευταίο βήμα είναι η επεξεργασία των Dimensions. Θα αναφερθούμε στο κάθε dimension ξεχωριστά. Έτσι έχουμε:

Customer: Όπως φαίνεται στην εικόνα, το Data Source View περιέχει όλα τα πεδία του Dimension (3η στήλη), ενώ στην 1η στήλη έχουμε τοποθετήσει τα Attributes του Dimensions που θέλουμε να εμφανίζονται. Δεν έχει οριστεί κάποια ιεραρχία για τη συγκεκριμένη διάσταση (Εικόνα 3.22).



Εικόνα 3.22: Dimension Customer

Shipment Method: Τα attributes του συγκεκριμένου Dimension είναι τα πεδία του αντίστοιχου πίνακα του View, δηλαδή το ShipmethodID που είναι και το Key Attribute και το Shipment Method. Δεν υπάρχει κάποια ορισμένη ιεραρχία.

Credit Card: Ομοίως με το Dim.ShipmentMethod τα attributes της διάστασης αυτής είναι το Credit Card ID (κλειδί) και το Credit Card.

Currency: Τα attributes είναι το Currency Rate ID (κλειδί) το Currency και το Currency Code, ενώ δεν υπάρχει κάποια ιεραρχία.

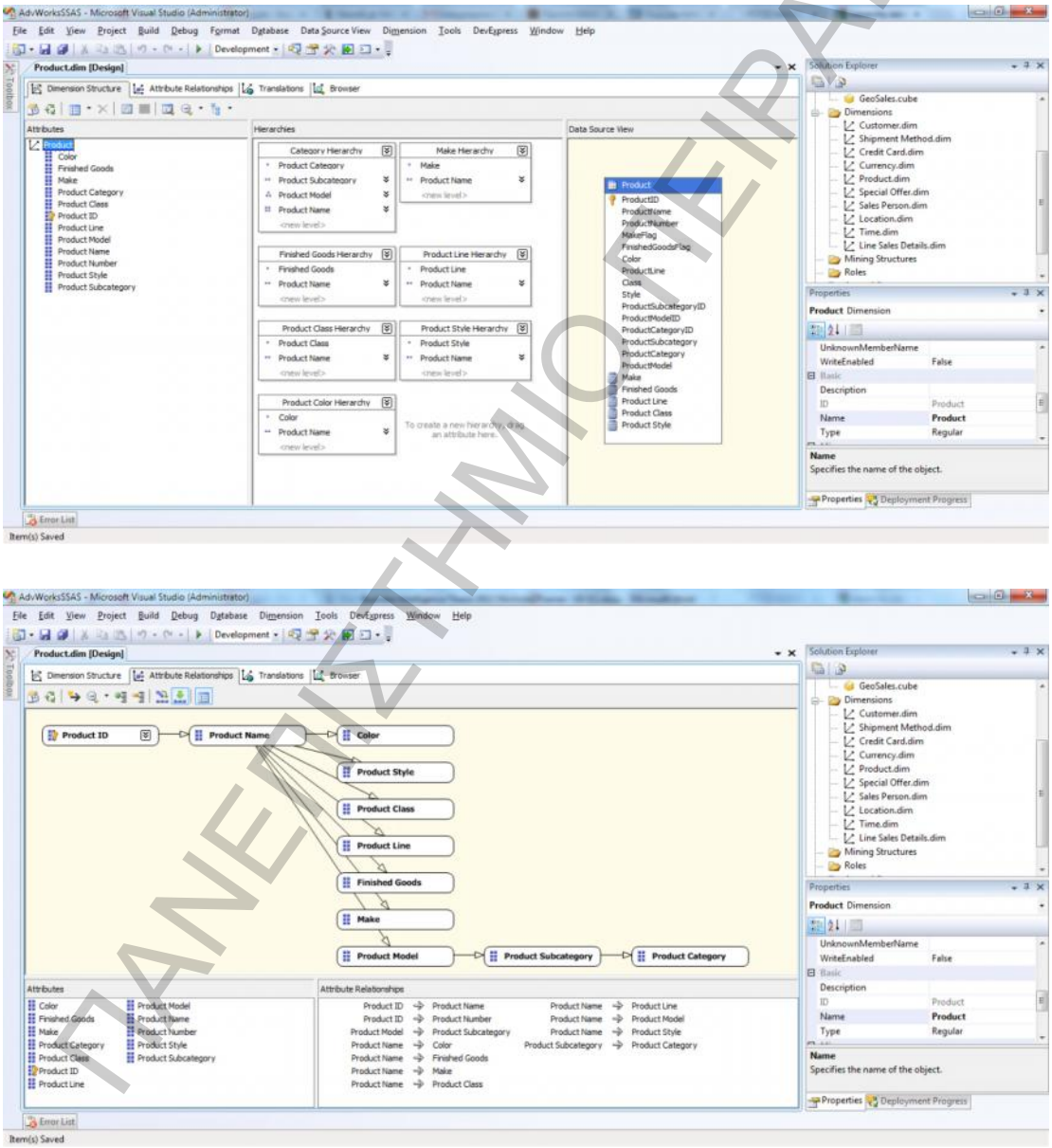
Special Offer: Τα attributes της διάστασης είναι τα Special Offer Key (κλειδί), το Product Name, το Special Offer Category, το Special Offer Description, το Special Offer Discount Perc και το Special Offer Type, ενώ δεν έχουν οριστεί ιεραρχίες.

Line Sales Details: Τα attributes που έχουν οριστεί είναι το Sales Order ID (κλειδί), το Online Order, το Order Status και το Sales Order Detail ID.

Product: Τα Attributes του συγκεκριμένου Dimension φαίνονται στην Εικόνα 3.23. Στη διάσταση αυτή έχουν οριστεί και ιεραρχίες όπως φαίνεται στην ίδια εικόνα. Στην ίδια εικόνα φαίνονται οι συσχετισμοί των Attributes.

Ένα χαρακτηριστικό των ιεραρχιών είναι ότι τα attributes τα οποία τις απαρτίζουν θα πρέπει να έχουν ένα μοναδικό κλειδί. Η κάθε εγγραφή της ιεραρχίας αποτελείται από τις τιμές των attributes που την απαρτίζουν, στη σειρά που εκείνα έχουν οριστεί. Ένας περιορισμός που προκύπτει είναι ότι δεν πρέπει να υπάρχουν διπλότυπα στις εγγραφές

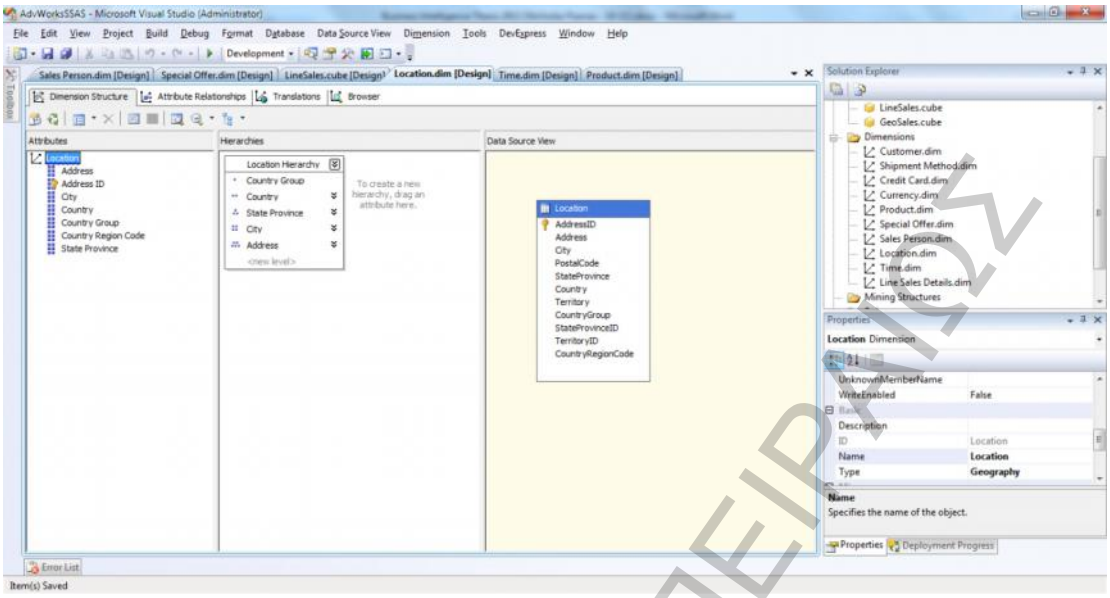
της ιεραρχίας. Για παράδειγμα η ιεραρχία "Η" αποτελείται από τα Attributes Α, Β, Γ και μια εγγραφή της είναι η Α1Β1Γ1. Δεν πρέπει να υπάρχει άλλη εγγραφή της ιεραρχίας "Η" με την τιμή Α1Β1Γ1. Στην περίπτωση του Product τα κλειδιά των Attributes που απαρτίζουν τις ορισμένες ιεραρχίες, εξασφαλίζουν ότι δεν υπάρχουν διπλές εγγραφές.



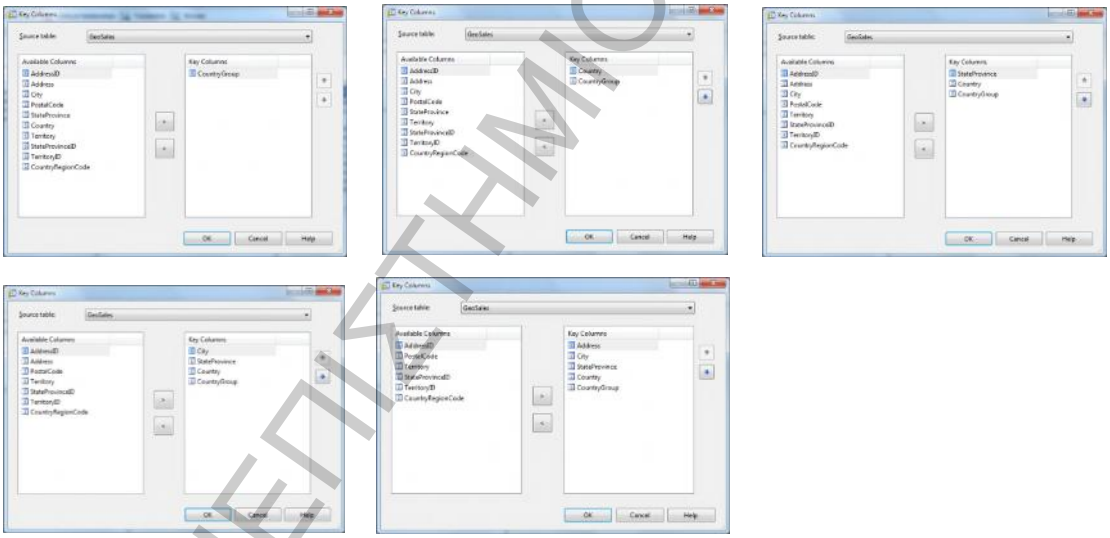
Εικόνα 3.23: Dimension Product και ιεραρχίες

Sales Person: Τα attributes της διάστασης είναι τα Business Entity ID (κλειδί), Gender, Job Title και Sales Person. Έχει οριστεί μια ιεραρχία με όνομα Sales Person Hierarchy και μέλη τα attributes Gender (1ο επίπεδο) και Sales Person (2ο επίπεδο). Οι τιμές των attributes αυτών είναι και κλειδιά γιατί οι συνδυασμοί τους δημιουργούν μοναδικές εγγραφές.

Location: Τα attributes που αποτελούν τη συγκεκριμένη διάσταση καθώς και η ιεραρχία που έχει οριστεί φαίνονται στην εικόνα 3.24. Αξίζει να σημειωθεί ότι οι τιμές των attributes της ιεραρχίας Location Hierarchy δεν μπορούν να αποτελέσουν κλειδιά της ιεραρχίας γιατί δημιουργούνται διπλότυπα (πχ εμφανίζονται παραπάνω από μια εγγραφές με την τιμή North America→ United States→ Oregon→ Albany). Η λύση που ακολουθήθηκε είναι να οριστεί μια συλλογή τιμών ως κλειδί για κάθε Attribute. Συγκεκριμένα, το attribute Country Group έχει ως column key τις τιμές του. Το Attribute Country, έχει ως column key την τιμή του αντίστοιχου Country Group και την τιμή τη δική του (Country). Το attribute State Province έχει ως Column Key την τιμή του Country Group που ανήκει, την τιμή του Country που ανήκει και την δική του τιμή κ.ο.κ. Έτσι λοιπόν, ουσιαστικά το κάθε ξεκινώντας από το 1ο επίπεδο της ιεραρχίας, το κλειδί του κάθε attribute που την απαρτίζει αποτελείται από την δική του τιμή και την τιμή του κλειδιού του attribute του προηγούμενου επιπέδου από αυτό. Με αυτό τον τρόπο επιτυγχάνεται η μοναδικότητα των εγγραφών της ιεραρχίας (εικόνα 3.25).



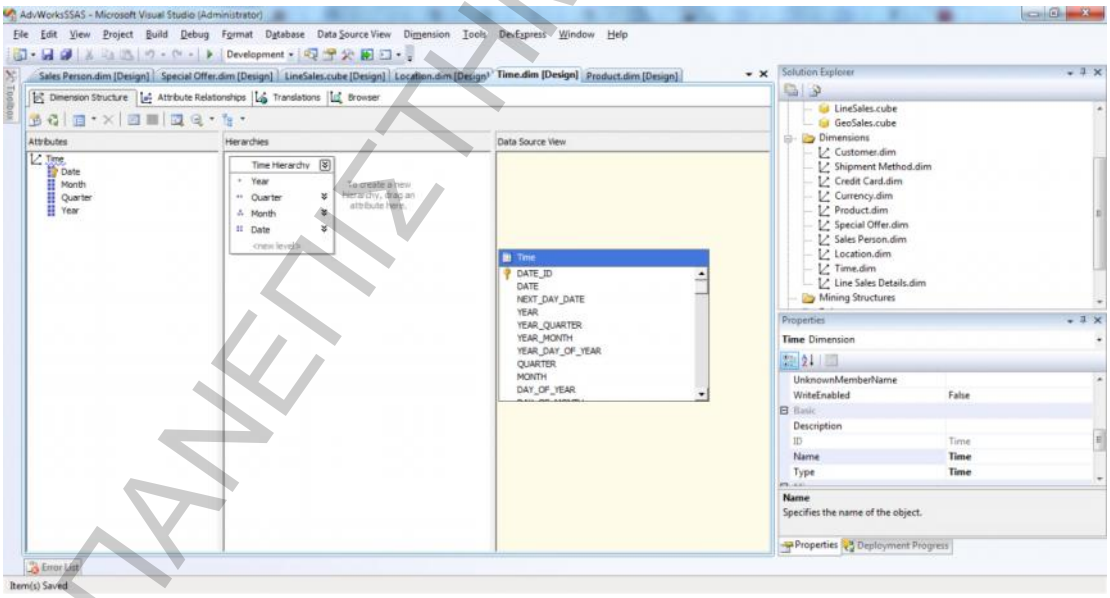
Εικόνα 3.24: Location Dimension και ιεραρχίες



Εικόνα 3.25: Ορισμός κλειδιών των μελών της ιεραρχίας Location

Time: Η εικόνα δείχνει τα attributes της χρονικής διάστασης καθώς και τα μέλη της ιεραρχίας Time Hierarchy (Εικόνα 3.26). Οι τιμές του αντίστοιχου πίνακα της συγκεκριμένης διάστασης προέρχονται από την εκτέλεση ενός SQL Script όπως περιγράφηκε στην προηγούμενη ενότητα. Κατά τη

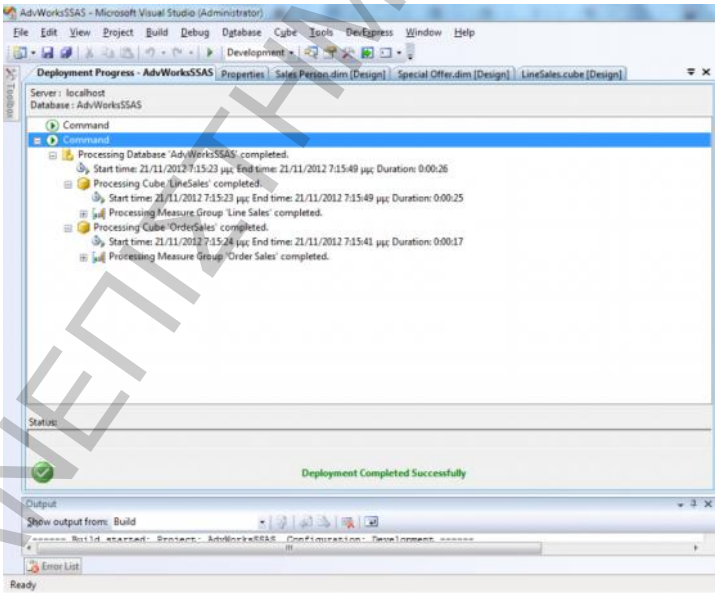
δημιουργία αυτών των τιμών προβλέφθηκε η δημιουργία πεδίων για κάθε attribute τα οποία θα δίνουν μοναδικές εγγραφές στην ιεραρχία αυτή. Έτσι το key column λαμβάνει τις τιμές αυτών των ειδικά διαμορφωμένων πεδίων και συγκεκριμένα Time.YEAR_MONTH για το Year, Time.YEAR_QUARTER για το Quarter, Time.YEAR_MONTH για το Month και Time.DATE_ID για το Date. Το Name όμως του κάθε attribute της ιεραρχίας, που είναι ουσιαστικά το display Text, είναι διαφορετικό ώστε να εμφανίζονται τιμές πιο φιλικές προς τον χρήστη (για παράδειγμα το Key value του Year Quarter για μια εγγραφή είναι 20092 ενώ το Name του είναι 2009 Q2). Οι αντιστοιχήσεις attributes της ιεραρχίας και Name είναι Time.YEAR_MONTH για το Year, Time.YEAR_QUARTER_NAME για το Quarter, Time.YEAR_MONTH_NAME_LONG για το Month και Time.DATE_FORMAT_MMM_D_YYYY για το Date.



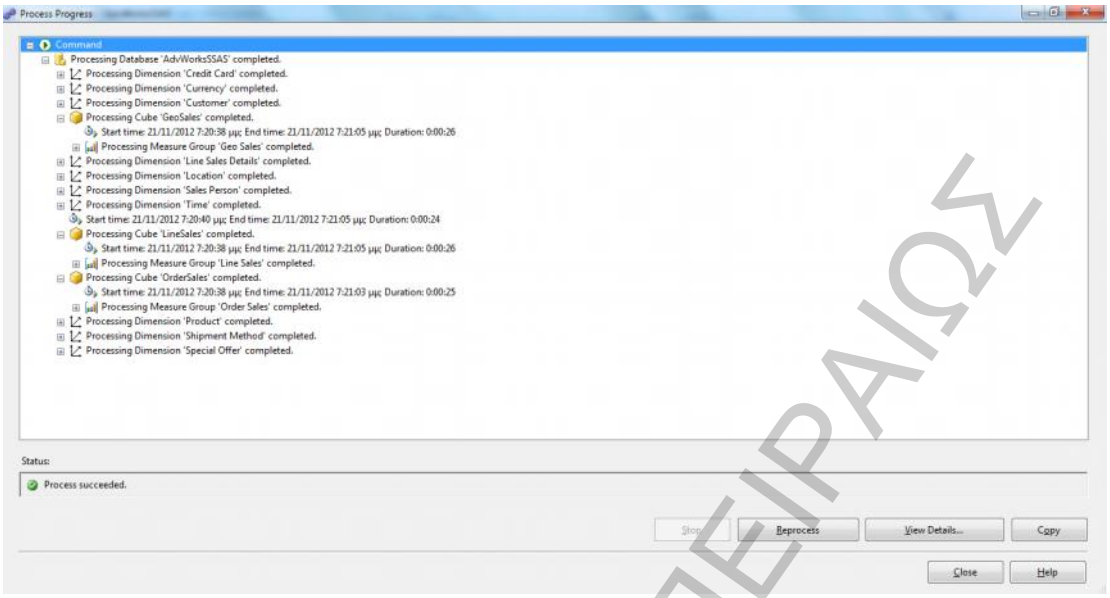
Εικόνα 3.26: Time Dimension

Επίσης πρέπει να σημειωθεί ότι ενώ στις υπόλοιπες ιεραρχίες των άλλων Dimensions η ταξινόμηση των τιμών των attributes είναι αλφαβητική, στην χρονική διάσταση επιθυμούμε να ταξινομούνται με βάση τον αριθμό- κωδικό που αντιστοιχεί στο κάθε attribute γι' αυτό και το order γίνεται βάση του κλειδιού του. (πχ αν γίνονταν αλφαβητικά θα εμφανίζονταν οι μήνες του Q3 με την σειρά August, July, June ενώ με τον τρόπο αυτό εμφανίζονται με τον σωστό τρόπο).

Έπειτα από το επιτυχές Deploy του Project (Εικόνα 3.27) και το Process των κύβων (Εικόνα 3.28) δημιουργείται το Service και οι κύβοι, γίνονται τα Pre-aggregations και πλέον τα δεδομένα είναι έτοιμα για προσπέλαση από τα αντίστοιχα Software (στην περίπτωσή μας το Business Intelligence Web Site).



Εικόνα 3.27: SSAS Project Deployment



Εικόνα 3.28: SSAS Cubes Process

3.5. Δημιουργία ASP.net BI Web Site (ASP.net Project)

Το τελευταίο βήμα στη δημιουργία του Συστήματος Business Intelligence της εταιρίας Adventure Works είναι η υλοποίηση του Web Portal μέσω του οποίου θα έχουμε πρόσβαση στα δεδομένα του Analysis Server και κατά προέκταση στα δεδομένα του Data Warehouse. Η τεχνολογία που χρησιμοποιήθηκε για τη δημιουργία του Web Site είναι το ASP.net της Microsoft με C# γλώσσα προγραμματισμού και η πλατφόρμα υλοποίησης είναι το Microsoft Visual Studio 2010. Η διαδικασία υλοποίησης του Web Site περιγράφεται στα παρακάτω βήματα.

Βήμα 1: Αρχικά πραγματοποιήθηκε η σχεδίαση της ιστοσελίδας, τα χρώματα, οι φωτογραφίες και όλο το αισθητικό κομμάτι. Η αρχική σελίδα στηρίζεται σε παραλλαγές του CSS κώδικα που προήλθε από την πηγή (28) (Styles/WelcomePage.css). Για το σχεδιασμό των υπόλοιπών σελίδων του Project χρησιμοποιήθηκε και τροποποιήθηκε ο κώδικας CSS της πηγής (27) (Styles/Site.css). Αντίστοιχα υπάρχουν 2 master αρχεία, το WelcomePage.Master και το Site.Master τα οποία περιέχουν asp ContentPlaceHolder Controls ώστε να τοποθετηθούν τα στοιχεία asp της κάθε σελίδας.

Βήμα 2: Περιγράφονται τα μέρη από τα οποία αποτελείται το Web Site καθώς και η λογική που έχει γραφεί ώστε να μπορεί το site να είναι λειτουργικό. Αναφέρεται ότι σε όσα σημεία του Project έχει γραφτεί κάποια λογική με χρήση C# κώδικα, έχουν τοποθετηθεί επεξηγηματικά σχόλια. Οι σελίδες από τις οποίες αποτελείται το Web Site είναι οι εξής:

- Default.aspx
- PivotTable.aspx
- PivotChart.aspx
- MapSettings.aspx
- Map.aspx
- Settings.aspx

Επιπλέον χρησιμοποιείται μια κλάση που ονομάζεται WebAppHelper, ένα xml αρχείο, το Settings.xml και ο φάκελος Images που περιέχει τα εικονίδια που χρησιμοποιούνται από την εφαρμογή.

Setting.xml: Επειδή δε χρησιμοποιείται κάποια βάση δεδομένων για την αποθήκευση απαραίτητων μεταβλητών του συστήματος, τις αποθηκεύουμε στο xml αρχείο. Οι μεταβλητές που αποθηκεύονται είναι το Connection String του Analysis Server (ConnectionString), τα Skin του pivot Table και του Pivot Chart (PivotTableSkin και PivotChartSkin) καθώς και το εικονίδιο του Marker που θα εμφανίζεται στον χάρτη (MapPointType).

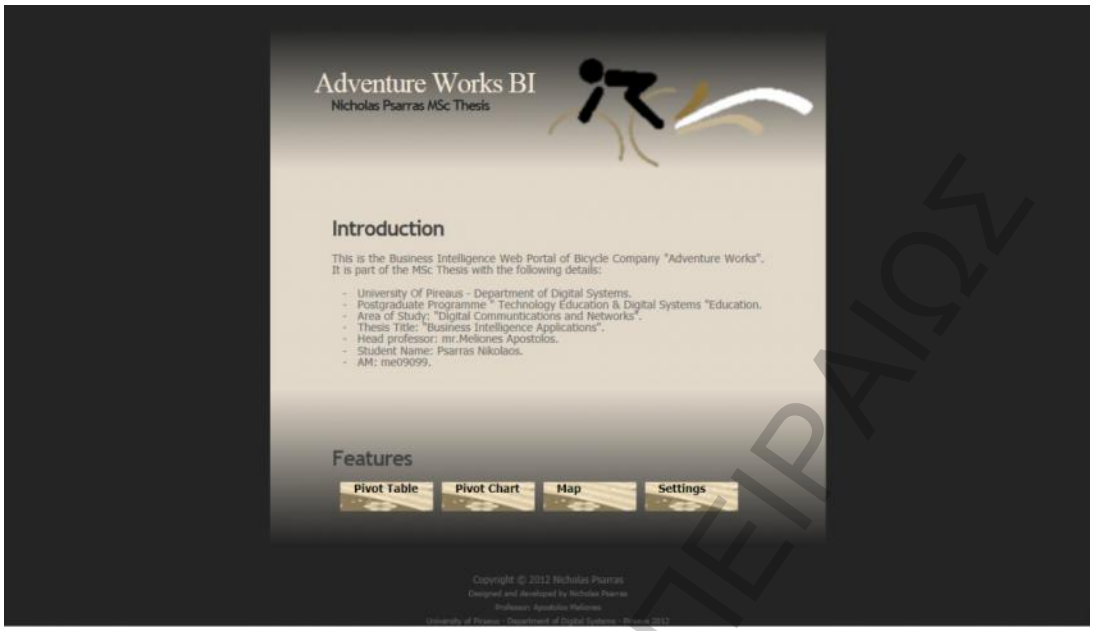
WebAppHelper: Είναι μια κλάση η οποία περιέχει βοηθητικές μεθόδους που μπορούν να είναι προσπελάσιμες από όλες τις σελίδες του Project. Κάποιες από τις μεθόδους χρησιμοποιούνται σε παραπάνω από ένα σημεία του κώδικα του Project και επομένως έτσι επιτυγχάνουμε την ύπαρξη διπλότυπων σημείων του ίδιου κώδικα. Οι μέθοδοι που υπάρχουν είναι οι εξής:

- ShowErrorMessage: Ορίζει ένα JavaScript κώδικα μέσω C# έτσι ώστε να εμφανίζεται στον Browser του Client ένα παράθυρο με το μήνυμα λάθους (που λαμβάνει ως παράμετρο).
- CheckXMLSettings: Ελέγχει αν υπάρχει το Settings.xml αρχείο καθώς και όλα τα απαραίτητα elements του αρχείου αυτού. Αν δεν υπάρχουν τότε τα δημιουργεί και τα αρχικοποιεί με συγκεκριμένες default τιμές.
- CreateChildElement: Δέχεται ως όρισμα το όνομα και την τιμή του Element και το δημιουργεί στο Settings.xml
- GetSettingsElement: Επιστρέφει την τιμή του element του Settings.xml που δέχεται ως όρισμα.
- InitComboWithCubes: Δέχεται σαν όρισμα ένα ComboBox Control και του προσθέτει ως τιμές τους κύβους που περιέχει ο Analysis Server και η DataBase στην οποία έχουμε συνδεθεί.
- ListCubes: Δέχεται σαν όρισμα το όνομα του Analysis Server και το όνομα της DataWarehouse Database και επιστρέφει μια λίστα με τους κύβους που εμπεριέχονται σε αυτή.
- InitComboWithDims: Δέχεται σαν όρισμα ένα ComboBox Control και το όνομα ενός κύβου και του προσθέτει ως τιμές τα Dimensions του κύβου αυτού.
- ListDims: Δέχεται σαν όρισμα το όνομα του Analysis Server, το όνομα της DataWarehouse Database και το όνομα του κύβου και επιστρέφει μια λίστα με τις διαστάσεις αυτού (dimensions).
- InitComboWithSkins: Αρχικοποιεί τις τιμές του ComboBox που δέχεται με τα διαθέσιμα Skins. Τα skins διαφέρουν ανάλογα με το

Control που θα εφαρμοστούν (Pivot Chart ή Pivot Table) και γι' αυτό το λόγο δέχεται ως όρισμα τον τύπο του Control

- InitComboWithMapStyles: Λαμβάνει τα ονόματα των εικονιδίων που βρίσκονται στο path "images/MapPoints/" και τα προσθέτει στο ComboBox που δέχεται ως όρισμα.
- SaveTxtSettings: Αποθηκεύει στο αρχείο Settings.xml τις τιμές που βρίσκονται σε TextBox Control (πχ ASConnectionString).
- SaveLKPSettings: Αποθηκεύει στο αρχείο Settings.xml τις τιμές που βρίσκονται σε Combobox Control (πχ PivotTableSkin).

Default.aspx: Είναι η αρχική σελίδα της εφαρμογής, χρησιμοποιεί το WelcomePage.Master και το WelcomePage.css. Εμφανίζονται κάποιες πληροφορίες για τη διπλωματική εργασία και τα links μέσω των οποίων οι χρήστες μπορούν να οδηγηθούν στις επιθυμητές σελίδες της εφαρμογής. Στο Page Load event υπάρχει μια κλήση στην μέθοδο WebAppHelper.CheckXMLSettings() για να γίνει ο έλεγχος του αν έχουν οριστεί/αποθηκευτεί οι απαραίτητες μεταβλητές για τη σωστή λειτουργία της εφαρμογής (εικόνα 3.29)



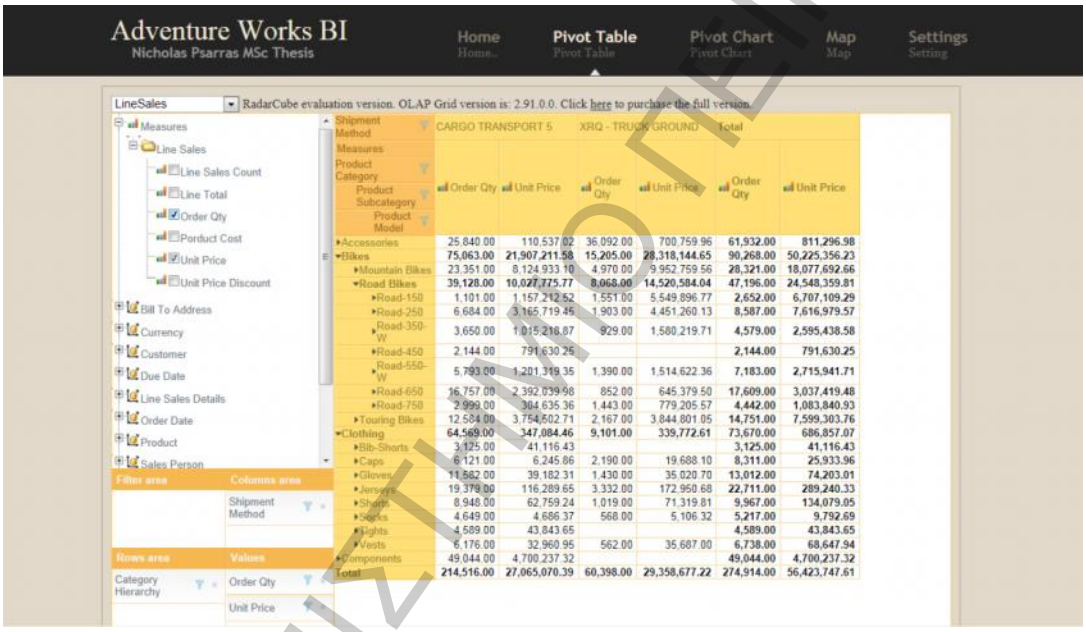
Εικόνα 3.29:Adventure Works BI Portal Default Web Page

PivotTable.aspx:

Στη σελίδα αυτή υπάρχει το Pivot Table Control της RadarSoft μέσω του οποίου οπτικοποιούνται τα δεδομένα του Data Warehouse σε ένα δυναμικό πίνακα. Επιπλέον υπάρχει ένα μη ορατό Control το οποίο ονομάζεται TMDCube και το οποίο συνδέεται στον Analysis, αντλεί τα δεδομένα και ορίζεται ως το Control από το οποία θα λάβει τα δεδομένα που θα οπτικοποιήσει το Pivot Table. Ο χρήστης διαλέγει τον επιθυμητό κύβο από το αντίστοιχο Combo Box και στην αριστερή στήλη εμφανίζονται τα Measures και τα Dimensions. Επιλέγει τα επιθυμητά δεδομένα και αυτά εμφανίζονται στον χώρο του πίνακα (Εικόνα 3.30).

Όσον αφορά την προγραμματιστική λογική της σελίδας (code behind), στο event Page_Load, την πρώτη φορά που εκτελείται ο κώδικας

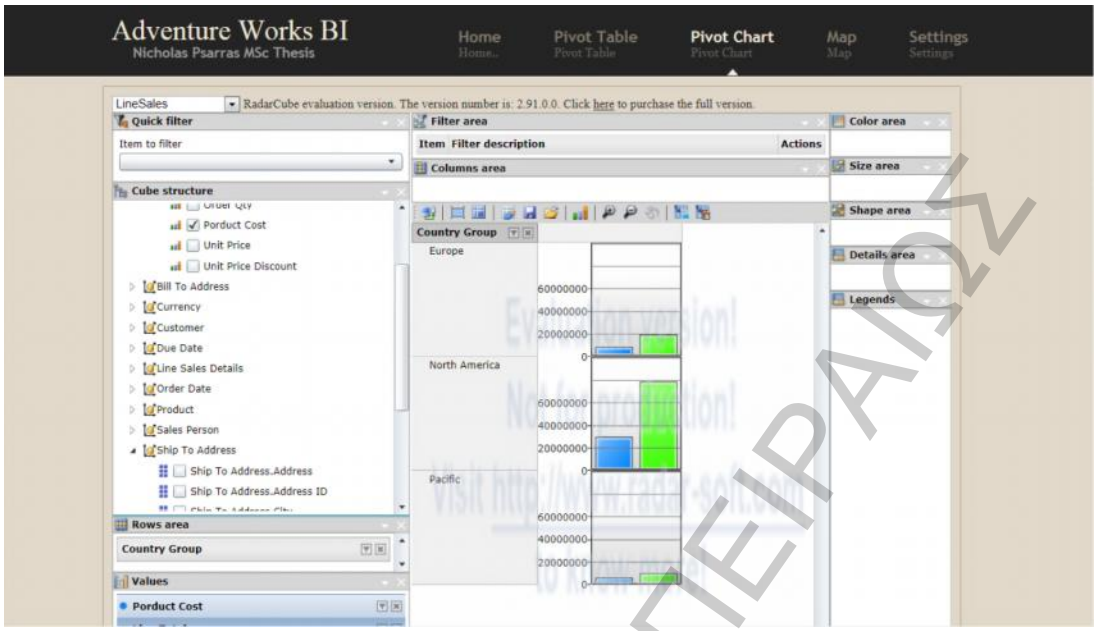
της σελίδας, προσθέτονται τα ονόματα των κύβων στο Combobox, αντλείται και αποθηκεύεται στο ConnectionString του Analysis Server και εφαρμόζεται το επιλεγμένο Skin στο PivotTable. Υπάρχει άλλο ένα Event το οποίο εκτελείται όταν αλλάξει η επιλογή του κύβου στο αντίστοιχο ComboBox. Σε αυτή τη περίπτωση γίνεται η σύνδεση με τον Analysis για το συγκεκριμένο κύβο, ενεργοποιείται ο κύβος και είναι πλέον διαθέσιμος για χρήση.



Εικόνα 3.30: Pivot Table Web Page

PivotChart.aspx:

Η διαφορά με το Pivot Table είναι ότι η οπτικοποίηση γίνεται πλέον όχι σε πίνακα αλλά σε διαγράμματα. Αντί για Pivot Table Control της RadarSoft χρησιμοποιείται το PivotChart RIA Control (βασίζεται στην τεχνολογία του Silverlight).Η προγραμματιστική λογική της σελίδας είναι ακριβώς η ίδια με την λογική του Pivot Table.(Εικόνα 3.31)



Εικόνα 3.31: Pivot Chart Web Page

MapSettings.aspx:

Στο Web Portal υπάρχει η δυνατότητα εμφάνισης στοιχείων και δεδομένων πάνω στον χάρτη. Για να μπορέσει να επιτευχθεί αυτό πρέπει να έχει δημιουργηθεί ένας κύβος ο οποίος να περιέχει εκτός των άλλων measures και τα measures longitude και latitude (αυτές οι δύο τιμές ορίζουν μονοσήμαντα ένα γεωγραφικό σημείο στο χάρτη). Επιπλέον θα πρέπει να υπάρχουν dimension(s) που να αντιστοιχούν στο Location στο οποίο θα παραπέμπουν τα συγκεκριμένα longitude και latitude. Επομένως για να εμφανιστούν επιτυχώς τα δεδομένα πάνω στο χάρτη πρέπει να επιλεγούν αυτές οι παράμετροι. Στη σελίδα αυτή εμφανίζεται αρχικά ένα Combobox με τους διαθέσιμους κύβους έτσι ώστε να επιλεγεί από το χρήστη ο κύβος που έχει αυτή τη συγκεκριμένη δομή, έπειτα εμφανίζεται ένα ListBox στο

οποίο ο χρήστης επιλέγει τα measures που θέλει να οπτικοποιήσει και τέλος εμφανίζεται ένα Combobox στο οποίο ο χρήστης επιλέγει το αντίστοιχο Location Dimension Hierarchy που θέλει. Η ιεραρχία αυτή έχει σαν τελευταίο επίπεδο το στοιχείο στο οποίο αντιστοιχεί το Longitude και Latitude που υπάρχει στα Measures της κάθε εγγραφής. Τέλος, μετά την επιτυχή επιλογή τους γίνεται ανακατεύθυνση της σελίδας στο Map.aspx για να επιτευχθεί η οπτικοποίηση πάνω στο χάρτη. Επιπλέον έχουν οριστεί ένα TMDCube και ένα TOLAPGrid τα οποία είναι βοηθητικά για να μπορέσουμε να αντλήσουμε τις πληροφορίες που χρειαζόμαστε (πχ η λίστα των measures για ένα επιλεγμένο κύβο).

Η προγραμματιστική λογική της σελίδας αρχικά περιλαμβάνει την Page_Load της σελίδας στην οποία αρχικοποιούνται οι τιμές των Controls της σελίδας και το Combobox των κύβων λαμβάνει τις τιμές του. Έπειτα το Event που ενεργοποιείται όταν επιλεγεί ο κύβος (lkpCubes_SelectedIndexChanged) κρύβει το ComboBox και το Label των κύβων και αφού αρχικοποιήσει με τιμές των measures του επιλεγμένου κύβου το αντίστοιχο ListBox, τότε το κάνει ορατό, μαζί με το κουμπί Next. Επιπλέον αποθηκεύεται σε μια Session μεταβλητή η επιλογή του κύβου έτσι ώστε να είναι διαθέσιμη στη σελίδα Map. Όταν ο χρήστης επιλέξει τα measures και πατήσει Next τότε ενεργοποιείται το αντίστοιχο event (btnNext_Click). Εκεί στη λίστα με τα επιλεγμένα Measures προστίθενται και τα Measures Latitude και Longitude (που προϋπάρχουν και είναι υποχρεωτικά) και αφού αποθηκευθούν σε μια Session μεταβλητή, εμφανίζεται το επόμενο Combobox επιλογής Location Dimension Hierarchy και το κουμπί Load Map. Τέλος, όταν ο χρήστης επιλέξει και το

Location Dimension Hierarchy που επιθυμεί και πατήσει το αντίστοιχο κουμπί, τότε ενεργοποιείται το Event butLoadMap_Click.Map.aspx στο οποίο πλέον αποθηκεύεται και η τελευταία Session μεταβλητή με τα Dimensions και φορτώνεται γίνεται ανακατεύθυνση στη σελίδα Maps.aspx για να φορτωθεί ο χάρτης.

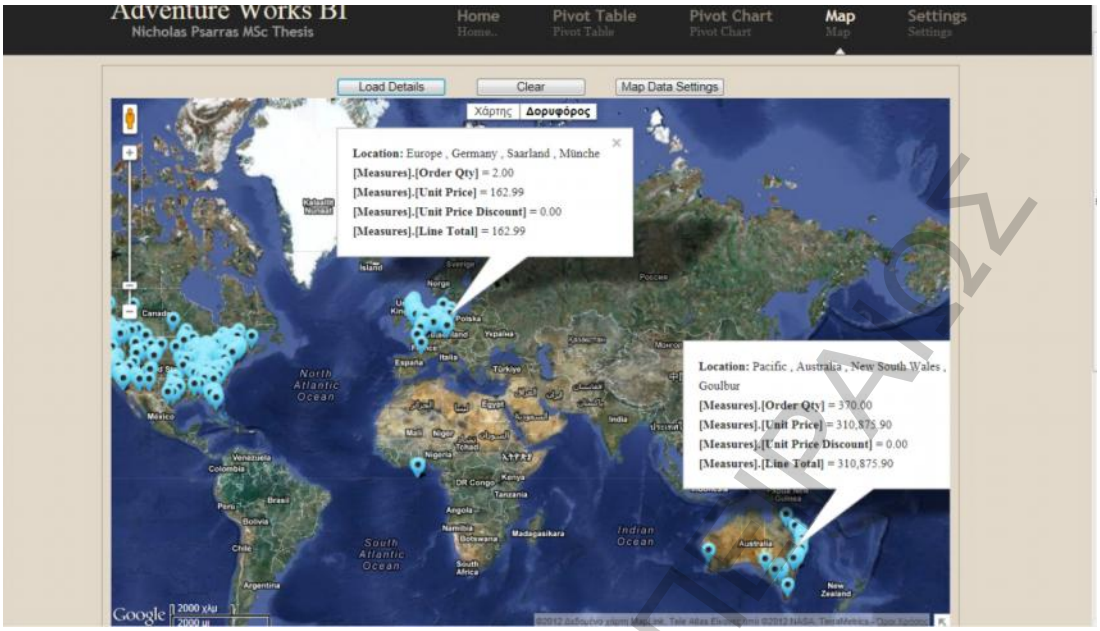
Map.aspx:

Η σελίδα αυτή περιλαμβάνει το χάρτη και 3 κουμπιά, το Load Data, το Clear και το Map Settings. Αρχικά κατά τη σύνδεση εμφανίζεται μόνο ο χάρτης και πρέπει να πατηθεί το κουμπί load markers για να φορτωθούν τα δεδομένα στο χάρτη (σε μορφή markers που όταν γίνει κλικ πάνω τους εμφανίζεται παράθυρο με τις επιθυμητές πληροφορίες). Το κουμπί clear αφαιρεί τους markers από το χάρτη ενώ το κουμπί MapSetting ανακατευθύνει στη σελίδα MapSettings.aspx για να οριστούν άλλες πληροφορίες σχετικά με το χάρτη. Επιπλέον δίνονται όλες οι δυνατότητες ενός Google χάρτη όπως η αλλαγή προβολής σε μορφή χάρτη ή δορυφορικής λήψης, το Google Street View καθώς και η αλλαγή zoom. Τέλος έχουν οριστεί ένα TOLAPGrid και ένα TMDCube τα οποία δεν είναι ορατά αλλά είναι απαραίτητα για να φορτωθούν τα δεδομένα από το Data Warehouse και να εμφανιστούν στο χάρτη (έπειτα από την κατάλληλη επεξεργασία) (Εικόνα 3.32)

Όσον αφορά τη προγραμματιστική λογική της σελίδας, αρχικά ενεργοποιείται το Event Page_Load το οποίο την πρώτη φορά που έχουμε πρόσβαση στη σελίδα ελέγχει αν είναι ορισμένες οι Session μεταβλητές που είναι απαραίτητες για να λειτουργήσει ο χάρτης. Έτσι αν οι τιμές

αυτές δεν έχουν οριστεί τότε γίνεται ανακατεύθυνση στην σελίδα MapSettings.aspx. Το κουμπί Clear ουσιαστικά ξαναφορτώνει τη σελίδα με τις αρχικές της ρυθμίσεις, επομένως δεν υπάρχουν φορτωμένα markers ενώ το κουμπί MapSettings κάνει ανακατεύθυνση στην αντίστοιχη σελίδα.

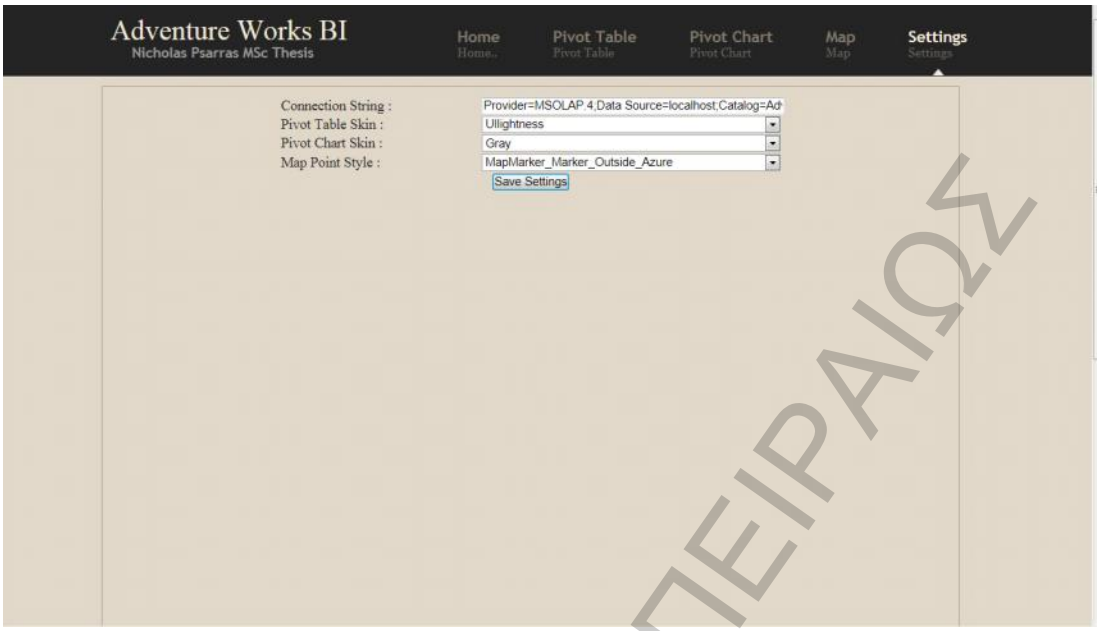
Το κουμπί Load Markers ενεργοποιεί ένα Event που ονομάζεται btnLoadMarkers_Click στο οποίο ουσιαστικά υλοποιείται όλη η προγραμματιστική λογική για να φορτωθούν τα δεδομένα στο χάρτη. Αρχικά λαμβάνονται και αποθηκεύονται οι Session μεταβλητές και έπειτα πραγματοποιείται σύνδεση στον επιλεγμένο κύβο του Analysis Server. Έπειτα φορτώνονται τα Measures στο μη ορατό Pivot Table και αφαιρούνται τα Totals. Το επόμενο βήμα είναι να φορτωθεί στο Pivot Table η επιλεγμένη Location Dimension Hierarchy. Υπενθυμίζεται ότι το τελευταίο Level της ιεραρχίας αντιστοιχεί στα Measures Longitude και Latitude της κάθε εγγραφής και γι' αυτό το λόγο γίνεται Expand η ιεραρχία του Pivot Table μέχρι το τελευταίο επίπεδο (Address). Πλέον το Pivot Table είναι στην επιθυμητή μορφή και γίνεται κλήση μιας μεθόδου του Pivot, της GetOlapData η οποία αποθηκεύει σε ένα DataTable σε μορφή Flat Data όλα τα δεδομένα που είναι φορτωμένα τη δεδομένη στιγμή στο Pivot Table. Τέλος, επόμενο στάδιο περιλαμβάνει τη δημιουργία Marker για κάθε εγγραφή του Data Table, ο ορισμός της θέσης του Marker (μέσω των Measures Longitude και Latitude που είναι προ-φορτωμένα), ο ορισμός των πληροφοριών που θα δείχνει το κάθε marker καθώς και του εικονιδίου του και η τοποθέτηση του Marker στο χάρτη. Το τελικό αποτέλεσμα φαίνεται στην εικόνα.



Εικόνα 3.32: Map Web Page

Settings.aspx: Στη σελίδα αυτή δίνεται η δυνατότητα στον χρήστη να αλλάξει τις ρυθμίσεις που υπάρχουν στο Settings.xml και να τις αποθηκεύσει σε αυτό. Οι ρυθμίσεις είναι το εικονίδιο του Marker του χάρτη, το Connection String σύνδεσης με τον Analysis Server καθώς και τα Skins των Pivot Table και Pivot Chart αντίστοιχα (Εικόνα 3.33).

Η προγραμματιστική λογική περιλαμβάνει το event Page_Load στο οποίο φορτώνονται οι αποθηκευμένες τιμές του Settings.xml μέσω των αντίστοιχων μεθόδων του WebAppHelper στα αντίστοιχα Controls και το Event btnSaveSettings_Click το οποίο αποθηκεύει τις τιμές των Controls στο Settings.xml



Εικόνα 3.33: Settings Web Page

Με την τελευταία περιγραφή ολοκληρώνεται η ανασκόπηση και επεξήγηση της υλοποίησης ολόκληρου του Business Intelligence συστήματος της εταιρίας Adventure Works.

4. Συμπεράσματα - Προτάσεις

4.1. Συμπεράσματα

Η παραδοσιακή προσέγγιση της ανάλυσης δεδομένων μέσω στατικών δεν μπορεί να ικανοποιήσει τις ολοένα και αυξανόμενες απαιτήσεις των υπευθύνων λήψεων αποφάσεων των εταιριών ως προς το όγκο, τη μορφή και την ταχύτητα άντλησης των επιθυμητών δεδομένων. Τα συστήματα επιχειρηματικής ευφυΐας προσφέρουν τα εργαλεία που χρειάζονται μιας και αποτελούν ένα χρήσιμο και λειτουργικό εργαλείο στα χέρια των υπευθύνων λήψεων αποφάσεων της κάθε εταιρίας. Μετατρέπουν τον μεγάλο όγκο των δεδομένων μιας εταιρίας σε πληροφορία την οποία παρέχουν ένα δυναμικό τρόπο εμφάνισης, στην κατάλληλη μορφή και λεπτομέρεια και σε τόσο ικανοποιητικό χρόνο ώστε να υποβοηθηθεί η διαδικασία της ανάλυσης και της αξιολόγησης των πληροφοριών, της λήψης αποφάσεων καθώς και της πραγματοποίησης προβλέψεων.

Το βασικό στοιχείο ενός συστήματος BI είναι το Data Warehouse το οποίο αποτελεί την πηγή των δεδομένων ανάλυσης και το οποίο αντλεί δεδομένα από διαφορετικές πηγές. Τα δεδομένα αυτά πρώτα συλλέγονται και έπειτα υφίστανται επεξεργασία και μετασχηματισμούς ώστε να έρθουν στην κατάλληλη μορφή που μπορεί να παρέχει γρήγορη και αξιόπιστη ανάλυση. Το επόμενο στάδιο είναι η δημιουργία κύβων οι οποίοι συσχετίζουν τα δεδομένα μεταξύ τους και προ-εκτελούν επεξεργασία των δεδομένων με μαθηματικές πράξεις με στόχο την γρήγορη πρόσβαση σε αυτά κατά τη διάρκεια της ανάλυσής τους, μιας

και πολλές πράξεις που θα πρέπει να γίνουν θα είναι ήδη υλοποιημένες. Η πρόσβαση στα δεδομένα του Data Warehouse μπορεί να επιτευχθεί είτε από έτοιμες λύσης που προσφέρουν εταιρίες πληροφορικής ανά τον κόσμο (Microsoft Excel, Microsoft SharePoint, Cognos BI, Microstrategy κ.α.) είτε μέσω Custom λύσεων που προσαρμόζονται στις ανάγκες του οργανισμού που εξυπηρετείται. Η οπτικοποίηση των δεδομένων γίνεται σε μορφή φιλική προς τον χρήστη, μιας και μπορούν να εμφανίζονται σε δυναμικούς πίνακες Pivot Tables, σε δυναμικά διαδραστικά διαγράμματα, σε κουκίδες πάνω σε χάρτη και γενικότερα σε μορφή που βοηθά τον χρήστη να χρησιμοποιήσει τις πληροφορίες που του παρέχονται για ανάλυση, εξαγωγή συμπερασμάτων, λήψη αποφάσεων ακόμα και εκτίμηση προβλέψεων.

4.2. Προτάσεις – Βελτιώσεις

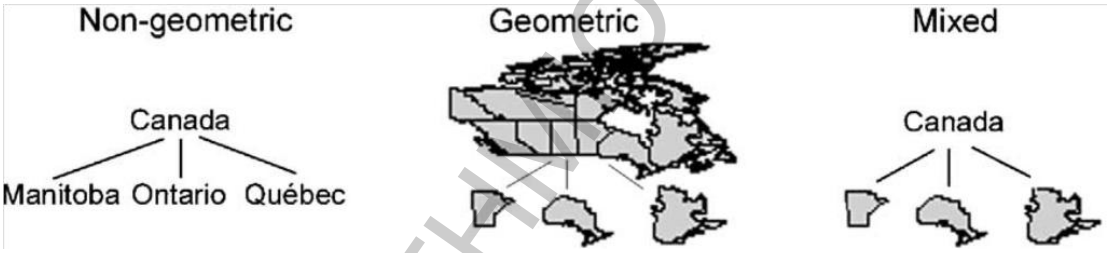
Στην παρούσα ενότητα αναπτύσσονται κάποιες ιδέες – προτάσεις χρήσης του BI οι οποίες σε αρκετές περιπτώσεις έχουν αρχίσει να μελετώνται και να υλοποιούνται.

4.2.1. Χρήση BI σε Γεωγραφικά Δεδομένα

Υπάρχουν έρευνες και μελέτες γύρω από το πώς θα μπορούσαν να χρησιμοποιηθούν τα οφέλη των BI συστημάτων σε συστήματα γεωγραφικών δεδομένων. Η λύση η οποία προτείνεται, σύμφωνα με το (13) είναι τα Spatial OLAP συστήματα. Αποτελούνται από μια multidimensional spatio-temporal βάση δεδομένων, έναν SOLAP Server και ένα SOLAP Client. Η βάση δεδομένων περιέχει τα γεωμετρικά/

γεωπολιτικά δεδομένα συνδυασμένα με Dimensions και Members. Ο Server πραγματοποιεί τους απαραίτητους υπολογισμούς για τους συνδυασμούς Measure-Dimensions όπως και ο OLAP Server, ενώ ο Client παρέχει πρόσβαση στα χωρικά δεδομένα των DB και τα εμφανίζει σε χάρτες, διαγράμματα και πίνακες. Τα Dimensions μπορεί να μην έχουν χωρικά δεδομένα (πχ ονόματα πόλεων) Non-geometric, να έχουν δεδομένα ενός σημείου ή να οριοθετούν μια περιοχή Geometric ή να περιέχουν και τις δυο παραπάνω πληροφορίες Mixed (Εικόνα 4.1).

Τα measures είναι γεωγραφικά δεδομένα όπως για παράδειγμα απόσταση, εμβαδό, αριθμός γειτονικών σημείων κτλ.



Εικόνα 4.1: Κατηγορίες Δεδομένων των SOLAP Dimensions (13)

4.2.2. Real Time BI

Οι αυξανόμενες απαιτήσεις σε ταχύτητα καθώς και το μεγάλο μέγεθος των δεδομένων/ πληροφοριών που εισέρχονται στις βάσεις δεδομένων ενός οργανισμού – εταιρίας κάνουν επιτακτική την ανάγκη της real-time και ad-hoc γρήγορης πρόσβασης στα κατάλληλα δεδομένα από τους κατάλληλους ανθρώπους όποτε τα ζητήσουν. Αυτό αποτελεί το real-time bi. Μέσα στη διάρκεια μιας ημέρας μπορεί να αλλάζουν οι τιμές των προϊόντων ή οι πολιτικές πώλησης της εταιρίας ενώ οι υπάλληλοι

μπορεί να βρίσκονται σε διαφορετικές τοποθεσίες εκτός της εταιρίας και να πρέπει να έχουν πρόσβαση σε αυτά. Επιπλέον, κάποια reports που δημιουργούνται κατά τη διάρκεια της ημέρας πρέπει να είναι προσβάσιμα από υπαλλήλους οι οποίοι βρίσκονται σε διαφορετικές τοποθεσίες εκτός του εταιρικού LAN δικτύου. Επομένως είναι επιτακτική η ανάγκη χρήσης Real-Time BI όπως και η εύκολη πρόσβαση σε αυτή τη πληροφορία μέσω του διαδικτύου με τρόπο ασφαλή. (12)

Το μέλλον των κλασσικών BI Συστημάτων είναι τα Real Time BI systems. Τα χαρακτηριστικά του Real-Time BI, σύμφωνα με το (12), είναι τα εξής:

- Μηδενική καθυστέρηση στην ανάκτηση των πληροφοριών
- Τα δεδομένα να είναι προσβάσιμα οποτεδήποτε χρειάζεται.
- Παροχή custom δεδομένων και όχι προκαθορισμένα reports.
- Τα δεδομένα του Data Warehouse ανταποκρίνονται στα πιο πρόσφατα δεδομένα και όχι σε δεδομένα που κάποιας χρονικής στιγμής στο παρελθόν.

Ερευνητικά, υπάρχει η λύση του RTBI το οποίο έχει την ίδια λειτουργικότητα με το παραδοσιακό BI με τη διαφορά ότι έχει πρόσβαση σε δεδομένα τα οποία εξάγονται Real-Time από τις παραγωγικές βάσεις δεδομένων. Ένα Real Time Business Intelligence System θα απαρτίζεται από (12):

- Στατικά Data Warehouses και data Shopping malls τα οποία θα παραμετροποιούνται δυναμικά από τους χρήστες που θα διαλέγουν τις πηγές δεδομένων που χρειάζονται.

- Πληροφορίες Meta-Data για τα δεδομένα.
- Taxonomies και ontologies οι οποίες θα περιγράφουν τα περιεχόμενα και θα παρέχουν Semantic Content πληροφορίες.
- Προηγμένα εργαλεία ETL τα οποία θα αντλούν και θα τροφοδοτούν με data τα εργαλεία ανάλυσης. Εκτός των Structured Data θα μπορεί να γίνει extract – transform και load σε Unstructured Data (πχ e-mail, reports, τηλεφωνήματα κτλ).
- Μηχανισμούς ανάδρασης σε operational συστήματα.

Οι τελικοί χρήστες θα έχουν τη δυνατότητα να διαλέξουν δεδομένα χωρίς να γνωρίζουν από ποια Data Sources προέρχονται. Επιπλέον η μετατροπή των δεδομένων σε πληροφορία θα γίνεται με βάση τις προτιμήσεις του χρήστη που τις αντλεί.

Γενικότερα το μέλλον των BI Συστημάτων είναι ένα σύστημα το οποίο θα μπορεί να διαχειριστεί σε πραγματικό χρόνο μεγάλο όγκο δεδομένων και με έξυπνες μεθόδους θα προσαρμόζει τα αποτελέσματα στις απαιτήσεις και τις ανάγκες του χρήστη, δίδοντάς τα σε κατάλληλη μορφή. Με αυτό τον τρόπο θα μπορέσει να διευρυνθεί η χρήση του και σε αντικείμενα πέραν των επιχειρήσεων.

4.2.3. Real Time αναζήτηση στο Web

Σύμφωνα με το (32) παγκοσμίως παράγονται περίπου 635.000 έως 2.120.000 Terabyte μοναδικής πληροφορίας ανά χρόνο και η πλειοψηφία αυτής αποθηκεύεται σε σκληρούς δίσκους υπολογιστών και server.

Πολλές από αυτές τις συσκευές αποτελούν κομμάτια του Internet και παρέχουν ικανοποιητική πρόσβαση σε πληροφορίες. Εκτός όμως από την αποθήκευση, είναι εξίσου σημαντικό να μπορέσουμε να ανακτήσουμε ένα μέρος της πληροφορίας αυτής το οποίο μας ενδιαφέρει. Όταν ένας χρήστης αναζητά μια πληροφορία πρέπει με κάποιο τρόπο να φιλτραριστεί ο μεγάλος όγκος των δεδομένων στον οποίο γίνεται η αναζήτηση και να εμφανιστούν στο χρήστη μόνο η επιθυμητή πληροφορία. Τις περισσότερες φορές τα αποτελέσματα που πραγματικά σχετίζονται με την αναζήτησή μας είναι διασκορπισμένα ανάμεσα σε άλλες πληροφορίες οι οποίες δεν μας είναι χρήσιμες (overload). Αυτό είναι ένα μεγάλο πρόβλημα και από την πλευρά του χρόνου και της προσπάθειας που χρειάζεται να καταβάλει ο χρήστης για να απομονώσει την χρήσιμη πληροφορία άλλα και από την πλευρά της χρήσης των πόρων του διαδικτύου που οδηγεί σε επιβάρυνση του.

Η τεχνολογία του Business Intelligence μπορεί να χρησιμοποιηθεί έτσι ώστε να βελτιωθεί αυτή η λειτουργία της αναζήτησης στο Web, ιδιαίτερα όταν τα δεδομένα είναι Real Time. Σύμφωνα με το (12), οι μηχανές αναζήτησης θα μπορέσουν να χρησιμοποιήσουν τη πληροφορία των Data Warehouses έτσι ώστε να παρέχουν πρόσβαση σε real time δεδομένα τα οποία θα προσαρμόζονται στις ανάγκες του χρήστη. Οι crawlers θα περισυλλέγουν δεδομένα τα οποία θα μπορούν με προσαρμοστικό τρόπο να μετατραπούν σε πληροφορία συνεχώς ανανεωμένη και προσαρμοσμένη στις ανάγκες του χρήστη. Επιπλέον ο τρόπος εμφάνισης των αποτελεσμάτων θα μπορεί να είναι κάτι διαφορετικό από λίστες δεδομένων. Αν επιτευχθεί η εισαγωγή δεδομένων

από unstructured data θα είναι δυνατή η αναζήτηση στοιχείων σε πηγές δεδομένων οι οποίες αυτή τη στιγμή δεν είναι δυνατό να αξιοποιηθούν.

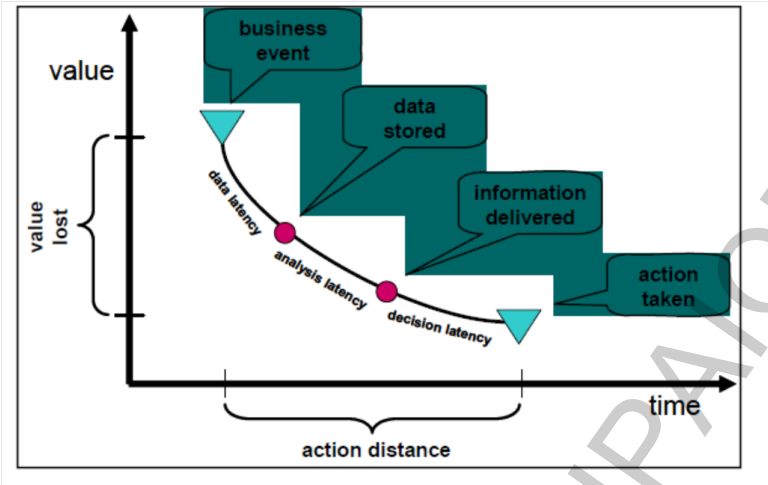
Η διαφορά με τα παραδοσιακά BI (32) είναι ότι τα αποτελέσματα αντί για εταιρικά δεδομένα θα μπορούν να είναι πληροφορίες που βρίσκονται στον αχανή χώρο του διαδικτύου. Αυτό σχετίζεται με τις τεχνικές αναζήτησης και συσχέτισης των δεδομένων (μέσω Data Warehouses) αλλά και τον τρόπο παρουσίασης των αποτελεσμάτων (trees, maps, γεωμετρικά σχήματα, charts κτλ σε αντίθεση με την λίστα υπερσυνδέσεων). Στην αναζήτηση που έχουμε συνηθίσει μέχρι σήμερα, ουσιαστικά βλέπουμε τα φύλλα χαμηλότερου επιπέδου ενός δέντρου που συσχετίζεται με την αναζήτησή μας. Ίσως όμως να βοηθούσε η εμφάνιση μιας ιεραρχικής δενδρικής δομής αυτών των δεδομένων μέσω της οποίας ο χρήστης θα μπορούσε να οδηγηθεί στην ομάδα αποτελεσμάτων που αναζητεί. Ή ο χρήστης να μπορούσε να δει σε ένα Chart τα Site στα οποία κατέληξαν χρήστες που έκαναν παρόμοιες αναζητήσεις.

Σύμφωνα με το (33) για πολλούς το Real-Time είναι συνώνυμο του στιγμιαίου. Στην περίπτωση του Real Time BI σε πολλές περιπτώσεις δε μπορούμε να έχουμε Real Time δεδομένα γιατί ακόμα και οι παραγωγικές βάσεις δεδομένων μπορεί να ανανεώνονται κάθε μήνα, ή μπορεί το κόστος της δημιουργίας δεδομένων σε πραγματικό χρόνο να είναι απαγορευτικό για μια εταιρία. Στις περισσότερες περιπτώσεις δε, η σπουδαιότητα των δεδομένων μειώνεται δραματικά όσο τα δεδομένα αυτά γίνονται πιο παλιά. Δηλαδή τα πιο πρόσφατα δεδομένα έχουν περισσότερη αξία από τα παλαιότερα, γι' αυτό και έχει αρχίσει να γίνεται

επιτακτική η ανάγκη του Real Time BI. Ο Richard Hackathorn κατηγοριοποιεί την χρονική καθυστέρηση σε 3 είδη (Εικόνα 4.2):

- Data Latency: Είναι ο χρόνος μεταξύ την χρονικής στιγμής που συμβαίνει το γεγονός και την χρονικής στιγμής που αυτό αποθηκεύεται στο Data Warehouse.
- Analysis Latency: Είναι η χρονική διάρκεια που οριοθετείται από τη στιγμή που έχουν αποθηκευτεί τα Δεδομένα μέχρι την στιγμή που αναλύονται.
- Decision Latency: Είναι ο χρόνος που χρειάζεται από την στιγμή που έχει γίνει η ανάλυση της πληροφορίας μέχρι τη στιγμή που λαμβάνεται κάποια δράση σχετικά με αυτή.

Το άθροισμα των τριών αυτών καθυστερήσεων αποτελεί τη συνολική καθυστέρηση (Total Latency) . Η μείωση των Data και Analysis Latency εξαρτάται άμεσα από τεχνικούς παράγοντες και σε αυτού του είδους τις καθυστερήσεις συνεισφέρει το Real Time Data Warehouse. Το Decision Latency εξαρτάται από τον ανθρώπινο παράγονται και το κατά πόσο χρησιμοποιούνται οι πληροφορίες στη λήψη αποφάσεων. Το να παρέχονται πιο σύγχρονα δεδομένα δεν έχει τόση σημασία όσο τη δυνατότητα να μπορούν να οδηγήσουν σε σωστές και καίριες αποφάσεις. Αυτός ακριβώς είναι και ο λόγος που άμεση προτεραιότητα των εταιριών είναι η μείωση του Decision Latency. Με τα BI συστήματα μειώνουμε την χρονική απόσταση μεταξύ της αποθήκευσης στην βάση δεδομένων και της λήψης της κατάλληλης απόφασης.



Εικόνα 4.2:Τα 3 Είδη Χρονικής Καθυστέρησης Στη Λήψη Αποφάσεων (33)

Στα Real Time BI πρέπει το ETL να γίνεται όσο το δυνατόν πιο αυτοματοποιημένα με τι μικρότερη δυνατή παρέμβαση ανθρώπινου παράγοντα. Έτσι το σύστημα πρέπει να μπορεί να υλοποιεί τις εσωτερικές διαδικασίες μόνο του αλλά και να γνωρίζει πότε και ποιον πρέπει να ειδοποιήσει (Alert) όταν απαιτείται η ανθρώπινη παρέμβαση. Επιπλέον πρέπει στην εταιρία που θα το εγκαταστήσει το κόστος του συστήματος να είναι πολύ μικρότερο από το κέρδος που θα της αποφέρει. Σε επόμενο στάδιο, μετά από ανάλυση των δεδομένων θα πρέπει να αυτοματοποιηθεί όσο το δυνατόν περισσότερο η διαδικασία ETL ώστε τα δεδομένα να φτάνουν από τα operational databases στους κύβους που προσφέρονται για ανάλυση. Επιπλέον θα πρέπει να γίνει ένας καλός διαχωρισμός των υπο-συστημάτων που απαιτείται να είναι Real Time και αυτών που θα μπορούσαν να μην είναι.

Παράδειγμα Χρήσης RTBI στην αεροπορική εταιρία Continental (33)

Μια εταιρία που χρησιμοποιεί πιλοτικά Real Time Data Warehousing είναι η αεροπορική εταιρία Continental Airlines. Για μια αεροπορική εταιρία είναι πολύ σημαντικό να έχει γρήγορη πληροφόρηση πριν πετάξει το αεροπλάνο. Η πληροφόρηση για τις άδειες θέσεις του αεροσκάφους πριν την απογείωση του αεροπλάνου μπορεί να οδηγήσει σε μια γρήγορη πολιτική μείωσης των τιμών έτσι ώστε να καλυφθεί. Από τη στιγμή που θα απογειωθεί και μετά είναι μια πληροφορία που χρησιμεύει μόνο για τον υπολογισμό την ζημιάς. Επιπλέον τα Real Time Data μπορούν να χρησιμεύσουν σε λειτουργικά προβλήματα όπως και σε Customer Relationship Management (CRM) φύσεως θέματα, όπως για παράδειγμα η καλύτερη εξυπηρέτηση πελατών υψηλής αξίας από την ώρα που κλείνουν το εισιτήριο μέχρι την ώρα που επιβιβάζονται στο αεροσκάφος.

4.2.4. Εφαρμογή BI στον τομέα της Υγείας

Η εφαρμογή BI Συστημάτων στην υγεία θα πρέπει να μπορεί να διευκολύνει τη διοίκηση και του κλινικού τμήματος αλλά και του διοικητικού τμήματος ενός νοσηλευτικού ιδρύματος. Η εφαρμογή σε ένα τέτοιο πεδίο απαιτεί ειδικές συνθήκες ώστε να μπορέσει να λειτουργήσει σωστά. Πρέπει τα δεδομένα να είναι Real Time και να υπάρχουν πηγές δεδομένες οι οποίες θα είναι διαθέσιμες και προσπελάσιμες από όλους τους φορείς υγείας, έτσι ώστε να υπάρχει μια καθολική πληροφόρηση. Επιπλέον, επειδή τα δεδομένα είναι ευαίσθητα θα πρέπει να παρέχεται ασφάλεια σε πολλά επίπεδα έτσι ώστε να διασφαλιστεί η ορθή χρήση των

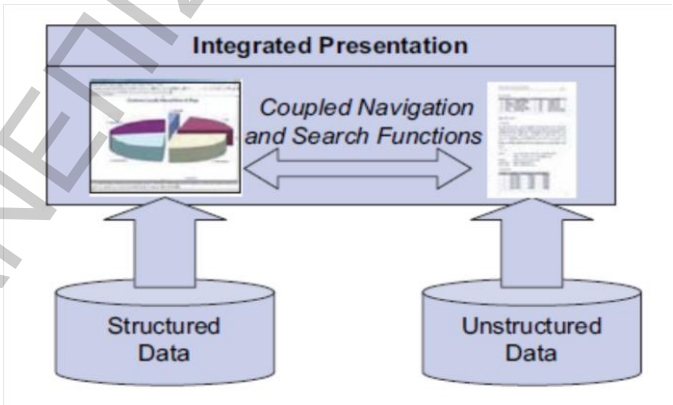
δεδομένων μόνο από τους αρμόδιους. Έτσι θα μπορέσουμε να επιτύχουμε όλα τα πλεονεκτήματα του BI Και σε επίπεδο οικονομικό και διοικητικό αλλά και σε κλινικό επίπεδο, αυξάνοντας τις επιδόσεις σε επίπεδο υγείας, αλλά και φροντίζοντας την οικονομική σταθερότητα και ανάπτυξη των ιδρυμάτων υγείας. (1)

4.2.5. Χρήση BI για ανάλυση Unstructured Data

Structured Data είναι τα δεδομένα τα οποία είναι εκχωρημένα σε πεδία τέτοιου τύπου ώστε να μπορούν να υφίστανται άμεση επεξεργασία από τον υπάρχον υπολογιστικό εξοπλισμό. Τα συστήματα BI χρησιμοποιούν Structured Data μιας και οι πηγές δεδομένων τους αποθηκεύουν τέτοιου είδους δεδομένα. (34) Τα Unstructured Data είναι τα δεδομένα που δεν έχουν κατάλληλη προκαθορισμένη δομή και μορφή ώστε να αποθηκευτούν και να ενταχθούν στη σχεσιακή λογική των παραδοσιακών βάσεων δεδομένων. Παραδείγματα Unstructured Data είναι τα βιβλία, άρθρα, έγγραφα, e-mail, αρχεία εικόνων, ήχου (ηχογραφήσεις συνομιλιών), βίντεο κτλ (35). Τα Semi-Structured Data περιέχουν κάποια δομή η οποία είναι τέτοιας μορφής ώστε να μην επιτρέπει την ένταξή τους στις σχεσιακές βάσεις δεδομένων. Παραδείγματα τέτοιων δεδομένων είναι αρχεία xml καθώς και όλες οι Markup γλώσσες, τα e-mail (όχι μόνο το body) κτλ. Το πρόβλημα δεν εντοπίζεται στο ότι τα δεδομένα αυτά δε μπορούν να αποθηκευτούν στις βάσεις δεδομένων αλλά στο ότι δε μπορούν να ταξινομηθούν, να κατηγοριοποιηθούν και να συσχετιστούν ώστε να μπορέσουμε να

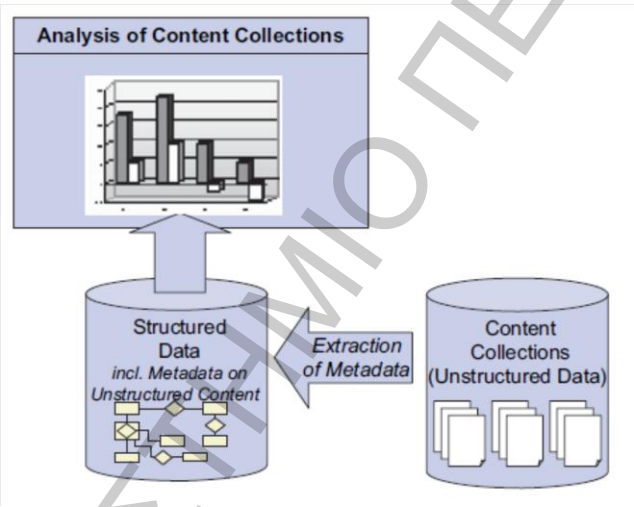
εξάγουμε πληροφορίες από αυτά (36). Οι ολοένα και αυξανόμενες απαιτήσεις έχουν αρχίσει να ορίζουν ως επιτακτική ανάγκη την ανάλυση Unstructured και Semi-structured Data καθώς και συνδυασμού αυτών. (34) Αποτελέσματα έρευνας έδειξαν ότι το 60% των CIO και CTO θεωρούν πως τα Semi-structured και Unstructured Data είναι απαραίτητη και χρήσιμη πληροφορία που θα βοηθήσει στην λήψη βέλτιστων αποφάσεων μέσω της ανάλυσής τους (36). Η πρόκληση είναι εύρεση εργαλείων μετατροπής των Unstructured και Semi-structured Data σε Structured καθώς και ο τρόπος οπτικοποίησης των αποτελεσμάτων. Σύμφωνα με το (34) , παραδείγματα προσεγγίσεων Unstructured Data είναι τα ακόλουθα:

- Integrated Presentation: Η πρόσβαση σε όλα τα δεδομένα πραγματοποιείται από ένα ενσωματωμένο User Interface. Έτσι για τα structured data που επιλέγει ένας χρήστης γίνεται μια αναζήτηση στα unstructured και τα αποτελέσματα φαίνονται το ένα δίπλα στο άλλο. Έτσι ο χρήστης μέσω ενός UI έχει πρόσβαση σε όλων των ειδών τα Data. (Εικόνα 4.3)



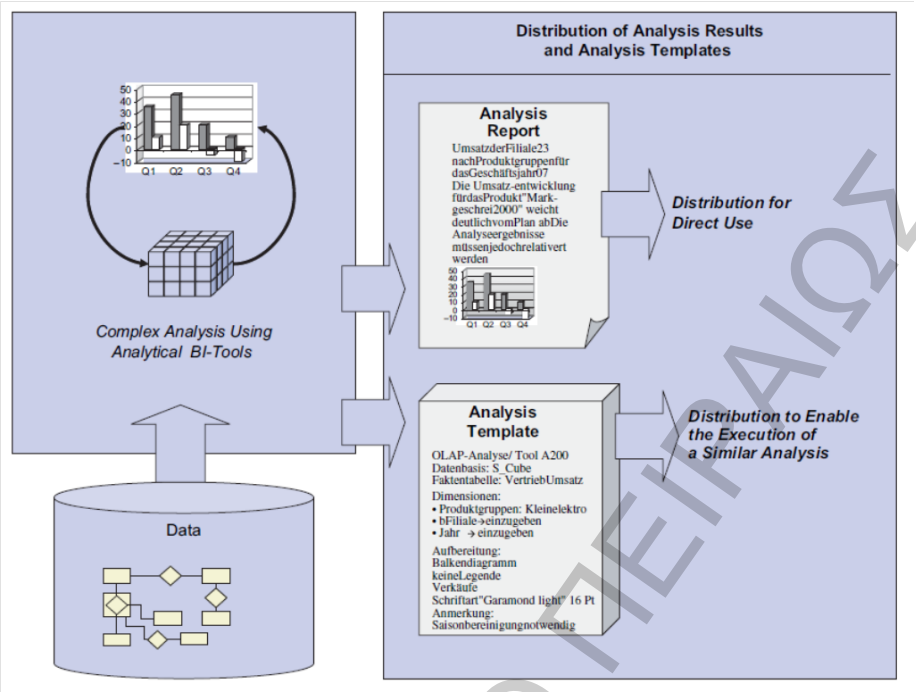
Εικόνα 4.3: Προσέγγιση Integrated Presentation για Unstructured Data (34)

- Analysis of Content: Η προσέγγιση αυτή βασίζεται στην μετατροπή των Unstructured Data σε Structured με την προσθήκη metadata: Τα Metadata πεδία χρησιμοποιούνται για κατηγοριοποίηση και επομένως μπορούν να υπολογιστούν σαν Dimensions. Επομένως μπορούμε να χρησιμοποιήσουμε ιδιότητες των Unstructured είτε ως Measures είτε ως Dimensions, όπως δηλαδή οποιοδήποτε Data Source. Μπορούν να χρησιμοποιηθούν και Text Mining τεχνικές για αυτόματη εξαγωγή Metadata. (Εικόνα 4.4)



Εικόνα 4.4: Προσέγγιση Analysis Content για Unstructured Data (34)

- Distribution of Analysis Results and Analysis Templates: Η τεχνική αυτή αναλύει τα δεδομένα της ανάλυσης που προκύπτει από ένα BI σύστημα. Δηλαδή πηγαίνουμε ένα βήμα πιο κάτω και προσπαθούμε τα αποτελέσματα της BI ανάλυσης να είναι είσοδος σε πιο πολύπλοκα συστήματα Knowledge Management τα οποία συνδυάζουν και στοιχεία από Unstructured δεδομένα ώστε να κρίνουν κατά πόσο η ανάλυση αυτή είναι η βέλτιστη και μπορεί να οδηγήσει στο επιθυμητό αποτέλεσμα. (Εικόνα 4.5)



Εικόνα 4.5: Προσέγγιση Distribution of Analysis Results and Analysis Templates για Unstructured Data (34)

Βιβλιογραφία

1. *Tobias Mettler, Vivian Vimarlund - Understanding Business Intelligence in the Context of Health Care (2008).*

2. *Luhn H P. - A business intelligence system. IBM Journal of Research and Development (1958).*

3. *Brian Larson - Delivering Business Intelligence with Microsoft SQL Server 2008 (2009).*

4. *Arnott, D., & Pervan, G. - A critical analysis of decision support systems research.*

5. *Okkonen, J., Pirttimäki, V., Hannula, M., & Lönnqvist, A. - Triangle of Business Intelligence, Performance Measurement and Knowledge Management.*

6. *Moss, L. T., & Atre, S. - Business Intelligence Roadmap: The Complete Project Lifecycle for Decision-Support Applications.*

7. *Raisinghani, M. S. - Business Intelligence in the Digital Economy: Opportunities, Limitations, and Risks: Idea Group Pub (2004).*

8. *Lonnqvist, A., & Pirttimaki, V. - The measurement of business intelligence. Information Systems Management (2006).*

9. *Williams, S., & Williams, N. - The Profit Impact of Business Intelligence (2007).*

10. *Aleš Popovič, Tomaž Turk, Jurij Jaklič - CONCEPTUAL MODEL OF BUSINESS VALUE OF BUSINESS INTELLIGENCE SYSTEMS (2010).*

11. *Celina M. Olszak and Ewa Ziemba - Business Intelligence Systems in the Holistic Infrastructure Development Supporting Decision-Making in Organisations (2006).*

12. *B Azvine, Z Cui, D D Nauck and B Majeed - Real Time Business Intelligence for the Adaptive Enterprise.*

13. *S. Rivest, Y. Bédard, M-J Proulx, M. Nadeau, F. Hubert, J. Pastor - SOLAP technology: Merging business intelligence withgeospatial technology for interactive spatio-temporal exploration and analysis of data (2005).*

14. *E. Turban - Decision Support and Business Intelligence Systems.*

15. *L. T. Moss - Business Intelligence Roadmap. The Complete Project.*

16. *N. Egger - SAP BW Data Modeling. Fort Lee: SAP Press, (2005).*

17. *Kimball, R. & Ross, M. - The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling.*

18. *Celina M. Olszak and Ewa Ziemba - Approach to Building and Implementing Business Intelligence Systems.*

19. *http://www.microsoft.com/en-us/bi/LearningCenter/resources.aspx.*

20. *http://help.sap.com/saphelp_nw70/helpdata/en/b2/e50138fede083de10000009b38f8cf/content.htm.*

21. *https://www.ibm.com/developerworks/mydeveloperworks/blogs/ibm-bi-capabilities/entry/ibm_cognos_10_bi_components_user_interfaces1?lang=en.*

22. *http://msdn.microsoft.com/en-us/library/ms124825(v=sql.100).aspx.*

23. *http://msdn.microsoft.com/en-us/library/ms124824(v=sql.100).aspx.*

24. *http://msdn.microsoft.com/en-us/library/ms124670(v=sql.100).aspx.*

25. *http://msdn.microsoft.com/en-us/library/ms124785(v=sql.100).aspx.*

26. *http://msdn.microsoft.com/en-us/library/ms124499(v=sql.100).aspx.*

27. *http://www.getfreewebdesigns.com/preview/?template=866.*

28. *http://www.instantshift.com/2010/10/13/clean-vcard-a-free-professional-xhtml-css-vcard-template/.*

29. *http://www.icons-land.com/vista-map-markers-icons.php.*

30. *http://msftdbprodsamples.codeplex.com/downloads/get/478216.*

31. *http://icodesnip.com/snippet/sql/get-everything-for-date-sql-code-snippets.*

32. *W. CHUNG, H. CHEN, JAY F. NUNAMAKER JR.- A Visual Framework for Knowledge Discovery on the Web: An Empirical Study of Business Intelligence Exploration (2005).*

33. *H.J. Watson, B.H. Wixom, J.A. Hoffer, R.Anderson-Lehman, A.M. Reynolds - Real-time Business Intelligence: Best Practices at Continental Airlines (2006).*

34. *Henning Baars, Hans George Kemper - Management Support with Structured and Unstructured Data - An Integrated Business Intelligence Framework.*

35. *http://www.information-management.com/issues/20040901/1009161-1.html .*

36. *Solomon Negash - BUSINESS INTELLIGENCE (2004).*