ΑΠΟΔΟΤΙΚΗ ΑΠΟΤΙΜΗΣΗ ΕΡΩΤΗΣΕΩΝ OLAP

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από την

Χαρά Παπαγεωργίου

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΟ ΛΟΓΙΣΜΙΚΟ

Ιούλιος 2009

ΑΦΙΕΡΩΣΗ

Την εργασία αυτή την αφιερώνω στην μητέρα μου, που τα τελευταία χρόνια και κυρίως κατά τη διάρκεια των σπουδών μου με στήριξε και μου συμπαραστάθηκε περισσότερο από ό,τι θα μπορούσε μία μεγάλη και ολοκληρωμένη οικογένεια να κάνει....

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ τον επιβλέποντα καθηγητή μου Παναγιώτη Βασιλειάδη, επίκουρο καθηγητή που Τμήματος Πληροφορικής του Πανεπιστημίου Ιωαννίνων, για την πολύτιμη καθοδήγησή του και τη σημαντικότατη βοήθειά του στην ολοκλήρωση της εργασίας αυτής.

Ακόμη θα ήθελα να ευχαριστήσω τα μέλη του εργαστηρίου και συγκεκριμένα τους Ευτυχία Μπαϊκούση, Παναγιώτη Δομουχτσίδη, Γιώργο Ρογκάκο και Αλίκη Πιλαλίδου για την βοήθειά τους και τις σημαντικές παρατηρήσεις τους κατά τη διάρκεια εκπόνησης της εργασίας μου. Τέλος θα ήθελα να ευχαριστήσω τη Νατάσα Καλύβα και τη Σεβαστή Ισπόγλου για τη συμπαράσταση των τριών αυτών χρόνων φοίτησής μου στο ΠΜΣ Πληροφορικής και κυρίως στη βοήθεια που μου πρόσφεραν τις τελευταίες μέρες ολοκλήρωσης της εργασίας αυτής.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

ΣΧΗΜΑ ΣΕΛ

ΠΕΡΙΛΗΨΗ

Χαρά Παπαγεωργίου του Ευαγγέλου και της Ελένης. MSc Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούλιος 2009. Τίτλος Διατριβής: Αποδοτική Αποτίμηση Ερωτήσεων OLAP. Επιβλέποντας: Παναγιώτης Βασιλειάδης.

Η Σύγχρονη Αναλυτική Επεξεργασία Δεδομένων (On-Line Analytical Processing - OLAP) είναι μια τεχνολογία επερώτησης βάσεων δεδομένων, που στηρίζεται στη θεώρηση της πληροφορίας με πολυδιάστατο τρόπο στο επίπεδο των πελατών. Οι συνηθισμένες πράξεις OLAP όπως επιλογή, συναθροιστική άνοδος και αναλυτική κάθοδος σε επίπεδα αδρομέρειας, όμως, είναι χρονοβόρες. Έτσι, απαιτούνται εξειδικευμένες τεχνικές επεξεργασίας των ερωτήσεων OLAP ώστε να εξοικονομείται χρόνος επεξεργασίας.

Στην εργασία αυτή, χρησιμοποιείται μια μοντελοποίηση της πολυδιάστατης πληροφορίας από τη βιβλιογραφία σαν βάση για την επεξεργασία λειτουργιών στους κύβους και παρουσιάζονται συντακτικοί χαρακτηρισμοί για τα προβλήματα της καταλληλότητας κύβων (ήτοι, του προβλήματος χρησιμοποιήσεως δεδομένων από κάποιον κύβο για να υπολογιστεί ένας άλλος κύβος) και της προκύπτουσας επανεγγραφής ερωτήσεων. Τα πειραματικά αποτελέσματα υποδεικνύουν ότι επιτυγχάνεται σημαντική επιτάχυνση με την εν λόγω επανεγγραφή των ερωτήσεων. Επιπλέον, παρουσιάζονται αποτελέσματα σε σχέση με την εσωτερική οργάνωση των επί μέρους αποτελεσμάτων, με σκοπό την μετέπειτα αποδοτική επαναχρησιμοποίησή τους. Τα πειραματικά αποτελέσματα υποδεικνύουν ότι είναι δυνατόν, με απλές μεθόδους εσωτερικής αναπαράστασης να επιτευχθεί υψηλή επίδοση στην ποιότητα των επιλεγομένων πλάνων επεξεργασίας μιας ερώτησης.

EXTENDED ABSTRACT IN ENGLISH

Papageorgiou Hara E. MSc Computer Science Department, University of Ioannina, Greece. July2009. Efficient Evaluation of OLAP queries. Thesis Supervisor: Panos Vassiliadis

On-Line Analytical Processing (OLAP) is a trend in database technology based on the multidimensional view of data. This thesis builds upon previous work that operates on the basis of a logical model for cubes based on the key observation that a cube is not a self-existing entity, but rather a view over an underlying data set. The model is powerful enough to capture all the commonly encountered OLAP operations such as selection, roll-up and drill-down, through a sound and complete algebra.

The first contribution of this thesis is the practical assessment of theoretical results on the processing of cube operations. Existing theoretical results provide syntactic characterizations for the problem of cube usability (i.e., the problem of using the tuples of a cube to compute another cube) for different classes of cubes; yet no practical assessment on the feasibility of the practical application of these results exists. In this thesis, we evaluate processing times for the decision of the cube usability problem as well as for the answering of a cube from another cube (as opposed to answering the cube from the detailed facts).

The second contribution of this thesis has to do with the problem of selecting the appropriate cube to answer a new cube request in the presence of several previous intermediate results. We exploit the fact that users perform sessions of cube operations and keep the history of previous operations as well as their results. We propose several algorithms that exploit the size of the results, the sequence of operations and the timestamp for the most recently created or used previous results and experimentally assess them with different thresholds for the number of cubes that the selection algorithm visits.

The results of this thesis show that (a) it is practically feasible and efficient to sustain the overhead of deciding the usability of a new cube from other cubes, in order to achieve better execution performance and (b) that simple in-memory organizations of the cubes' metadata are powerful enough to allow an approximate decision on the choice of the best possible cube with high degree of solution quality and overall efficiency.

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

1.1 OLAP και το πρόβλημα καταλληλότητας των κύβων

Η Σύγχρονη Αναλυτική Επεξεργασία Δεδομένων (On-Line Analytical Processing - OLAP) είναι μια κατηγορία λογισμικού που επιτρέπει σε αναλυτές και διοικητικά στελέχη να αποκτήσουν γνώση των δεδομένων μέσω μιας γρήγορης, συνεπούς και αξιόπιστης πρόσβασης σε μια μεγάλη ποικιλία όψεων της πληροφορίας που έχει μετασχηματιστεί από απλά δεδομένα, ώστε να αναπαριστά τη πολυδιάστατη θεώρηση ενός οργανισμού, όπως γίνεται αντιληπτή από το χρήστη. Τα εργαλεία OLAP εστιάζουν στην παροχή πολυδιάστατης ανάλυσης της πληροφορίας. Τα δεδομένα οργανώνονται σε *κύβους* ή αλλιώς *υπερκύβους* (*cubes* και *hypercubes* αντίστοιχα), που ορίζονται σε πολυδιάστατους χώρους, αποτελούμενους από πολλές διαστάσεις. Κάθε διάσταση αποτελείται από πολλά επίπεδα συνάθροισης. Τυπικές λειτουργίες OLAP περιλαμβάνουν την συνάθροιση ή ανάλυση της πληροφορίας, την επιλογή (*selection*) τμημάτων της και περιστροφής της παρουσίασης της, με βάση τις διαστάσεις της, στην οθόνη .

Υπάρχουν δύο πόλοι γύρω από τους οποίους συγκεντρώνονται τα εργαλεία OLAP και οι οποίοι έχουν να κάνουν με τη φυσική αποθήκευση των δεδομένων. Από τη μία πλευρά υπάρχει η *αρχιτεκτονική ROLAP* (Relational On-Line Analytical Processing) , και από την άλλη υπάρχει η *αρχιτεκτονική MOLAP* (Multidimensional On-Line Analytical Processing). Το πλεονέκτημα της αρχιτεκτονικής MOLAP είναι ότι παρέχει μια άμεση πολυδιάστατη όψη των δεδομένων, ενώ το πλεονέκτημα της αρχιτεκτονικής ROLAP είναι απλά μια πολυδιάστατη διαπροσωπεία σε σχεσιακά

συστήματα βάσεων δεδομένων. Η αρχιτεκτονική ROLAP έχει δύο πλεονεκτήματα: (α) μπορεί να ενσωματωθεί εύκολα σε υπάρχοντα σχεσιακά συστήματα, και (β) τα σχεσιακά δεδομένα μπορούν να αποθηκευθούν πιο αποδοτικά από τα πολυδιάστατα δεδομένα.

Σε μια αρχιτεκτονική ROLAP, τα δεδομένα οργανώνονται σε *σχήμα αστέρα* ή *νιφάδας* (*star* ή *snowflake* schema). Ένα σχήμα αστέρα αποτελείται από ένα κεντρικό *πίνακα πληροφοριών* (*fact* table) και διάφορους αποκανονικοποιημένους *πίνακες διάστασης* (*dimension* tables). Τα *μέτρα* (*measures*) της πληροφορίας αποθηκεύονται στον πίνακα πληροφοριών. Για κάθε διάσταση του πολυδιάστατου μοντέλου, υπάρχει και ένας πίνακας διάστασης με όλα τα επίπεδα συνάθροισης και τις επιπλέον ιδιότητες των επιπέδων αυτών. Η κανονικοποιημένη εκδοχή του σχήματος αστέρα είναι το σχήμα νιφάδας όπου κάθε επίπεδο μιας διάστασης έχει το δικό του πίνακα.

Το πρόβλημα, που προκύπτει από το μεγάλο μέγεθος των πινάκων και κυρίων του πίνακα πληροφοριών, είναι ο μεγάλος χρόνος εκτέλεσης κάποια ερώτησης. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί με τη χρήση υλοποιημένων κύβων που είναι παρόμοιο με το πρόβλημα υπαγωγής όψεων το οποίο έχει μελετηθεί αρκετά. Αντί το ερώτημα να τίθεται στους πίνακες της βάσης, μπορεί να τεθεί στους υλοποιημένους κύβους που σίγουρα θα είναι μικρότεροι από τους πίνακες κι έτσι θα έχουμε αποδοτικότερη εκτέλεση του ερωτήματος.

Τα ζητήματα τα οποία καλείται να λύσει κάποιος για να χρησιμοποιήσει τους υλοποιημένους κύβους είναι α) αν ο κύβος μπορεί να χρησιμοποιηθεί για την απάντηση ενός νέου ερωτήματος β) η επαναδιατύπωση ερωτήματος χρησιμοποιώντας τους υλοποιημένους κύβους αντί των άλλων πινάκων. Ένα άλλο ζήτημα το οποίο προκύπτει δοθέντων κάποιων υλοποιημένων κύβων που μπορούν να απαντήσουν ένα ερώτημα, είναι η επιλογή του κύβου που θα επιφέρει την αποδοτικότερη εκτέλεση του ερωτήματος. Το πρόβλημα αυτό αναλύεται στην εργασία αυτή και προτείνεται ένας αλγόριθμος με διάφορες παραλλαγές για την επιλογή του βέλτιστου κύβου στο σύνολο των υποψηφίων κύβων.

1.2 Δομή της Διατριβής

Η διατριβή περιέχει έξι κεφάλαια: Στο Κεφάλαιο 1 παρουσιάζονται βασικές έννοιες OLAP και ο ορισμός του προβλήματος. Στο Κεφάλαιο 2 παρουσιάζεται σχετική

εργασία με το πρόβλημα υπαγωγής και επαναδιατύπωσης ερωτημάτων. Στο Κεφάλαιο3 δίνεται το θεωρητικό υπόβαθρο και ο ορισμός του προβλήματος . Στο Κεφάλαιο 4 παρουσιάζεται η πειραματική μελέτη που έγινε εκτελώντας τον αλγόριθμο καταλληλότητας κύβου και την επαναδιατύπωση ερωτήματος που προτείνεται στο κεφάλαιο 3. Στο Κεφάλαιο 5 προτείνεται ο αλγόριθμος επιλογής κύβου και παρουσιάζονται και τα αποτελέσματα της πειραματικής μελέτης του. Στο Κεφάλαιο 6 περιλαμβάνει τα συμπεράσματα και τη συνεισφορά της εργασίας και μελλοντική δουλειά.

.

ΚΕΦΑΛΑΙΟ 2. ΣΧΕΤΙΚΗ ΕΡΓΑΣΙΑ

2.1 Εισαγωγή

Η διαχείριση συναθροιστικών δεδομένων είναι ένα θεμελιώδες ζήτημα σε πολλές εφαρμογές όπως σε αποθήκες δεδομένων και συστήματα OLAP. Στα συστήματα αυτά, τα ερωτήματα αφορούν συναθροίσεις σε δεδομένα μεγάλου όγκου. Η χρήση υλοποιημένων συναθροιστικών όψεων μπορούν να αυξήσουν σημαντικά την απόδοση της επεξεργασίας ενός ερωτήματος. Πολλές φορές τα δεδομένα των υλοποιημένων όψεων αντικαθιστούν τα αρχικά δεδομένα χωρίς να μπορούμε να έχουμε πρόσβαση σε αυτά. Αυτό γίνεται συνήθως για λόγους εμπιστευτικότητας είτε για δεδομένα που αφορούν κινητή υπολογιστική που χρειάζονται τοπικά διαθέσιμα δεδομένα, ή για στατιστικές βάσεις δεδομένων.

Εφόσον οι υλοποιημένες όψεις μπορούν να προσφέρουν βελτίωση στην απόδοση απάντησης ερωτημάτων, αποτελούν ένα σημαντικό συστατικό στη σχεδίαση μιας αποθήκης δεδομένων. Αν ο χρόνος εκτέλεσης ενός ερωτήματος, ήταν το μοναδικό θέμα που έπρεπε να λάβουμε υπόψη, ένα σκεπτικό θα ήταν να υλοποιήσουμε όλες τις πιθανές συναθροιστικές όψεις. Αυτό όμως το σενάριο δεν είναι εφικτό, εφόσον ο αριθμός των πιθανών συναθροιστικών όψεων είναι εκθετικός ως προς τον αριθμό των διαστάσεων που εξετάζεται το σύνολο δεδομένων. Για να υλοποιηθούν αυτές οι όψεις, θα απαιτούνταν τεράστιο υπολογιστικό κόστος και τεράστιος χώρος για αποθήκευση των όψεων αυτών, καθώς επίσης θα έπρεπε να ληφθεί υπόψη επίσης το γεγονός ενημέρωσης των όψεων, που αυτό στους περισσότερους οργανισμούς γίνεται

συνήθως κάθε μέρα. Συνολικά τα θέματα που εξετάζονται στον τομέα των υλοποιημένων όψεων είναι τα εξής:

- Ευχρηστία όψεων
- Επαναδιατύπωση ερωτήματος
- Επιλογή όψης για απάντηση ερωτήματος
- Επιλογή όψης για αποθήκευση
- Ενημέρωση όψεων

Στην εργασία αυτή μας απασχολούν τα προβλήματα ευχρηστίας κάποιας σχεσιακής όψης και της επαναδιατύπωσης ενός ερωτήματος χρησιμοποιώντας την όψη αντί μία σχεσιακή σχέση. Στο κεφάλαιο αυτό παρουσιάζονται συνθήκες που πρέπει να ισχύουν για να μπορεί μία υλοποιημένη όψη να απαντήσει μία δεδομένη ερώτηση, καθώς επίσης και κάποιοι αλγόριθμοι επαναδιατύπωσης ερωτήματος χρησιμοποιώντας όψεις αντί του αρχικού σχεσιακού σχήματος.

2.2 Δυνατότητα χρήσης όψεων

Στο [GT03] παρουσιάστηκε πότε ένα ερώτημα μπορεί να απαντηθεί από ένα άλλο, διατυπώνοντας κάποιον θεωρητικό κανόνα και στη συνέχεια παρουσίασαν έναν αλγόριθμο επαναδιατύπωσης ενός ερωτήματος, χρησιμοποιώντας ένα σύνολο όψεων αντί για τους αρχικούς σχεσιακούς πίνακες.

Ορισμός 1: Έστω `Q` και `Q′` δύο σύνολα από ερωτήματα σε ένα σχήμα s. Θεωρούμε ότι το `Q` υπάγει το `Q′` γράφοντας `Q|=Q′`, αν και μόνον αν για κάθε ζεύγος `I1` και `I2` στιγμιοτύπων του σχήματος s, η ισότητα `Q(I1)=Q(I2)` συνεπάγεται ότι `Q′(I1)=Q′(I2)`

Το οποίο σημαίνει ότι αν κάνουμε μία ερώτηση σε δύο διαφορετικά στιγμιότυπα και πάρουμε το ίδιο σύνολο αποτελεσμάτων και στα δύο, τότε αν κάνουμε την νέα ερώτηση στα δυό ίδια στιγμιότυπα το σύνολο των αποτελεσμάτων πρέπει επίσης να είναι το ίδιο μεταξύ τους για να μπορεί το αρχικό ερώτημα να μπορέσει να υπάγει το νέο. Κατ' επέκταση προκύπτει ο εξής ορισμός:

Ορισμός 2: Δοθέντος ενός συνόλου από όψεις `V` σε ένα σχήμα `s` και ενός στιγμιοτύπου `I` στο σχήμα , ορίζουμε ως I^V την επέκταση του `I` στις όψεις του `V`. Δύο ερωτήματα `q` και `q′` είναι `equivalent modulo V`, αν για κάθε στιγμιότυπο `I` στο σχήμα

s, $q(I^V)=q'(I^V)$. Αυτό συνεπάγεται ότι το q' είναι μία ισοδύναμη επαναδιατύπωση του q.

Στο [SDJL96] παρουσιάζονται οι συνθήκες που πρέπει να ισχύουν για να μπορεί μία όψη να απαντήσει ένα ερώτημα. Για να μπορέσουν να μελετηθούν οι συνθήκες, αναφέρουμε κάποια στοιχεία της σημειογραφίας που είναι απαραίτητα.

ColSel(Q): Οι στήλες προβολής που δεν είναι στη συνθήκη GROUP BY

Groups(Q): Οι στήλες ομαδοποίησης

Cols(V): Οι στήλες της όψης

Sel(V): Όλες οι στήλες που βρίσκονται στο SELECT της όψης

Conds(Q): Οι συνθήκες επιλογής στο WHERE του ερωτήματος

AGG(A): Η συναθροιστική συνάρτηση για την στήλη A. Η AGG μπορεί να είναι είτε MIN, MAX, SUM, COUNT .

Για να μπορεί μία όψη V να απαντήσει ένα ερώτημα Q, πρέπει:

- Η όψη V να συμπεριλαμβάνει όλες τις απαραίτητες κολώνες για τον υπολογισμό του Q. Διαισθητικά μία στήλη A χρειάζεται στο ερώτημα Q αν εμφανίζεται στο αποτέλεσμα του Q, ή αν το Q χρειάζεται να υπολογίσει μία συνθήκη στην οποία συμμετέχει το A και δεν έχει υπολογιστεί η συνθήκη αυτή στην όψη V.
- Η όψη V να περιέχει όλες τις πλειάδες που χρειάζονται στο Q. Διαισθητικά αυτό σημαίνει ότι μία πλειάδα είναι απαραίτητη για το ερώτημα Q, αν ικανοποιεί τις συνθήκες που υπολογίζονται στο Q.

Παρακάτω παρουσιάζονται οι συνθήκες αυτές με πιο λεπτομερή περιγραφή.

Συνθήκη 1: Αν υπάρχει μία ένα προς ένα απεικόνιση φ από την V στο Q.

Συνθήκη 2: Αν μία στήλη A στο ColSel(Q) □Groups(Q) είναι μία κολώνα στην φ(Cols(V)), τότε η Sel(V) πρέπει να έχει μία στήλη B_A τέτοια ώστε στις Conds(Q) να υπονοείται ότι $(A=B_A)$

Συνθήκη 3: Έστω ότι AGG(A) είναι μία στήλη στο Sel(Q). Αν η στήλη A ανήκει στην φ(Cols(V)) τότε

1. Αν η AGG είναι η MIN, MAX, SUM τότε η Sel(V) πρέπει να έχει μία στήλη B_A τέτοια ώστε στις Conds(Q) να εννοείται ότι $(A=\varphi(B_A))$

2. Αν η AGG είναι η COUNT τότε η Sel(V) δεν πρέπει να είναι άδεια.

Συνθήκη 4: Πρέπει να υπάρχει ένας συνδυασμός κατηγορημάτων τέτοιος ώστε:

1. Conds(Q) είναι ισοδύναμο του φ(Conds(V)) & Conds′.

2. Conds′ περιλαμβάνει μόνο τις στήλες στο φ(Sel(V)) Y (Cols(Q) – φ(Cols(V)))

Στο survey του Halevy [Ha01] δίνονται οι συνθήκες για τη δυνατότητα χρήσης μίας όψης για τον υπολογισμό ενός ερωτήματος. Οι συνθήκες αυτές αφορούν την περίπτωση που το ερώτημα δεν περιέχει συναθροίσεις κι ερωτήσεις

- Πρέπει να υπάρχει μία απεικόνιση ψ των πινάκων που αναφέρονται στο From της όψης V στους πίνακες που αναφέρονται στο from του ερωτήματος Q.
- Η όψη V πρέπει να εφαρμόζει τις συνθήκες επιλογής και σύζευξης των ατόμων του Q στην απεικόνιση ψ ή πρέπει να εφαρμόζει πιο χαλαρή επιλογή και να προβάλει όλα τα πεδία που είναι απαραίτητα για να υπολογιστούν τα άτομα του Q
- Η όψη V πρέπει να συμπεριλαμβάνει τα πεδία που είναι απαραίτητα για την επιλογή του Q εκτός αν αυτά τα πεδία μπορούν να ανακτηθούν από άλλη όψη.

Αυτό που αναφέρει για τα συναθροιστικά ερωτήματα είναι αυτά που παρουσιάσαμε παραπάνω στο [DJLS03].

2.3 Επαναδιατύπωση ερωτήματος

Με τα παραδείγματα που ακολουθούν εξηγούμε τη σημειογραφία που είναι απαραίτητη για τη μελέτη του αλγορίθμου επαναδιατύπωσης ερωτήματος που πρότειναν στο [GT03]

Παράδειγμα 2.1

Έστω η σχέση R(i,c,x,s,y,p,z) με σχήμα (Call-id , Cust, Day, Source, Dest, Plan, Dur). Θεωρούμε τις παρακάτω συναρτήσεις

count C^{icsypz} R(icxsypz) που δηλώνει για κάθε μέρα x πόσα τηλέφωνα έγιναν

sum $\sum_{z}^{ixsyp}$ R(icxsypz) δηλώνει πόσες ώρες μίλησε ο κάθε πελάτης

$\text{Perc}_{x,p}\ R(icxsypz) = \frac{C^{icsyz}R(icxsypz)}{C^{icsypz}R(icxsypz)}$ δηλώνει για κάθε ζευγάρι μέρα/πλάνο το ποσοστό των τηλεφώνων κάθε πλάνου που έγιναν εκείνη τη μέρα ως προς το σύνολο όλων των τηλεφώνων που έγιναν εκείνη τη μέρα.

$Avg_z^{ixsyp}R(icxsypz) = \frac{C_z^{ixsyp}R(icxsypz)}{C^{ixsypz}R(icxsypz)}$ δηλώνει για κάθε πελάτη το μέσο όρο $\frac{C(Day, Source, Plan)}{C(Day, Source)}$ διάρκειας ομιλίας του

Παράδειγμα 2.2:

Έστω το παρακάτω ερώτημα:

```
Query Q1:
select Day, Source, Plan, N/D
from ((select Day, Source, Plan, count(*) as N
       from Intercalls
       group by Day, Source, Plan )
       natural join
       (select Day, Source, count(*) as D
       from Intercalls
       group by Day, Source))
```

Το ερώτημα αυτό σύμφωνα με τη σημειογραφία που χρησιμοποιείται συμβολίζεται ως $\frac{C(Day, Source, Plan)}{C(Day, Source)}$

Ακολουθεί ο αλγόριθμος επαναδιατύπωσης ερωτήματος

Αλγόριθμος AggRew1

Είσοδος: ένα ερώτημα F(X) για να επαναδιατύπωση (F= C, S_Y, M_Y) και ένα σύνολο από όψεις fi(Wi) (fi=C, S_{Ui}, M_{Ui})

Έξοδος: ένα επαναδιατυπωμένο ερώτημα F'(X) χρησιμοποιώντας μόνο τις όψεις ή ένα μήνυμα λάθους

```
switch(F)
case C:
       for each count view C(Wi)
              if X⊆Wi
              then return F'(X)= Σ^{Wi−X}(C(Wi))(X)
case S_Y:
```

for each count view C(Wi)

if (X∪Y) ⊆ Wi

then return F′(X)=$\overline{\Sigma^{Wi-X} Y * (C(Wi))}(X)$

for each sum view S_{Ui}(Wi)

if Y=Ui and X⊆Wi

then return F′(X)=$\overline{\Sigma^{Wi-X} (S_{Ui})(Wi))}(X)$

case M_Y:

for each count view C(Wi)

if (X∪Y) ⊆ Wi

then return F′(X)=$\overline{\Pi^{Wi-X} (Y^{C(Wi)})}(X)$

for each multiply view M_{Ui}(Wi)

if Y=Ui and X⊆Wi

then return F′(X)=$\overline{\Pi^{Wi-X} (M_{Ui}(Wi))}(X)$

endswitch

return a failure message

Στη συνέχεια παρουσιάζεται ο αλγόριθμος υπολογισμού επαναδιατύπωσης του ερωτήματος με χρήση όψεων που παρουσιάστηκε στο [DJLS96]. Ο αλγόριθμος αυτός ισχύει για τη δημιουργία ερωτήματος που δεν περιέχει τον τελεστή UNION και το ερώτημα δεν έχει την πρόταση HAVING.

Αλγόριθμος ConjViewSingleBlock

Βήμα 1: Αντικατέστησε όλους τουε πίνακες στο φ(Tables(V)) με φ(V)

Βήμα 2: Αντικατέστηκε κάθε στήλη A στο Groups(Q)∪ColSel(Q) ∪AggSel(Q) με $\varphi(B_A)$

Βήμα 3: Βρες ένα λογικό συνδυασμό από κατηγορήματα Conds′ που να ικανοποιούν τη συνθήκη 4. Αντικατέστησε τα Conds(Q) στο Q με τα Conds′

Βήμα 4: Βρες μία στήλη συνάθροισης COUNT(A) στο Sel(Q) τέτοια ώστε να υπάρχει στο φ(Cols(V)) και όχι στο φ(Sel(V)). Αντικατέστησε COUNT(A) με COUNT(B) όπου B είναι στήλη στο φ(V).

Στο survey του Halevy προτείνεται ο Bucket αλγόριθμος για επαναδιατύπωση ερωτήματος όμως δεν αφορά συναθροιστικές ερωτήσεις. Σε γενικές γραμμές αυτός ο αλγόριθμος δημιουργεί κάδους για κάθε σχεσιακό άτομο του ερωτήματος. Στη συνέχεια ελέγχει κάθε όψη και εισάγει στον κάδο κάθε ατόμου την όψη αρκεί να πληροί κάποιες προϋποθέσεις. Συγκεκριμένα οι όψεις πρέπει να αναφέρονται στον

ίδιο πίνακα που αναφέρεται το άτομο του ερωτήματος, τα άτομα πρέπει να αφορούν το ίδιο σύνολο τιμών και οι μεταβλητές που βρίσκονται στην κεφαλή του ερωτήματος πρέπει να υπάρχουν και στην κεφαλή της όψης. Στη συνέχεια μπορούν να προκύψουν διάφοροι συνδυασμοί βάζοντας ένα άτομο από κάθε κάδο.

Στο [CNS98] προτείνονται δύο αλγόριθμοι επαναδιατύπωσης ερωτήματος για τις περιπτώσεις που η συνάρτηση συνάθροισης είναι η `count` ή η `sum`. Η ιδέα στην οποία στηρίζονται οι αλγόριθμοι αυτοί είναι στο αυξητικά να καλύπτονται τα άτομα του ερωτήματος από όψεις, απ' την άποψη να ικανοποιούν οι όψεις τα άτομα αυτά. Οι όψεις επιλέγονται τυχαία και σε περίπτωση που δεν μπορούν να καλύψουν το ερώτημα, ο αλγόριθμος επιλέγει διαφορετική. Αν δεν βρεθεί κάποια όψη τότε ο αλγόριθμος αποτυγχάνει και δεν υπολογίζεται επαναδιατύπωση ερωτήματος.

Στο [CNS98] αναφέρεται ως συνθήκη ισοδυναμίας δύο ερωτήσεων η συνθήκη οι δύο ερωτήσεις να είναι `equivalent modulo V`. Στο [Co05] αναφέρεται ότι δύο ερωτήματα για να είναι ισοδύναμα στην περίπτωση που είναι `quasilinear`, αρκεί να είναι ισομορφικά, δηλαδή να υπάρχει μία απεικόνιση από το ένα ερώτημα στο άλλο και αντίστροφα. `Quasilinear` είναι τα ερωτήματα που κανένα θετικό κατηγόρημα δεν υπάρχει παραπάνω από μία φορά. Στην κλάση των `quasilinear` ερωτημάτων ανήκουν οι συναρτήσεις `max, top2, count, sum, prod, parity` και `avg`.

2.4 Συνολική εικόνα σχετικής εργασίας

Στον παρακάτω πίνακα παρουσιάζονται λεπτομερώς, τι περιλαμβάνουν οι προτάσεις που δόθηκαν από δημοσιεύσεις που παρουσιάζονται σύντομα στις επόμενες παραγράφους.

Πίνακας 2.1 Συνολική εκτίμηση

	Υπαγωγή ερώτησης	**Επαναδιατύπωση ερώτησης**	**Συνθήκη Επιλογής**	**Συναρτήσεις**	**OLAP**
GT03	√	√	ερωτήσεις χωρίς where clause	sum, count, mult, perc, avg	-
SDJL96	√	√	Συζευκτικές ερωτήσεις	sum, count, min, max	-
CNS98	√	√	Συζευκτικές ερωτήσεις	count, sum	-
Co05	√	-	Συζευκτικές ερωτήσεις	for all funcs	-

Παρατηρούμε ότι σε όλες τις εργασίες έχουν δοθεί ορισμοί και σε μερικές και αλγόριθμοι για υπαγωγή και επαναδιατύπωση ερωτήσεων. Ωστόσο κανένας από τους αλγορίθμους και από τους θεωρητικούς ορισμούς δεν ισχύουν για ιεραρχίες OLAP δηλαδή για σχεσιακά σχήματα με συναρτησιακές εξαρτήσεις.

ΚΕΦΑΛΑΙΟ 3. ΚΥΒΟΙ ΓΙΑ ΠΟΛΥΔΙΑΣΤΑΤΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

3.1 Βασικές Έννοιες OLAP

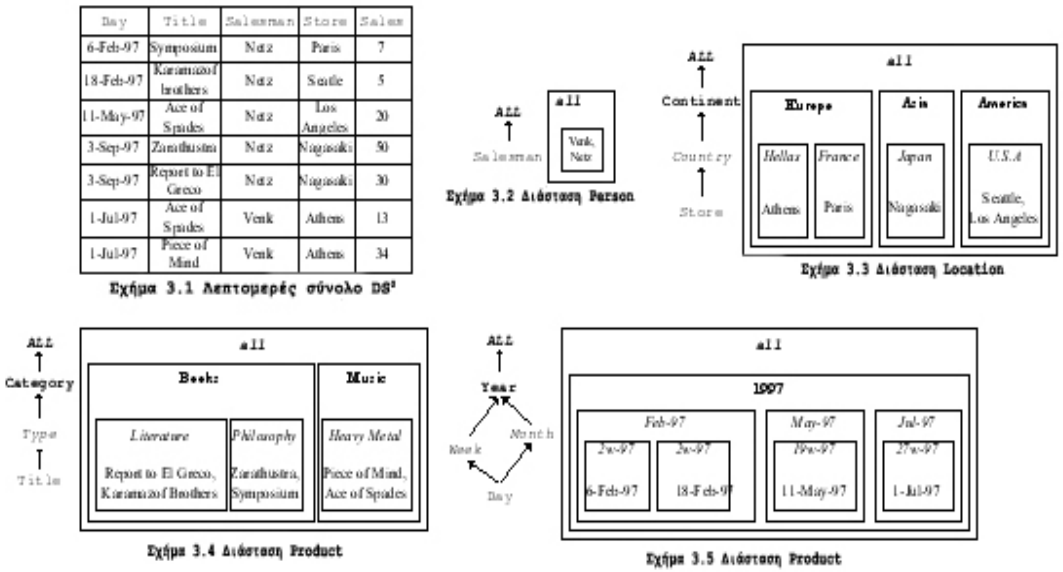
Το θεωρητικό υπόβαθρο της εργασίας αυτή, το οποίο παρουσιάζεται στο παρόν κεφάλαιο, έχει παρουσιασθεί στην εργασία [VaSk00]. Συγκεκριμένα παρουσιάζονται οι βασικές οντότητες και λειτουργίες του προτεινόμενου μοντέλου. Οι οντότητες αυτές συμπεριλαμβάνουν διαστάσεις, σύνολα δεδομένων και κύβους. Οι λειτουργίες περιλαμβάνουν επιλογές και αλλαγή στη λεπτομέρεια των δεδομένων. Το μοντέλο επεκτείνει προηγούμενες προσεγγίσεις και συγκεκριμένα τις [Vass98, CaTo97, Lehn98].

Ένα από τα κύρια χαρακτηριστικά των OLAP εφαρμογών είναι η πολυδιάστατη θεώρηση των δεδομένων (*multidimensional view of data*) σε ότι αφορά τον τρόπο με τον οποίο τα αντιμετωπίζει ο χρήστης. Εν γένει, σε ότι αφορά το χρήστη, τα δεδομένα θεωρούνται αποθηκευμένα σε ένα *πολυδιάστατο πίνακα* (*multi-dimensional array*), ο οποίος αποκαλείται και *κύβος* ή *υπερκύβος* (*Cube* και *HyperCube* αντίστοιχα). Ο κύβος είναι μια ομάδα από *κελιά δεδομένων* (*data cells*). Κάθε κελί χαρακτηρίζεται μονοσήμαντα από τις αντίστοιχες τιμές των διαστάσεων του κύβου. Τα περιεχόμενα του κελιού ονομάζονται *μέτρα* (*measures*) και αναπαριστούν τις αποτιμώμενες αξίες

του πραγματικού κόσμου. Τα μέτρα είναι συναρτησιακά εξαρτημένα, με τη σχεσιακή έννοια, από τις διαστάσεις του κύβου.

Μια *διάσταση* (*dimension*) ορίζεται ως "ένα δομικό χαρακτηριστικό ενός κύβου, που αποτελείται από μια λίστα τιμών, οι οποίες είναι όλες του ιδίου τύπου σε ότι αφορά την αντίληψη των δεδομένων από το χρήστη" [OLAP97]. Με άλλα λόγια, μια διάσταση μοντελοποιεί όλους τους τρόπους με τους οποίους τα δεδομένα μπορούν να συναθροιστούν σε σχέση με μια συγκεκριμένη παράμετρο του περιεχομένου τους. Κάθε διάσταση έχει μια σχετική *ιεραρχία επιπέδων* συνάθροισης των δεδομένων (*hierarchy of levels*). Αυτό σημαίνει, με απλά λόγια, ότι η διάσταση μπορεί να θεωρηθεί από πολλά επίπεδα λεπτομέρειας. Τυπικά, μια *διάσταση* D είναι ένα δίκτυο (lattice) $(\mathbf{L}, \prec)$: $\mathbf{L}=(L_1, \ldots, L_n, \mathrm{ALL})$. Απαιτείται το ανώτερο επίπεδο του δικτύου να είναι πάντα το επίπεδο ALL, έτσι ώστε να μπορούμε να ομαδοποιήσουμε όλες τις τιμές της διάστασης σε μία τιμή 'all'. Το κάτω όριο του δικτύου ονομάζεται *λεπτομερές επίπεδο* (*detailed level*) της διάστασης. Για παράδειγμα, ας θεωρήσουμε τη διάσταση Date της εικόνας 3.1. Τα επίπεδα της διάστασης Date είναι Day, Week, Month, Year και ALL. Το επίπεδο Day είναι το πλέον λεπτομερές επίπεδο. Το επίπεδο ALL είναι το πλέον υψηλό επίπεδο συνάθροισης για όλες τις διαστάσεις. Το να συναθροίζουμε την πληροφορία στο επίπεδο ALL μιας διάστασης σημαίνει ότι πρακτικά αγνοούμε τη διάσταση κατά τη συνάθροιση (συναθροίζουμε, δηλαδή, τα δεδομένα με βάση όλες τις άλλες διαστάσεις, πλην αυτής).

Η σχέση μεταξύ των τιμών των επιπέδων επιτυγχάνεται με τη χρήση ενός συνόλου συναρτήσεων της μορφής anc_{L1}^{L2}. Μια συνάρτηση anc_{L1}^{L2} αντιστοιχίζει μια τιμή του επιπέδου L_2 στο επίπεδο L_1. Για παράδειγμα, $\mathrm{anc}_{\mathrm{Month}}^{\mathrm{Year}}(\mathrm{Feb\text{-}97})=1997$. Θα αποκαλούμε τις συναρτήσεις αυτές και *συναρτήσεις προγόνου* (ancestor functions).



Σχήμα 3.1 Δεδομένα και Ιεραρχίες παραδείγματος

Οι κυριότερες λειτουργίες πολυδιάστατων βάσεων είναι η *επιλογή* (*selection*) και η *πλοήγηση* (*navigation*). Η επιλογή χρησιμοποιείται οπουδήποτε ένα κριτήριο εφαρμόζεται στα δεδομένα με βάση τις τιμές μιας διάστασης, με σκοπό να περιοριστεί το σύνολο των παρουσιαζομένων δεδομένων. Η πλοήγηση είναι ένας όρος για να περιγραφούν οι διαδικασίες που χρησιμοποιούν οι χρήστες για να εξερευνούν την πληροφορία ενός κύβου με αλληλεπιδραστικό τρόπο, αλλάζοντας το επίπεδο λεπτομέρειας της πληροφορίας, όπως αυτή τους παρουσιάζεται [JLVV00, OLAP97]. Πιθανές πράξεις πλοήγησης είναι: (α) *Συναθροιστική Άνοδος* (*Roll-up*) που αντιστοιχεί στη συνάθροιση των δεδομένων από χαμηλότερο σε υψηλότερο επίπεδο λεπτομέρειας στην ιεραρχία μιας διάστασης, (β) *Αναλυτική Κάθοδος* (*Drill-Down*) που είναι η αντίστροφη λειτουργία της συναθροιστικής ανόδου και επιτρέπει την ανάλυση της πληροφορίας από υψηλότερο σε χαμηλότερο επίπεδο λεπτομέρειας, και (γ) *Τεμαχισμός* (*Slicing)* που αντιστοιχεί στην συνάθροιση των δεδομένων σε σχέση με ένα υποσύνολο μόνο των διαστάσεων του κύβου. Για παράδειγμα, ας θεωρήσουμε τη διάσταση `Date`: συναθροίζοντας από το επίπεδο `Month` στο επίπεδο `Year` είναι μια πράξη συναθροιστικής ανόδου και αναλύοντας από το επίπεδο `Month` στο επίπεδο `Day` είναι μια πράξη αναλυτικής καθόδου. Στο μοντέλο που παρουσιάζεται, ο τεμαχισμός είναι μια πράξη συναθροιστικής ανόδου στο επίπεδο `ALL`.

Στο προτεινόμενο μοντέλο, ορίζουμε την έννοια *σύνολο δεδομένων* (*data set*) σαν ένα σύνολο πλειάδων κάτω από ένα συγκεκριμένο σχήμα. Επιπλέον, θεωρείται η ύπαρξη ενός *λεπτομερούς συνόλου δεδομένων* (*detailed data set*), ενός συνόλου δεδομένων, δηλαδή, που ορίζεται στα πλέον λεπτομερή επίπεδα όλων των διαστάσεων του σχήματός του. Αυτό το λεπτομερές σύνολο δεδομένων είναι η κεντρική πηγή πληροφορίας, η οποία θα παρέχει δεδομένα σε όλους τους κύβους που θα παραχθούν κατά τη διάρκεια μιας OLAP συνόδου (π.χ., μπορεί να είναι ο πίνακας πληροφοριών σε μια συγκεντρωτική αποθήκη δεδομένων).

Κάτι το οποίο πρέπει να επισημανθεί είναι ότι ο κύβος δεν είναι μια ανεξάρτητη οντότητα (όπως συνήθως παρουσιάζεται στη βιβλιογραφία) αλλά μια όψη πάνω στο λεπτομερές σύνολο δεδομένων. Ως συνήθως, μια όψη (και άρα, ένας κύβος) μπορεί είτε να είναι αποθηκευμένη ή όχι. Κατά συνέπεια, ο κύβος μπορεί να αντιμετωπιστεί με δύο τρόπους: απλά ως σύνολο δεδομένων, ή σαν μια επερώτηση. Στο μοντέλο που παρουσιάζεται παραμένει αυτός ο δυϊσμός τυπικά: ένας κύβος δεν είναι απλά ένα σύνολο πλειάδων, αλλά έχει και ένα ορισμό, μέσω μιας επερώτησης που ανάγει τον υπολογισμό του κύβου σε μια σειρά πράξεις πάνω στο αποθηκευμένο λεπτομερές σύνολο δεδομένων.

Τυπικά ένας κύβος c είναι μία τετράδα από ένα λεπτομερές σύνολο δεδομένων, μια συνθήκη επιλογής πάνω σε λεπτομερή επίπεδα, επίπεδα συνάθροισης και μέτρα συνάθροισης και αθροιστικές συναρτήσεις από το σύνολο $\{sum,min,max,count\}$ στα λεπτομερή μέτρα.

Για να περιγραφτεί διαισθητικά ο υπολογισμός ενός κύβου, εφαρμόζεται πρώτα η συνθήκη επιλογής στα λεπτομερή δεδομένα. Στη συνέχεια, αντικαθίστανται οι τιμές των επιπέδων των πλειάδων του αποτελέσματος με τις αντίστοιχες τιμές των προγόνων τους (στα επίπεδα του σχήματος του κύβου) και στη συνέχεια συναθροίζονται τις πλειάδες, παράγοντας μία μόνο τιμή για κάθε μέτρο, μέσω της αντίστοιχης αθροιστικής συνάρτησης. Ένα σύνολο δεδομένων μπορεί να εκφραστεί σαν κύβος, χρησιμοποιώντας την εκφυλισμένη συνθήκη επιλογής $true$. Για παράδειγμα, ο κύβος του λεπτομερούς συνόλου δεδομένων DS^0 του σχήματος 2 εκφράζεται ως:

$c^0=(DS^0,true,[day,day,item,salesman,city,sales],sum(sales))$.

Η προσέγγιση αυτή εισάγει ένα ισχυρό μηχανισμό έκφρασης, ικανό να μοντελοποιήσει απ' ευθείας πράξεις όπως η αναλυτική κάθοδος και η αλλαγή της

αθροιστικής συνάρτησης, με απώτερο στόχο τη μοντελοποίηση σειρών πράξεων, όπως συμβαίνει στα OLAP συστήματα. Η αναγωγή του ορισμού του κύβου σε μια κανονικοποιημένη μορφή φαίνεται να είναι η μόνη εναλλακτική λύση που προσφέρει αυτή τη δυνατότητα.

Τυπικά, το μοντέλο αποτελείται από τα παρακάτω στοιχεία:

- Κάθε *διάσταση* (*dimension*) D είναι ένα δίκτυο (lattice) $(\mathbf{L}, \prec)$ τέτοιο ώστε: $\mathbf{L}=(L_1,\ldots,L_n,ALL)$ είναι ένα πεπερασμένο υποσύνολο από *επίπεδα* (*levels*) και $\prec$ είναι μια σχέση μερικής διάταξης ορισμένη μεταξύ των επιπέδων του $\mathbf{L}$, έτσι ώστε $L_1 \prec L_i \prec ALL$ για κάθε $1 \le i \le n$.
- Μια οικογένεια συναρτήσεων anc_{L1}^{L2} που ικανοποιεί τις παρακάτω συνθήκες (επεκτείνοντας την πρόταση του [CaTo97]):
  1. Για κάθε ζεύγος επιπέδων L_1 και L_2 έτσι ώστε $L_1 \prec L_2$, η συνάρτηση anc_{L1}^{L2} απεικονίζει κάθε στοιχείο του $dom(L_1)$ σε ένα στοιχείο του $dom(L_2)$.
  2. Δοθέντων των επιπέδων L_1, L_2 και L_3 έτσι ώστε $L_1 \prec L_2 \prec L_3$, η συνάρτηση anc_{L1}^{L3} ισοδυναμεί με τη σύνθεση $anc_{L1}^{L2} \circ anc_{L2}^{L3}$.
  3. Για κάθε ζεύγος επιπέδων L_1 και L_2 έτσι ώστε $L_1 \prec L_2$, η συνάρτηση anc_{L1}^{L2} είναι μονότονη, ήτοι, $\forall x,y \in dom(L_1), L_1 \prec L_2 : x<y \Rightarrow anc_{L1}^{L2}(x) \le anc_{L1}^{L2}(y)$.
  4. Για κάθε ζεύγος επιπέδων L_1 και L_2 η συνάρτηση anc_{L1}^{L2} καθορίζει ένα σύνολο από πεπερασμένες κλάσεις ισοδυναμίες X_i έτσι ώστε: $\forall x,y \in dom(L_1), L_1 \prec L_2 : anc_{L1}^{L2}(x)=anc_{L1}^{L2}(y) \Rightarrow x,y$ ανήκουν στο ίδιο X_i.
  5. Η σχέση $desc_{L1}^{L2}$ είναι η αντίστροφη της συνάρτησης anc_{L1}^{L2} -τουτέστιν,

     $desc_{L1}^{L2}(l) = \{x \in dom(L) : anc_{L1}^{L2}(x) = l\}$.
- Κάθε *σύνολο δεδομένων* (*data set*) DS ορισμένο πάνω σε ένα σχήμα $S=[L_1,\ldots,L_n,M_1,\ldots,M_m]$ είναι ένα πεπερασμένο σύνολο πλειάδων ορισμένων πάνω στο S έτσι ώστε το $[L_1,\ldots,L_n]$ να είναι πρωτεύον κλειδί (με τη συνήθη έννοια του όρου).
- Κάθε *συνθήκη επιλογής* (*selection condition*) φ είναι μια έκφραση σε διαζευκτική κανονική μορφή. Ένα *άτομο* (*atom*) της συνθήκης επιλογής είναι της μορφής $true$, $false$ ή $x \ \theta \ y$, όπου θ είναι ένας τελεστής από το σύνολο ($>$, $<$, $=$, $\ge$, $\le$, $\ne$) και

κάθε ένα εκ των x και y μπορεί να είναι ένα από τα παρακάτω: (α) ένα επίπεδο L, (β) μια τιμή l, (γ) μια έκφραση της μορφής $anc_{L1}^{L2}(L_1)$ όπου $L_1 \prec L_2$ και (δ) μια έκφραση της μορφής $anc_{L1}^{L2}(l)$ όπου $L_1 \prec L_2$ και $l \in dom(L_1)$. Το *λεπτομερές ισοδύναμο της* φ (*detailed equivalent of* φ), συμβολιζόμενο με φ^0, είναι μια συνθήκη επιλογής που προκύπτει από την εξής διαδικασία: κάθε στιγμιότυπο του ονόματος ενός επιπέδου L στη φ, αντικαθίσταται από την ισοδύναμη έκφραση $anc_{L^0}^{L}(L^0)$, όπου το L^0 είναι το λεπτομερές επίπεδο της διάστασης στην οποία ανήκει το L.

– Κάθε κύβος (*cube*) c ορισμένος πάνω στο σχήμα $[L_1,\ldots,L_n,M_1,\ldots,M_m]$, είναι μια έκφραση της μορφής: $c=(DS^0,\varphi,[L_1,\ldots,L_n,M_1,\ldots,M_m],[agg_1(M_1^0),\ldots,agg_m(M_m^0)])$, όπου το DS^0 είναι ένα λεπτομερές σύνολο δεδομένων ορισμένο πάνω στο σχήμα $S=[L_1^0,\ldots,L_n^0,M_1^0,\ldots,M_k^0]$, $m \le k$, φ είναι μια λεπτομερής συνθήκη επιλογής, $M_1^0,\ldots,M_m^0$ είναι λεπτομερή μέτρα, $M_1,\ldots,M_m$ είναι συναθροισμένα μέτρα, L_i^0 και L_i είναι επίπεδα τέτοια ώστε $L_i^0 \prec L_i$, $1 \le i \le n$ και agg_i, $1 \le i \le m$ είναι αθροιστικές συναρτήσεις από το σύνολο $\{sum,min,max,count\}$. Η έκφραση που χαρακτηρίζει τον κύβο έχει την παρακάτω σημασιολογία:

$$c=\{x \in dom(L_1)\times\ldots\times dom(L_n)\times dom(M_1)\times\ldots\times dom(M_m) \mid \exists y \in \varphi(DS^0), x[L_i]=anc_{L_i^0}^{L_i}(y[L_i^0]),\ 1 \le i \le n,\ x[M_j]=agg_j(\{q \mid \exists z \in \varphi(DS^0), x[L_i]=anc_{L_i^0}^{L_i}(z[L_i^0]), 1 \le i \le n, q=z[M_j^0]\}), 1 \le j \le m\}.$$

Η *Άλγεβρα Κύβων* (*Cube Algebra* -CA) αποτελείται από τρεις λειτουργίες:

1. *Πλοήγηση* (*Navigate*): Έστω $S=[L_1,\ldots,L_n,M_1,\ldots,M_m]$ ένα σχήμα και $agg_1,\ldots,agg_m$ αθροιστικές συναρτήσεις. Εάν L_i^a και L_i ανήκουν στην ίδια διάσταση D_i και $agg_i \in \{sum,min,max,count\}$ τότε η πλοήγηση ορίζεται ως ακολούθως:
   - $nav(c^a,S,agg_1,\ldots,agg_m)=(DS^0,\varphi^a,S,[agg_1(M_1^0),\ldots,agg_m(M_m^0)])$.
2. *Επιλογή* (*Selection*): Έστω φ μια συνθήκη επιλογής εφαρμόσιμη στο c^a. Τότε, ορίζουμε την πράξη επιλογής ως:

$$\sigma_{\varphi}(c^{a}) = (DS^{0}, \varphi^{a} \wedge \varphi^{0}, [L_1^a, \ldots, L_n^a, M_1^a, \ldots, M_m^a], [agg_1(M_1^0), \ldots, agg_m(M_m^0)])$$

όπου το φ^0 είναι το λεπτομερές ισοδύναμο της συνθήκης επιλογής φ.

3. *Αποκόλληση Μέτρου* (*Split measure*): Έστω M ένα μέτρο του σχήματος του κύβου c. Χωρίς βλάβη της γενικότητας, έστω ότι το M είναι το M_m. Τότε, η αποκόλληση μέτρου ορίζεται ως εξής:

$$\pi_{M_m}(c^{a}) = (DS^{0}, \varphi^{a}, [L_1^a, \ldots, L_n^a, M_1^a, \ldots, M_{m-1}^a], [agg_1(M_1^0), \ldots, agg_m(M_{m-1}^0)]).$$

Παράδειγμα 3.1. Για να υποστηριχτεί τη συζήτηση, προσαρμόζεται το παράδειγμα του [Micr98] σε μια διεθνή εκδοτική εταιρεία με ταξιδεύοντες πωλητές που πουλάνε βιβλία και CD σε βιβλιοπωλεία σε όλο τον κόσμο. Η βάση δεδομένων (σχήματα 3.1-3.5) αποθηκεύει πληροφορία για τις πωλήσεις που ένας πωλητής πέτυχε σε μια πόλη, μια δεδομένη χρονική στιγμή. Οι διαστάσεις του παραδείγματος είναι `Person` (σχήμα 3.2), `Location` (σχήμα 3.3), `Product` (σχήμα 3.4) και `Date` (σχήμα 3.5). Το μέτρο `Sales` είναι συναρτησιακά εξαρτημένο από τις διαστάσεις `Date`, `Product`, `Person` και `Location`.

Η οργάνωση της πληροφορίας σε διαφορετικά επίπεδα συνάθροισης (διαστάσεις, δηλαδή) είναι απαραίτητη για τον απλό λόγο ότι οι χρήστες είναι μάλλον απίθανο να κάνουν ερωτήσεις απ' ευθείας στα λεπτομερή δεδομένα. Αντίθετα, ενδιαφέρονται περισσότερο για την συναθροισμένη πληροφορία που τους δίνει μια γενικότερη εικόνα της κατάστασης, και ζητούν την περαιτέρω ανάλυσή της μόνο σε ειδικές περιπτώσεις (π.χ. στα καταστήματα εκείνων των πόλεων που έκαναν συνολικά τις υψηλότερες ή χαμηλότερες πωλήσεις).

Στη συνέχεια παρουσιάζονται τρεις επερωτήσεις και η αντίστοιχη αλγεβρική αναπαράστασή τους. Οι επερωτήσεις αυτές θα μπορούσαν κάλλιστα να αποτελούν μια σειρά πράξεων σε μια OLAP σύνοδο. Τα αποτελέσματα των επερωτήσεων φαίνονται στα σχήματα 3.6, 3.7 και 3.8, αντίστοιχα.

Επερώτηση 1. Βρες τις μέγιστες πωλήσεις ανά μήνα, κατηγορία προϊόντος, πωλητή και χώρα.

$$c^1=nav(DS^0,[Month,Category,Salesman,Country,Max\_val],max(sales))=$$
$$(DS^0,true,[Month,Category,Salesman,Country,Max\_val],max(sales)).$$

Επερώτηση 2. Βρες τις μέγιστες πωλήσεις εκτός Αμερικανικής ηπείρου ανά μήνα, κατηγορία προϊόντος, πωλητή και χώρα.

$$c^2=\sigma_{anc^{continent}_{country}(country)\neq \text{'America'}}(c^1)=(DS^0,anc^{continent}_{city}(City)\neq 'America',$$
$$[Month,Category,Salesman,Country,Max\_val],max(sales)).$$

Επερώτηση 3. Βρες το άθροισμα των πωλήσεων εκτός Αμερικανικής ηπείρου ανά μήνα, τύπο προϊόντος και χώρα.

$$c^3=nav(c^2,[Month,Type,ALL,Country,Sum\_val],sum(Sales))=$$
$$(DS^0,anc^{continent}_{city}(City)\neq 'America',[Month,Type,ALL,Country,Sum\_val],$$
$$sum(sales)).$$

Στη διάρκεια αυτής της συνόδου, ο χρήστης έκανε τα εξής:

1. μια συναθροιστική άνοδος από το λεπτομερές σύνολο δεδομένων,
2. μια επιλογή,
3. ένας τεμαχισμός (της διάστασης `Person`) συνδυασμένος με μια αναλυτική κάθοδο (από το επίπεδο `Category` στο επίπεδο `Type`) και μια αλλαγή αθροιστικής συνάρτησης (από `max` σε `sum`).

Στην πρώτη λειτουργία φαίνεται ότι η σημασιολογία της πλοήγησης επιτρέπει να χρησιμοποιηθεί ένα οποιοδήποτε όνομα (π.χ., `Max_val`) για το μέτρο που υπολογίζει τη μέγιστη τιμή ανά ομάδα συνάθροισης.

Στη δεύτερη λειτουργία η έκφραση $anc^{continent}_{country}(Country)$ που είναι απευθείας εφαρμόσιμη στο σχήμα (και τα δεδομένα) του κύβου c^1 μετασχηματίζεται στην ισοδύναμή της $anc^{continent}_{city}(City)$, η οποία είναι εφαρμόσιμη στο λεπτομερές σύνολο δεδομένων DS^0, μέσω του ορισμού της έννοιας της λεπτομερούς συνθήκης επιλογής.

Το μοντέλο αυτό υπογραμμίζει την ιδέα *ότι ο κύβος μπορεί να αντιμετωπισθεί ταυτόχρονα και σαν επερώτηση και σαν σύνολο πλειάδων*. Στο παράδειγμα που μόλις αναφέρθηκε είναι εμφανές ότι ήταν το γεγονός ότι κρατήθηκε η ιστορία των επιλογών, που επέτρεψε να γίνει αναλυτική κάθοδος και να αλλάξει η αθροιστική συνάρτηση. Η εναλλακτική αντιμετώπιση για την αναλυτική κάθοδο θα ήταν κάποια

πράξη σύνδεσης του c^2 με το DS^0. Το ίδιο ισχύει επίσης και για την αλλαγή της αθροιστικής συνάρτησης. Χρησιμοποιώντας την ιστορία των επιλογών μπορεί (α) να αποφευχθεί η εκτέλεση μιας δαπανηρής πράξης σύνδεσης και (β) να βελτιστοποιηθεί πιθανώς περισσότερο η εκτέλεση μιας λειτουργίας με τη χρήση ήδη υπολογισμένων κύβων.

Όπως έχει τονιστεί αυτό είναι ένα *λογικό* μοντέλο κύβων. Δεν υποστηρίζεται ότι ο *φυσικός* υπολογισμός των αποτελεσμάτων θα έπρεπε να εκτελείται από το λεπτομερή κύβο. Ενώ η αναλυτική κάθοδος και η αλλαγή αθροιστικής συνάρτησης δεν ξεφεύγουν από αυτό τον κανόνα, οι επιλογές και οι συναθροιστικές αναβάσεις μπορούν αντίθετα να εκτελεστούν τοπικά. Στην περίπτωση επιλογής αρκεί να περάσουν τα δεδομένα από το φίλτρο της συνθήκης επιλογής. Στην περίπτωση της συναθροιστικής ανόδου σε υψηλότερα επίπεδα λεπτομέρειας αρκεί η κατάλληλη ομαδοποίηση των πλειάδων και η εφαρμογή της κατάλληλης αθροιστικής συνάρτησης. Οι απλές αυτές διαπιστώσεις γενικεύονται στην επόμενη ενότητα με μια ισχυρότερη προσέγγιση ικανή να εντοπίσει εάν ένας κύβος μπορεί να υπολογισθεί από τα δεδομένα ενός άλλου κύβου απλά συγκρίνοντας τους ορισμούς τους.

Month	Category	Salesman	Country	Max val
Feb 97	Books	Netz	France	7
Feb 97	Books	Netz	USA	5
May 97	Music	Netz	USA	20
Sept 97	Books	Netz	Japan	50
July 97	Music	Venk	Greece	34

Σχήμα 3.6: Κύβος c^1 - Πλοήγηση

Month	Category	Salesman	Country	Max val
Feb 97	Books	Netz	France	7
Sept 97	Books	Netz	Japan	50
July 97	Music	Venk	Greece	34

Σχήμα 3.7: Κύβος c^2 - Επιλογή

Month	Type	ALL	Country	Sum_val
Feb 97	Philosophy	All	France	7
Sep 97	Philosophy	All	Japan	50
Sep 97	Literature	All	Japan	30
Jul 97	Heavy Metal	All	Greece	47

Σχήμα 3.8: Κύβος c^3 - Σύνθετη σειρά πράξεων

Θεώρημα 3.1. Η Άλγεβρα Κύβων CA είναι *συνεπής* (το αποτέλεσμα, δηλαδή, όλων των λειτουργιών είναι πάντα κύβος) και *πλήρης* (κάθε κύβος, δηλαδή, μπορεί να υπολογιστεί από ένα συνδυασμό των πράξεων της άλγεβρας CA).

3.2 Το πρόβλημα της καταλληλότητας των κύβων

Περιγραφή του προβλήματος. Υπάρχουν περιπτώσεις όπου υπάρχει η ανάγκη να αποφασιστεί αν μια όψη μπορεί να χρησιμοποιηθεί για να υπολογιστεί μια άλλη όψη. Δύο γνωστά παραδείγματα είναι τα εξής: (α) οι χρήστες OLAP εργαλείων κάνουν αλληλεπιδραστικές πλοηγήσεις στα δεδομένα, με αλλαγές στη λεπτομέρεια της παρουσίασής τους και (β) ο σχεδιαστής της συγκεντρωτικής αποθήκης δεδομένων έχει να επιλέξει, ανάμεσα σε πολλές υποψήφιες, ποιες όψεις θα υλοποιήσει. Στην πρώτη περίπτωση, ο χρήστης επιλέγει κάποια δεδομένα και κάνει κάποια πράξη πάνω τους. Το αποτέλεσμα της νέας λειτουργίας μπορεί φυσικά να υπολογισθεί από τα λεπτομερή δεδομένα, αλλά είναι δυνατόν και να υπολογισθεί από κύβους προϋπολογισμένους ή προσωρινώς αποθηκευμένους στη λανθάνουσα μνήμη. Στη δεύτερη περίπτωση, ο σχεδιαστής της συγκεντρωτικής αποθήκης δεδομένων χρειάζεται κάποιους αλγόριθμους για να αποφασίσει αν θα αποθηκεύσει επιπλέον όψεις (πιθανά επικαλυπτόμενες) ώστε οι ερωτήσεις των χρηστών να απαντώνται πιο γρήγορα. Ορισμένες φορές, ο πλεονασμός όψεων μπορεί να επιταχύνει και τη διαδικασία ανανέωσης [ThLS99, LSTV99, Gupt97]. Τμήμα του αλγορίθμου σχεδίασης είναι και μια μέθοδος που αποφασίζει αν μια όψη μπορεί να χρησιμοποιηθεί για να υπολογίσει μια άλλη όψη (ή εν γένει μια ερώτηση). Γενικεύοντας τα παραπάνω, μπορεί κανείς να πει ότι το πρόβλημα έγκειται στην απόκριση του κατά πόσον μπορεί ένας κύβος να υπολογισθεί από ένα ενδιάμεσο επίπεδο συνάθροισης αντί από το λεπτομερές σύνολο δεδομένων.

Τυπικά, έστω DS^0 ένα λεπτομερές σύνολο δεδομένων. Έστω επίσης C^{old} και C^{new} δύο κύβοι ορισμένοι πάνω στο DS^0. Εξ' ορισμού, οι κύβοι C^{old} και C^{new} μπορούν να υπολογισθούν από το DS^0. Το *πρόβλημα της καταλληλότητας κύβων* (*cube usability problem*) έγκειται στην απόφαση, εάν οι πλειάδες του C^{old} μπορούν να χρησιμοποιηθούν για τον υπολογισμό του C^{new}. Είναι σαφές ότι πρόβλημα είναι

υποπερίπτωση του προβλήματος της *υπαγωγής όψεων* (*view subsumption*), που έχει ήδη εξερευνηθεί αρκετά στο χώρο των σχεσιακών βάσεων δεδομένων [Ullm97].

Προβλήματα των υπαρχόντων προσεγγίσεων. Έχει προϋπάρξει σημαντική δουλειά στο παρελθόν για την επίλυση του προβλήματος της υπαγωγής όψεων και της επανεγγραφής ερωτήσεων όταν είναι παρούσες αποθηκευμένες όψεις [NuSS98, CoNS99, DJLS96, CKPS95, GuHQ95, LMSS95, ChSh96, YaLa85]. Παρ' όλα αυτά, οι προηγούμενες προσεγγίσεις είναι προσαρμοσμένες στο σχεσιακό μοντέλο και αδυνατούν να εκμεταλλευθούν τα ιδιαίτερα χαρακτηριστικά της πολυδιάστατης μοντελοποίησης κύβων. Θα δωθούν δύο παραδείγματα για να αναδειχθούν τα προβλήματα αυτά.

Παράδειγμα 3.2 Διαισθητικά, θα περίμενε κανείς ότι για να λυθεί το πρόβλημα της καταλληλότητας κύβων ο νέος κύβος c^{new} θα έπρεπε:

1. Να είναι ορισμένος στις ίδιες διαστάσεις με τον c^{old} και σε υψηλότερο ή ίσο επίπεδο.
2. Να είναι ορισμένος στο ίδιο μέτρο του DS^0 και επιπλέον η αθροιστικές συναρτήσεις agg^{new} και agg^{old} να είναι ίδιες.
3. Να έχει μια συνθήκη επιλογής πιο περιορισμένη από το c^{old}, τουτέστιν η φ^{new} να εγκλείεται στην φ^{old} με το συνήθη σχεσιακό τρόπο.

Ο έλεγχος των συνθηκών 1 και 2 είναι εύκολος, φυσικά. Για να γίνει όμως η σύγκριση της συνθήκης 3, πρέπει να μετασχηματιστούν οι συνθήκες επιλογής των δύο κύβων ώστε να αντιμετωπιστούν σα συζευκτικές επερωτήσεις [Ullm89]. Θα δειχθεί ότι οι υπάρχουσες σχεσιακές τεχνικές δεν επαρκούν για λύσουν το πρόβλημα.

Cube c^1

Month	Sales
Feb	5
May	20
Sep	80

Cube c^2

Month	Sales
Feb	**12**
May	20
Sep	80

Cube c^3

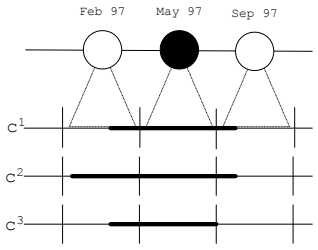
Month	Sales
Feb	5
May	20

φ_1=18-Feb-97≤day≤3-Sep-97 ∧Salesman=Netz

φ_2=6-Feb-97≤day≤3-Sep-97∧Salesman=Netz

φ_3=18-Feb-97≤day≤31-May-97∧Salesman=Netz

(α)



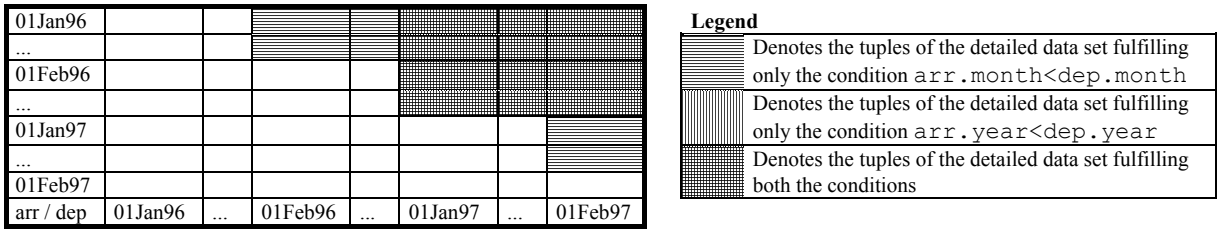
(β)

Σχήμα 3.2 Προβλήματα καταλληλότητας κύβων

Έστω το λεπτομερές σύνολο δεδομένων DS^0 του σχήματος 2. Έστω c^i, $1 \le i \le 3$ κύβοι ορισμένοι ως $c^i=[DS^0, \varphi_i, [Month,ALL,ALL,ALL,ALL,ALL,ALL,Sales], sum(sales)]$. Το σχήμα 10a περιγράφει το επίπεδο Month, το μέτρο Sales και τις συνθήκες επιλογής για κάθε κύβο. Το πρόβλημα είναι αν ο (νέος) κύβος c^3 μπορεί να υπολογισθεί από τις πλειάδες των (ήδη υπαρχόντων) κύβων c^1 και c^2. Αφού οι συνθήκες 1, 2 και 3 ισχύουν, θα μπορούσε κανείς να ισχυρισθεί ότι αυτό είναι εφικτό. Όμως, όπως φαίνεται και στο σχήμα 10α, μόνο ο c^1 μπορεί να χρησιμοποιηθεί για τον υπολογισμό του c^3. Η διαισθητική επεξήγηση του προβλήματος φαίνεται στο σχήμα 10β. Υπάρχουν τρεις οριζόντιοι άξονες ορισμένοι στο επίπεδο day για κάθε έναν από τους κύβους c^1, c^2 και c^3. Οι έντονες γραμμές δείχνουν το σύνολο των ημερών που συμμετέχουν στον υπολογισμό του αντίστοιχου κύβου. Ο κύβος c^3 είναι ορισμένος στο επίπεδο month και κατά συνέπεια διαχωρίζουμε τους τρεις άξονες σε σχέση με τη συνάρτηση anc_{day}^{month}. Όπως φαίνεται υπάρχουν τρία τμήματα διαχωρισμού: Feb'97, May'97 και Sep'97. Ο κύβος c^3 μπορεί να υπολογισθεί από τον c^1 επειδή για όλα τα τμήματα διαχωρισμού του c^3 (i.e., Feb'97, May'97), οι κύβοι c^1 και c^3 καλύπτουν ακριβώς τις ίδιες μέρες. Αυτό δεν ισχύει για τους κύβους c^1 και c^2.

Παράδειγμα 3.3. Έστω η περίπτωση που ο κύβος c^1 έχει μια συνθήκη επιλογής φ_1=arr.year<dep.year (όπου arr είναι συντομογραφία για τη διάσταση arrival date και dep για τη διάσταση departure date). Έστω ακόμα ότι ο κύβος c^2 είναι ορισμένος στο επίπεδο μήνα και έχει μια συνθήκη επιλογής φ_2=arr.month<dep.month. Φαίνεται ότι ο κύβος c^1 μπορεί να υπολογισθεί από τον c^2. Αυτό σημαίνει ότι αν ο c^2 είναι αποθηκευμένος, μπορούν να χρησιμοποιηθούν οι πλειάδες του για τον υπολογισμό του c^1. Μπορεί να γίνει αυτού του είδους ο συλλογισμός εκμεταλλευόμενοι τη σχέση μηνών και χρόνων, η οποία εκφράζεται από την ιεραρχία της χρονικής διάστασης μέσω των συναρτήσεων anc.

Στο Σχήμα 3.3 παρουσιάζεται μια γραφική απεικόνιση του προβλήματος. Οι πλειάδες του λεπτομερούς κύβου αναπαριστώνται σαν κελιά στο δισδιάστατο χώρο. Ο οριζόντιος άξονας αναπαριστά τη διάσταση departure date και ο κάθετος άξονας τη διάσταση arrival date (χάριν ευκολίας αγνοούμε τις υπόλοιπες διαστάσεις του παραδείγματος). Όπως μπορεί να παρατηρήσει κανείς, οι πλειάδες του λεπτομερούς συνόλου δεδομένων που πληρούν την συνθήκη arr.month<dep.month είναι γνήσιο υποσύνολο των πλειάδων που πληρούν τη συνθήκη arr.month<dep.month.



Σχήμα 3.3 Γραφική αναπαράσταση της αξιολόγησης των δεδομένων για το πρόβλημα της καταλληλότητας κύβων

Συνεισφορά. Στην ενότητα αυτή θα δειχθεί ότι το πρόβλημα καταλληλότητας κύβων ανάγεται σε απλούς ελέγχους και πράξεις. Διαφορετικοί έλεγχοι χρησιμοποιούνται για διαφορετικές κλάσεις ερωτήσεων. Θα ερευνηθούν οι συνθήκες επιλογής δύο κατηγοριών: (α) συνθήκες επιλογής με άτομα που περιέχουν τιμές (της μορφής, δηλαδή, $L\theta l, L\theta anc_{L1}^{L2}(l)$, κλπ.) και (β) συνθήκες επιλογής με άτομα που περιέχουν μόνο επίπεδα (της μορφής, δηλαδή, $L_1\theta L_2$, $L\theta anc_{L1}^{L2}(L_1)$, κλπ

Στο υπόλοιπο κεφάλαιο, για λόγους απλότητας θα θεωρηθεί ότι οι κύβοι έχουν μόνο ένα μέτρο. Όλα τα αποτελέσματα επεκτείνονται εύκολα σε κύβους με περισσότερα μέτρα [NuSS98]. Έστω $c^{new}=(DS^0, \varphi^{new}, [\mathbf{L}^{new}, M^{new}], agg^{new}(M))$ ο νέος κύβος και $c^{old}=(DS^0, \varphi^{old}, [\mathbf{L}^{old}, M^{old}], agg^{old}(M))$ ο υποψήφιος κύβος, όπου $\mathbf{L}^{new}$ και $\mathbf{L}^{old}$ είναι σύνολα επιπέδων από τα σύνολα διαστάσεων $\mathbf{D}^{new}$ και $\mathbf{D}^{old}$ αντίστοιχα, M^{new} και M^{old} είναι μέτρα, και τέλος, agg^{new} και agg^{old} είναι αθροιστικές συναρτήσεις.

3.3 Ισοδύναμοι μετασχηματισμοί για άτομα που εμπλέκουν τιμές

Έστω δύο επίπεδα L^{old} και L^{new}, τέτοια ώστε $L^{old} \prec L^{new}$. Η συνάρτηση $anc_{L^{old}}^{L^{new}}$ ορίζει ένα διαχωρισμό των τιμών του L^{old} σε σχέση με τις τιμές του L^{new} (π.χ. ο διαχωρισμός με βάση το year στο επίπεδο month). Έστω ακόμα δύο άτομα a_1 και a_2 ορισμένα στο L^{old}. Για να γίνει μια συνάθροιση στο L^{new}, τα δύο άτομα πρέπει να έχουν τα ίδια εύρη τιμών για κάθε τμήμα διαχωρισμού που ορίζει το L^{new} πάνω στο L^{old}.

Γενικεύοντας την παρατήρηση αυτή, στην περίπτωση που δύο συνθήκες επιλογής περιέχουν ένα μεγαλύτερο αριθμό ατόμων, πρέπει:

(1) Να μετασχηματιστούν οι συνθήκες επιλογής σε συνεχή εύρη τιμών για κάθε διάσταση.

(2) Να αναχθούν τα άτομα στο ίδιο επίπεδο χρησιμοποιώντας τους κατάλληλους μετασχηματισμούς (ώστε να μπορούν να συγκριθούν).

(3) Να ελεγχθεί αν η ευρύτερη συνθήκη επιλογής είναι κατάλληλα ορισμένη για τις οριακές συνθήκες της άλλης συνθήκης επιλογής

Ο παρακάτω βοηθητικός ορισμός εισάγει την έννοια του *διαστήματος διάστασης* (*dimension interval*), που είναι ένα συμπαγές εύρος τιμών πάνω στο πεδίο ορισμού ενός επιπέδου.

Ορισμός 3.1: Ένα *διάστημα διάστασης* (*dimension interval* -DI) είναι ένα από τα παρακάτω (α) `true`, (β) `false` και (γ) μια έκφραση της μορφής $l_1 \le L \le l_2$, όπου το `L` είναι μια μεταβλητή που αναπαριστά ένα επίπεδο μιας διάστασης και l_1 και l_2 είναι τιμές.

Atom	Dimension Interval	Atom	Dimension Interval
True	true	$anc_L^{L'}(L) < l$	$-\infty < L \le max(desc_L^{L'}(prev(l)))$
False	false	$l \le anc_L^{L'}(L)$	$min(desc_L^{L'}(l)) \le L < +\infty$
$anc_L^{L'}(L) = l$	$min(desc_L^{L'}(l)) \le L \le max(desc_L^{L'}(l))$	$l < anc_L^{L'}(L)$	$min(desc_L^{L'}(next(l))) \le L < +\infty$
$anc_L^{L'}(L) \le l$	$-\infty < L \le max(desc_L^{L'}(l))$		

Σχήμα 3.4 Μετασχηματίζοντας άτομα σε διαστήματα διάστασης

Το σχήμα 3.4 δείχνει πώς απλά άτομα μπορούν να μετασχηματιστούν σε διαστήματα διάστασης. Οι τιμές $-\infty$ και $+\infty$ έχουν την προφανή σημασιολογία. Οι συναρτήσεις `prev` και `next` επιστρέφουν την προηγούμενη και την επόμενη τιμή του `l` στο πεδίο ορισμού του `L` αντίστοιχα.

Εν γένει, για να αποφασιστεί αν ένας κύβος c^{old} μπορεί να χρησιμοποιηθεί για τον υπολογισμό του c^{new}, πρέπει να διαχωριστεί το λεπτομερές επίπεδο κάθε διάστασης με βάση το αντίστοιχο επίπεδο του c^{new}. Αν για κάθε τμήμα διαχωρισμού του c^{new}, υπάρχει ένα ταυτοτικό τμήμα διαχωρισμού του c^{old}, τότε ο c^{old} μπορεί να χρησιμοποιηθεί για να υπολογιστεί ο c^{new}. Τυποποιείται αυτή η σχέση αυτή, μέσω του Ορισμού 3.2

Ορισμός 3.2. Εγκλεισμός L: Έστω `D` ένα σύνολο διαστάσεων και φ^{old}, φ^{new} δύο συνθήκες επιλογής που περιλαμβάνουν επίπεδα μόνο από το `D`. Έστω **L** ένα σύνολο επιπέδων που το καθένα ανήκει σε διαφορετική διάσταση του `D`. Έστω ακόμα και δύο κύβοι $c^{new} = (DS^0, \varphi^{new}, [\mathbf{L}, M], agg(M))$ και $c^{old} = (DS^0, \varphi^{old}, [\mathbf{L}, M], agg(M))$, ορισμένοι πάνω σε

ένα τυχαίο σύνολο δεδομένων DS^0. Η συνθήκη επιλογής φ^{new} *εγκλείεται κατά* **L** (**L**-*contained*) στη φ^{old} (που συμβολίζεται $\varphi^{new} \subseteq_{\mathbf{L}} \varphi^{old}$) αν $c^{new} \subseteq c^{old}$ για οποιοδήποτε σύνολο δεδομένων DS^0.

Algorithm `Check_Atoms_Usability`.
Input: Two conjunctions of atoms **a** και **b** involving only values, και a set of levels **L'** .
Output: true if $\mathbf{a} \subseteq_{\mathbf{L}'} \mathbf{b}$, false otherwise.

1. Write all atoms of **a** και **b** as DI's using the transformations of Figure 5.12.
2. Group all DI's of **a** και **b** by dimension level και produce for every set a single DI having the most restrictive boundaries. Let **a'** και **b'** be the result, respectively.
3. For every DI a of **a'**
4. If a is defined over dimension level $D_i.L^0$ that does not exist in any DI of **b'** Then
5. Introduce DI $-\infty \le D_i.L^0 \le \infty$ to **b'** .
6. EndFor
7. `flag = false`
8. For every DI a of **a'**
9. `flag = false`
10. For every DI b of **b'**
11. For every dimension level L' of **L'** involved in b
12. Case $A_s < B_s$ or $B_e < A_e$ or b=false
13. `flag = true`
14. Case $L \neq L'$ και $A_s \neq \min(desc_L^{L'}(anc_L^{L'}(A_s)))$ και $A_s \neq B_s$
15. `flag = false`
16. Case $L \neq L'$ και $A_e \neq \max(desc_L^{L'}(anc_L^{L'}(A_e)))$ και $A_e \neq B_e$
17. `flag = false`
18. Default
19. `flag = true`
20. EndFor
21. EndFor
22. If `flag = false` Then
23. Return `false`
24. EndFor
25. Return `true`

Σχήμα 3.5 Αλγόριθμος `Check_Atoms_Usability`

Για να αντιμετωπιστεί το πρόβλημα της καταλληλότητας κύβων μεταξύ κύβων διαφορετικών επιπέδων λεπτομέρειας, αρχίζει ο έλεγχος από τον εγκλεισμό των συνθηκών επιλογής τους. Η ανάλυση προς το παρόν δεν καλύπτει την περίπτωση του $\neq$, αλλά αυτό θα αντιμετωπισθεί στην υποενότητα 3.4.

Ο Αλγόριθμος `Check_Atoms_Usability` του σχήματος 3.5 παίρνει ως είσοδο δύο συζεύξεις ατόμων, **a** και **b**, που αφορούν μόνο τιμές. Ο αλγόριθμος επιστρέφει `true` αν το **a** εγκλείει κατά **L** το **b** σε σχέση με το σύνολο επιπέδων **L'**, και `false` σε κάθε άλλη περίπτωση. Ο αλγόριθμος προχωρεί ως ακολούθως. Αρχικά, ο Αλγόριθμος `Check_Atoms_Usability` επανεγγράφει όλα τα άτομα των **a** και **b** σε διαστήματα

διάστασης χρησιμοποιώντας τους μετασχηματισμούς του σχήματος 3.4 (Γραμμή 1). Έπειτα ομαδοποιεί όλα τα διαστήματα διάστασης των **a** και **b** ανά επίπεδο και παράγει για κάθε σύνολο, ένα μόνο διάστημα διάστασης με τα πλέον περιορισμένα άκρα. Το αποτέλεσμα αποθηκεύεται στα σύνολα διαστημάτων διάστασης **a′** και **b′** αντίστοιχα (Γραμμή 2). Οι Γραμμές 3-6 ελέγχουν εάν υπάρχει επίπεδο $D_i.L$ του διαστήματος διάστασης a που ανήκει στο **a′**, το οποίο δεν ανήκει σε κανένα διάστημα διάστασης του **b′**. Στην περίπτωση αυτή, ο αλγόριθμος εισάγει το διάστημα $-\infty \leq D_i.L^0 \leq +\infty$ στο **b′** (Γραμμή 5). Τέλος οι γραμμές 7-24 ελέγχουν αν για κάθε διάστημα διάστασης b του **b** υπάρχει ένα διάστημα διάστασης a στο **a** τέτοιο ώστε $b \subseteq_{L'} a$ για κάποιο επίπεδο $L' \in$ **L′**. Ειδικότερα, οι γραμμές 12-21 ελέγχουν κατά πόσον το διάστημα διάστασης a είναι εγκλεισμένο κατά **L** σε σχέση με το διάστημα διάστασης b. Οι γραμμές 12-13 ελέγχουν αν το διάστημα διάστασης b είναι ευρύτερο του a. Ο αλγόριθμος επιστρέφει `true` αν τα οριακά τμήματα διαχωρισμού είναι ταυτοτικά ίσα (Γραμμές 14-17).

Στο παράδειγμα 3.4 αν χρησιμοποιηθεί ο Αλγόριθμος `Check_Atoms_Usability` θα συναχθεί ότι η φ_1 εγκλείει κατά **L** τη φ_3 (σε σχέση με το επίπεδο `Month`), ενώ η φ_2 όχι. Ακόμα, είναι ενδιαφέρον να παρατηρήσει κανείς ότι σε σχέση με το επίπεδο `year`, ούτε η φ_1 ούτε η φ_2 εγκλείουν κατά **L** τη φ_3.

3.4 Ισοδύναμοι μετασχηματισμοί για άτομα που εμπλέκουν μόνο επίπεδα

Ακολουθώντας το [Ullm89], έστω η ύπαρξη δύο άπειρων, καθολικά ταξινομημένων πεδίων **L** και **L′** ισομορφικά στους ακέραιους. Έστω επίσης η συνάρτηση f η οποία είναι καθολική και μονότονη πάνω στο **L**, και η οποία απεικονίζει τις τιμές του **L** σε τιμές του **L′**. Η οικογένεια των συναρτήσεων `anc` πληροί αυτά τα κριτήρια.

Υποτίθεται ότι δίνεται ένα σύνολο από ανισότητες της μορφής `A`, `A`, `A`, `f(X)<f(Y)`, `f(X)≤f(Y)`, `f(X)≠f(Y)` και ισότητες της μορφής `f(X)=f(Y)`. Δεν επιτρέπονται ισότητες της μορφής `A`. Αν ένας τέτοιος *υπό-στόχος* (*subgoal*) βρεθεί σε μία ερώτηση, αντικαθίσταται κάθε εμφάνιση του `X` με `Y`. Επίσης αντικαθίσταται κάθε ζεύγος ανισοτήτων `f(X)≤f(Y)` και `f(Y)≤f(X)`, όπου `X`,`Y` είναι διακριτές μεταβλητές με `f(X)=f(Y)`.

Θα χρησιμοποιηθεί το παρακάτω σύνολο αξιωμάτων για αυτές τις ανισότητες:

A1	X≤X	A8	X≤Z, Z≤Y, X≤W, W≤Y και W≠Z συνάγουν X≠Y
A2	X<Y συνάγει X≤Y	A9	X≤Y συνάγει f(X)≤f(Y)
A3	X<Y συνάγει X≠Y	A10	f(X)<f(Y) συνάγει X<Y
A4	X≤Y και X≠Y συνάγουν	A11	f(X)≠f(Y) συνάγει X≠Y
A5	X≠Y συνάγει Y≠X	A12	f(X)≤f(Y) και f(Y)≤f(X) συνάγει =f(Y)
A6	X<Y και Y<Z συνάγουν	A13	f(X)=f(Y) και f(Y)≤f(Z) συνάγει f(X)≤
A7	X≤Y και Y≤Z συνάγουν X	A14	f(X)=f(Y) και f(Y)≠f(Z) συνάγει f(X)≠

A15 f(X)=f(Y) συνάγει f(X)≤f(Y)

Σχήμα 3.6 Αξιώματα για έλεγχο εγκλεισμού κατά L.

Υποτίθεται ότι τα μοντέλα είναι αναθέσεις ακεραίων σε μεταβλητές. Εκφράσεις της μορφής f(X) αντιμετωπίζονται επίσης σαν μεταβλητές. Για τις μεταβλητές της μορφής X επιβάλλονται τα αξιώματα A1 ως A9 και για τις μεταβλητές της μορφής f(X) τα αξιώματα A1 ως A15.

Θεώρημα 3.2: Τα αξιώματα είναι συνεπή και πλήρη.

Για να ελεγχθεί εάν ένα σύνολο ανισοτήτων T συνάγεται από ένα άλλο σύνολο ανισοτήτων S υπολογίζεται το κλείσιμο του S^+ εφαρμόζοντας τα αξιώματα A1-A15 μέχρι να μην παράγουν νέες ανισότητες. Τότε, ελέγχεται εάν το T είναι υποσύνολο του S^+.

3.5 Ελέγχοντας την καταλληλότητα (Usability) των κύβων

Στην υποενότητα αυτή συνδυάζονται τα αποτελέσματα των υποενοτήτων 3.3 και 3.4 για να κατασκευαστεί ένας τρόπος ελέγχου για το πρόβλημα καταλληλότητας των κύβων. Μπορούν να χρησιμοποιηθούν λογικοί μετασχηματισμοί για να μετατραπεί οποιαδήποτε έκφραση σε μια ισοδύναμη έκφραση που αποτελείται από διαζεύξεις συζεύξεων που δεν συμπεριλαμβάνουν ≠ και ¬ [Ende72]. Το Θεώρημα 3.3 παρέχει ικανά κριτήρια για τη δυνατότητα χρήσης ενός κύβου c^{old} στον υπολογισμό ενός κύβου c^{new}. Ο Αλγόριθμος Cube_Usability περιγράφει τα συγκεκριμένα βήματα που απαιτούνται για τον υπολογισμό αυτό.

Θεώρημα 3.3: Έστω ένα λεπτομερές σύνολο δεδομένων $DS^0=[L_1^0,\ldots,L_n^0,M^0]$ και δύο κύβους $c^{old}=(DS^0,\varphi_{old},[L_1^{old},\ldots,L_n^{old},M_{old}],agg_{old}(M^0))$ και $c^{new}=(DS^0,\varphi_{new},[L_1^{new},\ldots,L_n^{new}$

$, M_{new}], agg_{new}(M^0))$. Εάν $agg_{old}=agg_{new}$, $L_i^{old} \prec L_i^{new}$, $1 \le i \le n$, και μία από τις παρακάτω περιπτώσεις ισχύει για τα φ_{old} και φ_{new}:

- φ_{old} και φ_{new} περιέχουν συζεύξεις ατόμων μόνο της μορφής $L_i \theta L_j$, όλα τα επίπεδα L_i, L_j είναι υψηλότερα από τα αντίστοιχα επίπεδα του σχήματος του c^{old} (δηλαδή $L_{i,j}^{old} \prec L_{i,j}$) και τέλος, το φ_{old} ανήκει στο κλείσιμο του φ_{new}, ή,
- φ_{old} και φ_{new} αφορούν συζεύξεις ατόμων της μορφής $L\theta l$ και $\varphi_{new} \subseteq_{[L_1^{new}, \ldots, L_n^{new}]} \varphi_{old}$,

τότε ο Αλγόριθμος Cube_Usability υπολογίζει σωστά c^{new} από τις πλειάδες του c^{old}.

Algorithm Cube_Usability.

Input: A detailed data set $DS^0=[L_1^0, \ldots, L_n^0, M^0]$ και two cubes $c^{old}=(DS^0, \varphi_{old}, [L_1^{old}, \ldots, L_n^{old}, M_{old}], agg_{old}(M^0))$ και $c^{new}=(DS^0, \varphi_{new}, [L_1^{new}, \ldots, L_n^{new}, M_{new}], agg_{new}(M^0))$ such that φ_{old} και φ_{new} involve either (a) conjunctions of atoms of the form $L\theta l$ or (b) conjunctions of atoms of the form $L\theta L'$ where L και L' are levels και l is a value.

Output: A rewriting that calculates cube c^{new} from the tuples of c^{old}.

1. If all atoms of φ_{old} και φ_{new} involve conjunctions of atoms of the form $L\theta l$ Then
2. For every atom $a=anc_{L^0}^{L}(L^0)\,\theta l$ in φ_{new} (or equivalent to this form)
3. If L^{old} is the respective level in the schema of c^{old} και $L^{old} \prec L$ Then
4. Transform a to $anc_{L^{old}}^{L}(L^{old})\,\theta l$
5. EndIf
6. ElseIf L^{old} is the respective level in the schema of c^{old} και $L \prec L^{old}$ Then
7. Transform a to $L^{old}\theta' anc_{L}^{L^{old}}(l)$ where $\theta'=\theta$ except for two cases:
   (a) $a=anc_{L^0}^{L}(L^0)<l$ και $l \neq \min(desc_{L}^{L^{old}}(anc_{L}^{L^{old}}(l)))$ where $\theta' = \le$,
   (b) $a=anc_{L^0}^{L}(L^0)>l$ και $l \neq \max(desc_{L}^{L^{old}}(anc_{L}^{L^{old}}(l)))$ where $\theta' = \ge$
8. EndIf
9. EndFor
10. EndIf
11. If all atoms of φ_{old} και φ_{new} involve conjunctions of atoms of the form $a=anc_{L^0}^{L}(L^0)\,\theta anc_{L^{0'}}^{L'}(L^{0'})$ (or equivalent to this form), where both L και L' are higher than the respective levels of c^{old} Then
12. For every atom $a=anc_{L^0}^{L}(L^0)\,\theta anc_{L^{0'}}^{L'}(L^{0'})$ in φ_{new}
15. Transform a to $anc_{L^{old}}^{L}(L^{old})\,\theta anc_{L^{old'}}^{L'}(L^{old'})$
16. EndFor
17. EndIf
18. Apply the transformed selection condition to c^{old} και derive a new data set DS^1.
19. Replace all the values of DS^1 with their ancestor values at the levels of c^{new}, resulting in a new data set DS^2.
20. Aggregate ("group by" in the relational semantics) on the tuples of DS^2, so that we produce c^{new}.

Σχήμα 3.7 Αλγόριθμος Cube_Usability

Το Θεώρημα 3.3 ελέγχει για πιθανή καταλληλότητα ζεύγη κύβων που έχουν συζευκτικές συνθήκες επιλογής που δεν περιέχουν ≠ και ¬. Κύβοι που περιλαμβάνουν διαζευκτικές συνθήκες επιλογής μπορούν να αντιμετωπισθούν με το γνωστό τρόπο [Ullm89].

Να σημειωθεί επίσης ότι το αντίστροφο του θεωρήματος (‘και μόνο εάν’) δεν ισχύει. Έστω η περίπτωση μιας συγκεκριμένης διάστασης D, που περιλαμβάνει δύο επίπεδα low και high, όπου η σχέση desc είναι συνάρτηση (που σημαίνει ότι η συνάρτηση anc_{low}^{high} έχει αντίστροφη και η αντιστοιχήσει από λεπτομερείς σε υψηλότερου επιπέδου τιμές είναι 1:1). Τότε, αν και η συνθήκη (2) του Θεωρήματος 3.3 παραβιάζεται, ένας κύβος στο επίπεδο high μπορεί να χρησιμοποιηθεί και για τον υπολογισμό ενός κύβου στο επίπεδο low. Ακόμα, είναι εύκολο να κατασκευαστεί ένα παράδειγμα που να δείχνει ότι οι παραπάνω τεχνικές δεν εφαρμόζονται στην κλάση ερωτήσεων που περιέχουν άτομα και των δύο κατηγοριών του Θεωρήματος 3.3

Παράδειγμα 3.4 Έστω c^{new} και c^{old} δύο κύβοι πάνω στο DS^0 του σχήματος 16, ορισμένοι ως εξής:

$c^{old}=(DS^0,\varphi_{old},$[Month,Country,Type,Salesman,Sum_old],sum(Sales)) και

$c^{new}=(DS^0,\varphi_{new},$[Month,Country,Categoty,Salesman,Sum_new],sum(Sales))

where φ_{old}=18-Feb-97≤day ∧ day≤3-Sep-97 ∧ $anc_{Item}^{Category}$(Item)= "Books" και φ_{new}=1-Mar-97≤day ∧ day≤3-Sep-97 ∧ "Literature"≤anc_{Item}^{Type}(Item) ∧ anc_{Item}^{Type}(Item)≤ "Philosophy".

Για να ελεγχθεί αν ο c^{new} μπορεί να υπολογισθεί από τον c^{old} εφαρμόζεται το Θεώρημα 3.3. Τα σχήματα και οι αθροιστικές συναρτήσεις των δύο κύβων είναι συμβατά (συνθήκες (1) και (2) του Θεωρήματος 3.3). Επιπλέον, η φ_{new} είναι **L**-επικαλυπτόμενη από τη φ_{old} σε σχέση με τα επίπεδα του c^{new}. Ακολουθώντας τις γραμμές 2-10 του Αλγόριθμου Cube_Usability, μετασχηματίζουμε τη φ_{new} ώστε να είναι εφαρμόσιμη στο σχήμα του κύβου c^{old}. Οι μετασχηματισμοί των γραμμών 3-8 καταλήγουν στην

$\varphi_{n@o}$=Mar-97≤Month ∧ Month≤Sep-97∧"Literature"≤Type ∧ Type≤"Philosophy".

Εφαρμόζεται η μετασχηματισμένη συνθήκη επιλογής στον c^{old} (όπως φαίνεται στο σχήμα 3.15a) και παράγεται ένα νέο σύνολο δεδομένων DS^1 (όπως φαίνεται στο

σχήμα 3.15b). Έπειτα αντικαθίστανται όλες οι τιμές του DS^1 με τις αντίστοιχές του στα επίπεδα του c^{new} (Γραμμή 19), καταλήγοντας σε ένα νέο σύνολο δεδομένων DS^2 (όπως φαίνεται στο σχήμα 3.15c). Τέλος, συναθροίζονται οι πλειάδες του DS^2 και παράγεται το c^{new} (όπως φαίνεται στο σχήμα 3.15d). ¦

Month	Type	Salesman	Country	Sum_old
Feb-97	Literature	Netz	USA	5
Sep-97	Philosophy	Netz	Japan	50
Sep-97	Literature	Netz	Japan	30

(a)

Month	Type	Salesman	Country	Sum_1
Sep-97	Philosophy	Netz	Japan	50
Sep-97	Literature	Netz	Japan	30

(b)

Month	Category	Salesman	Country	Sum_2
Sep-97	Book	Netz	Japan	50
Sep-97	Book	Netz	Japan	30

(c)

Month	Category	Salesman	Country	Sum_new
Sep-97	Book	Netz	Japan	80

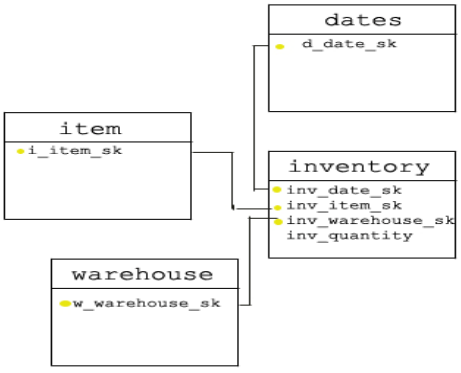
(d)

Σχήμα 3.8 Υπολογίζοντας το c^{new} από το c^{old}.

ΚΕΦΑΛΑΙΟ 4. ΠΕΙΡΑΜΑΤΙΚΗ ΜΕΛΕΤΗ

4.1 Δεδομένα και σχήμα βάσης δεδομένων

Στο κεφάλαιο αυτό παρουσιάζεται η πειραματική μελέτη και οι μετρήσεις που έγιναν για την εύρεση μεταβατικής κλειστότητας, για το χρόνο εκτέλεσης του αλγορίθμου καταλληλότητας κύβου και για το χρόνο απάντησης οποιουδήποτε κύβου από κάποιον άλλον που τον απαντά. Τα πειράματα εκτελέστηκαν σε λειτουργικό σύστημα Windows Vista 32-bit και σε υπολογιστή με επεξεργαστή Intel Core Duo 2GH και μνήμη 2GB. Για τη μελέτη των αλγορίθμων αυτών χρησιμοποιήθηκαν τα δεδομένα του μέτρου επίδοσης TPC –DS.



Σχήμα 4.1 Αρχικό σχεσιακό σχήμα αστέρα

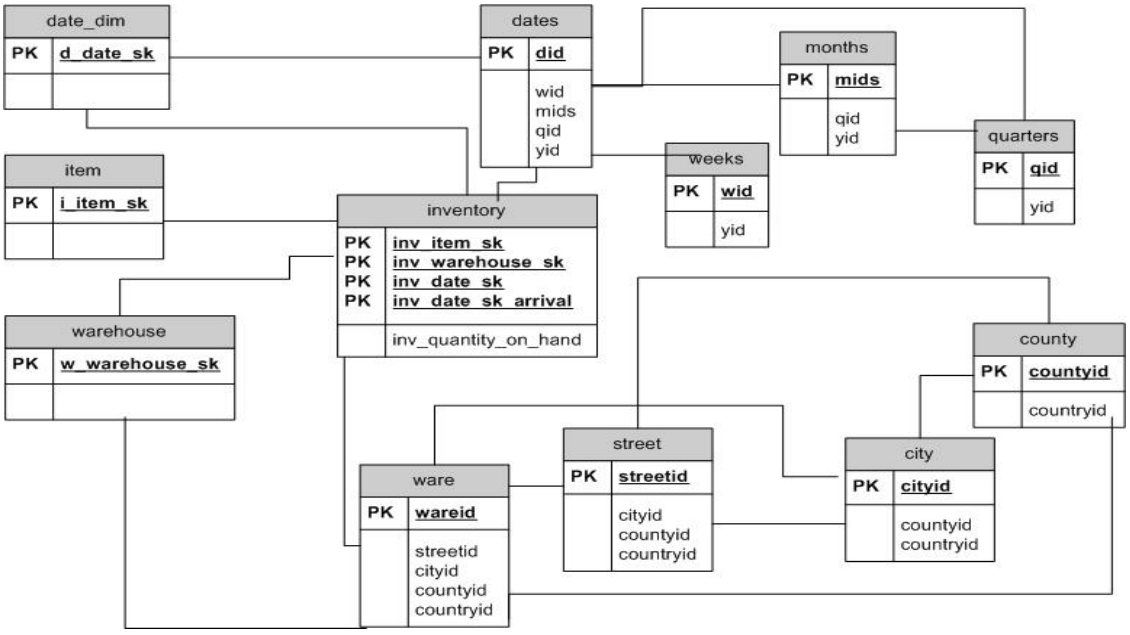
Το TPC-DS έχει εφτά πίνακες πληροφοριών, από τους οποίους διαλέξαμε τον `inventory`, ο οποίος σχετίζεται με σχέση ξένου κλειδιού με τρείς πίνακες διάστασης,

τον `warehouse`, τον `dates` και τον `item`. Αρχικά το σχήμα της βάσης ακολουθούσε το σχήμα αστέρα και η σχέση κλειδιού –ξένου κλειδιού μεταξύ των πινάκων ήταν όπως παρουσιάζεται στο παραπάνω σχήμα.

Βλέπουμε ότι ο πίνακας `inventory` έχει ένα σύνθετο πρωτεύον κλειδί που αποτελείται από τρία πεδία, τα οποία συνδέονται με σχέση ξένου κλειδιού με τα πρωτεύοντα κλειδιά των άλλων τριών πινάκων. Επίσης έχει το πεδίο `inv_quantity_on_hand`, το οποίο είναι αυτό στο οποίο θα εφαρμόσουμε τις συναρτήσεις συνάθροισης,

Στη συνέχεια στον πίνακα `inventory` προσθέσαμε το πεδίο `inv_date_sk_arrival`, το οποίο είναι επίσης ξένο κλειδί στον πίνακα `warehouse` και αποτελεί κι αυτό μέρος του σύνθετου πρωτεύοντος κλειδιού. Ο λόγος για τον οποίο προστέθηκε η κολώνα αυτή, είναι για να μπορέσουμε να εφαρμόσουμε ερωτήματα με συνθήκη επιλογής τύπου `Level θ Level`.

Έπειτα προσθέσαμε τους πίνακες που παρουσιάζονται στο παρακάτω σχήμα για να επιταχύνουμε τις συζεύξεις των πινάκων μεταξύ τους. Το σχήμα της βάσης πλέον ακολουθεί το πρότυπο του σχήματος νιφάδας.



Σχήμα 4.2 Σχεσιακό σχήμα νιφάδας του τελικού σχήματος

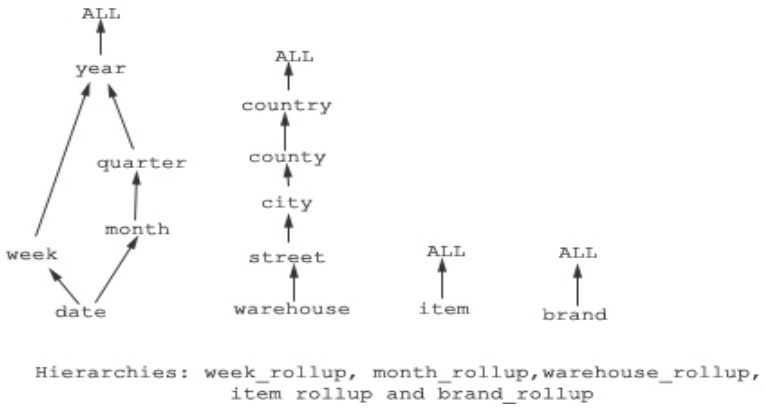
4.2 Διαστάσεις και ιεραρχίες του σχήματος

4.2.1 Ιεραρχίες του γενικού σχήματος

Από τους παραπάνω πίνακες του σχήματος νιφάδας προκύπτουν κάποιες διαστάσεις και κάποιες ιεραρχίες. Οι διαστάσεις που δημιουργούνται είναι η `date`, η `warehouse` και η `item`. Ενδεικτικά παρατίθεται η δήλωση μιας διάστασης με τη δηλωτική γλώσσα που χρησιμοποιήσαμε.

```
CREATE DIMENSION warehouse_dim
LEVEL warehouse       IS warehouse.w_warehouse_sk
LEVEL street       IS warehouse.hier_street
LEVEL city      IS warehouse.hier_city
LEVEL county   IS warehouse.hier_county
LEVEL country    IS warehouse.hier_country
```

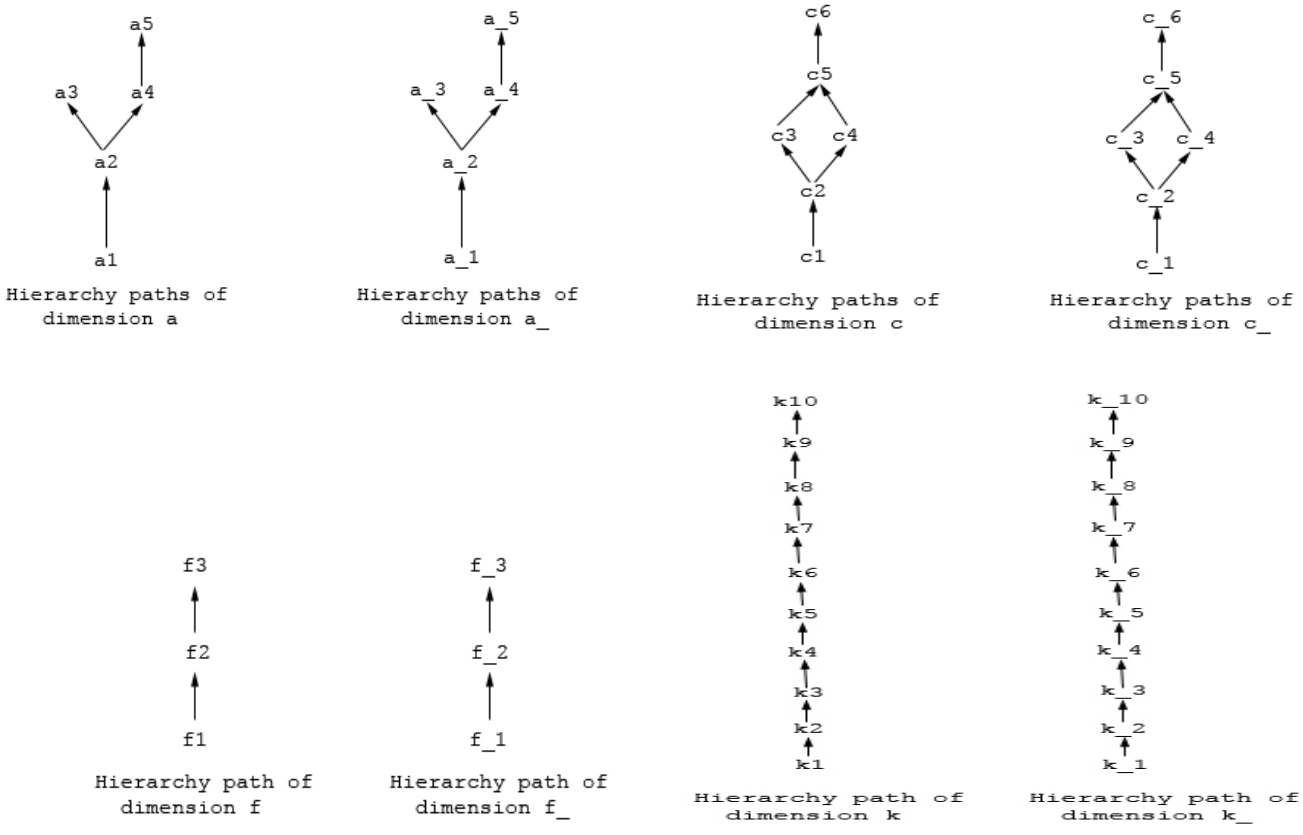
Από τις διαστάσεις αυτές προκύπτουν και οι ιεραρχίες `month_rollup`, `week_rollup`, `warehouse_rollup`, `item_rollup`, `brand_rollup`. Συγκεκριμένα από την date διάσταση παράγονται δύο ιεραρχικά μονοπάτια.



Σχήμα 4.3 Οι ιεραρχίες του σχήματος

4.2.2 Ιεραρχίες για την εκτέλεση πειραμάτων με κύβους με συνθήκη επιλογής τύπου Level θ Level

Για τη μελέτη εύρεσης της μεταβατικής κλειστότητας δημιουργήσαμε διαστάσεις με 3,5 και 10 επίπεδα. Από αυτές προέκυψαν ιεραρχίες ύψους 3,4,5 και δέκα. Για να μπορέσουμε να δημιουργήσουμε τα άτομα τύπου LevelθLevel δημιουργήσαμε και τις αντίστοιχες ομόλογες ιεραρχίες. Η ιεραρχίες έχουν ονόματα από `a` έως `t` και οι αντίστοιχες ομόλογές τους συμβολίζονται με `a_` έως `t_`. Συγκεκριμένα οι ιεραρχίες από a έως e παράγουν δύο μονοπάτια ιεραρχίας. Οι ιεραρχίες `f` έως `j` έχουν τρία επίπεδα και παράγουν μόνο ένα μονοπάτι ιεραρχία. Οι `a` και η `b` είναι ίδιες, οι `c-e` επίσης, καθώς και οι `f - j`. Η k είναι ίδια με τις υπόλοιπες ιεραρχίες από l έως `t`. Ευνόητο είναι ότι το ίδιο σχήμα ακολουθούν και οι ομόλογές τους. Παρακάτω παρουσιάζονται οι ιεραρχίες αυτές.



Σχήμα 4.4 Ιεραρχίες για την εκτέλεση πειραμάτων για εύρεση μεταβατικής κλειστότητας

4.3 Μελέτη χρόνου έυρεσης μεταβατικής κλειστότητας

Για τον υπολογισμό της μεταβατικής κλειστότητας έγιναν πειράματα δημιουργώντας ιεραρχίες ύψους 3-5 και ιεραρχίες ύψους 10. Οι πρώτες 5 διαστάσεις (`A-E`), οι οποίες είναι ύψους 3-5 παράγουν δύο ιεραρχίες ενώ οι επόμενες 5 (`F-J`) παράγουν από μία ιεραρχία. Επίσης δημιουργήσαμε και τις ομόλογές τους, έτσι ώστε να μπορούμε να δημιουργήσουμε ερωτήματα με συνθήκες επιλογής του τύπου `Level θ Level`. Τα πειράματα έγιναν με 2, 5 και 10 άτομα στη συνθήκη επιλογής. Κάθε πείραμα το εκτελέσαμε από 3 φορές. Στον πίνακα που ακολουθεί παρουσιάζονται οι συνθήκες επιλογής, οι οποίες συμμετείχαν στα πειράματα και για τη διευκόλυνση εξαγωγής συμπερασμάτων παρατίθενται και το πλήθος των ατόμων που παράγονται μετά τον υπολογισμό της μεταβατικής κλειστότητας. Να υπενθυμίσουμε, ότι το πιο χαμηλό επίπεδο είναι το επίπεδο με αριθμό ένα και καθώς αυξάνεται ο αριθμός του επιπέδου αυξάνεται και το ύψος του στην ιεραρχία. Όσο πιο χαμηλά βρισκόμαστε στην ιεραρχία τόσο περισσότερους προγόνους έχει το άτομο αυτό.

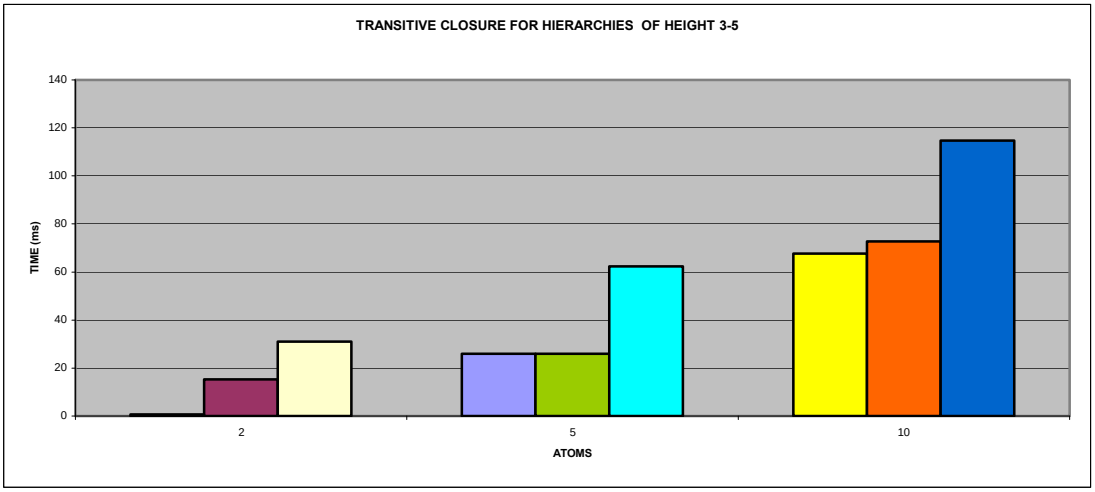
Πίνακας 4.1 Συνθήκες επιλογής για ιεραρχίες ύψους 3-5

	Συνθήκες Επιλογής	**# ατόμων**
2 άτομα	**1)** a3<a_3,b3<b_3	8
	2) a1<a_1,b3!=b_3	10
	3) a1<a_1,b1<b_1	16
5 άτομα	**4)** f1<f_1,g1<g_1,h1<h_1,i1<i_1,j1<j_1	28
	5) a1<a_1,b1<b_1,c1!=c_1,i1!=i_1,j1!=j1	22
	6) a1<a_1,b1<b_1,c1<c_1,d1<d_1,e1<e_1	37
10 άτομα	**7)** a2<a_2,b2<b_2,c2<c_2,d2<d_2,e2<e_2,f2<f_2,g2<g_2,h2<h_2,i2<i_2,,j2<j_2	59
	8) a1<a_1,b1<b_1,c1<c_1,d1<d_1,e1<e_1,f1!=f_1,g1!=g_1,h1!=h_1,i1!=i_1,j1!=j_1	50
	9) a1<a_1,b1<b_1,c1<c_1,d1<d_1,e1<e_1,f1<f_1,g1<g_1,h1<h_1,i1<i_1,j1<j_1	68

Πίνακας 4.2 Συνθήκες επιλογής για ιεραρχίες ύψους 10

	Συνθήκες επιλογής	**# ατόμων**
2 άτομα	1) k5<k_5,l5<l_5	18
	2) k1<k_1,l1!=l_1	15
	3) k1<k1, l1<l_1	26
5 άτομα	4) k5<k_5,l5<l_5,m5!=m_5,n5!=n_5,o5!=o_5	24
	5) k1<k_1,l1<l_1,m1!=m_1,n1!=n_1,o1!=o_1	32
	6) k1<k_1,l1<l_1,m1<m_1,n1<n1,o1<o1	65
10 άτομα	7) k5<k_5,l5<l_5,m5<m_5,n5<n_5,o5<o_5,p5!=p_5,q5!=q_5,r5!=r_5,s5!=s_5,t5!=t_5	55
	8) k5<k_5,l5<l_5,m5<m_5,n5<n_5,o5<o_5,p5<p_5,q5<q_5,r5<r_5,s5<s_5,t5<t_5	90
	9) k1<k_1,l1<l_1,m1<m_1,n1<n_1,o1<o_1,p1<p_1,q1<q_1,r1<r_1,s1<s_1,t1<t_1	130

Παρακάτω παρατίθεται η γραφική παράσταση υπολογισμού μεταβατικής κλειστότητας για συνθήκες επιλογής με άτομα που προκύπτουν από ιεραρχίες ύψους τρία έως πέντε. Η κάθε ράβδος αντιπροσωπεύει το κάθε πείραμα, του οποίου υπολογίστηκε ο μέσος χρόνος εκτέλεσής του. Ο άξονας x χωρίζεται σε τρία τμήματα εκ των οποίων το πρώτο περιέχει τα πειράματα που έγιναν με συνθήκες επιλογής με δύο αριθμό ατόμων, το δεύτερο τμήμα περιέχει τα πειράματα με πέντε αριθμό ατόμων και το τελευταίο τμήμα τα πειράματα με δέκα αριθμό ατόμων. Στον άξονα y είναι ο χρόνος μετρημένος σε χιλιοστά του δευτερολέπτου.



Σχήμα 4.5Υπολογισμός της μεταβατικής κλειστότητας σε ιεραρχίες ύψους 3-5

- Η μεταβατική κλειστότητα της πρώτης συνθήκης επιλογής στην περίπτωση των δύο ατόμων βλέπουμε ότι υπολογίζεται πολύ γρήγορα. Αυτό οφείλεται στο ότι και τα δύο άτομα είναι στο υψηλότερο σημείο της ιεραρχίας ορισμένα με αποτέλεσμα να μην έχουν προγόνους και όλα τα υπόλοιπα άτομα της κλειστότητας να υπολογίζονται πολύ γρήγορα μιας και ο αλγόριθμος στο μόνο σημείο που καθυστερεί είναι στο σημείο που αναζητά τους προγόνους.
- Η δεύτερη ράβδος αντιπροσωπεύει το πείραμα το οποίο περιέχει δύο άτομα εκ των οποίων το ένα είναι ορισμένο στο υψηλότερο σημείο της ιεραρχίας ενώ το άλλο άτομο στο χαμηλότερο. Βλέπουμε ότι παρόλο που έχουν μόνο δύο άτομα διαφορά σε πλήθος, η διαφορά στο χρόνο είναι σημαντική. Αυτό οφείλεται στο γεγονός, ότι στο δεύτερο πείραμα τα άτομα παράγονται από το Αξίωμα 9, σύμφωνα με το οποίο

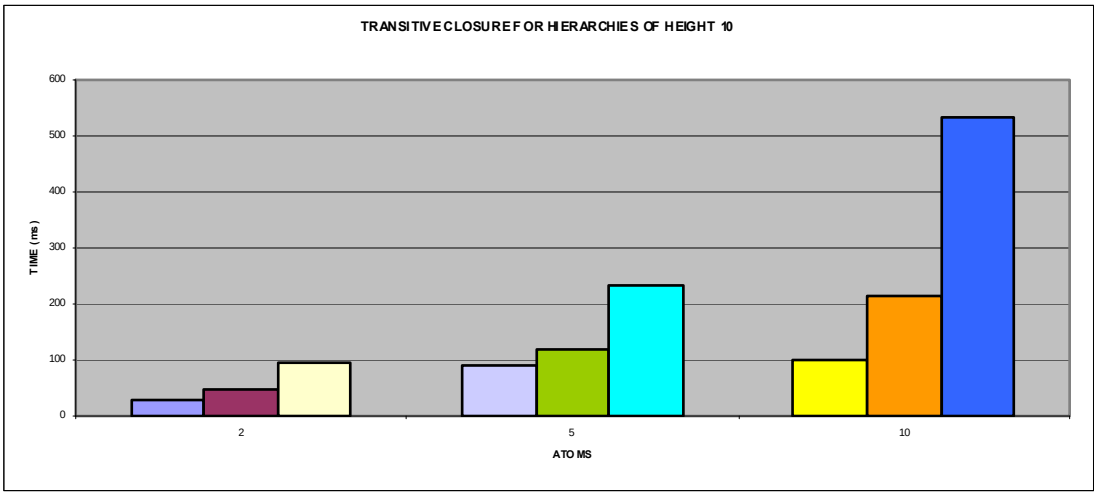
αναζητάμε για προγόνους. Η συνάρτηση αυτή προσπελαύνει όλη την ιεραρχία για να βρει τους προγόνους του κάθε επιπέδου και είναι λογικό το αξίωμα αυτό να καθυστερεί περισσότερο σε σχέση με τα υπόλοιπα αξιώματα, τα οποία παράγουν κατευθείαν αυτόματα τα κατάλληλα άτομα χωρίς να χρειαστεί να ανατρέξουν σε ιεραρχίες.

- Η τρίτη ράβδος αφορά την εκτέλεση του αλγορίθμου εύρεσης μεταβατικής κλειστότητας για συνθήκη επιλογής που περιέχει άτομα τα οποία είναι και τα δύο ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας. Είναι το πείραμα που στην περίπτωση των δύο ατόμων στην συνθήκη επιλογής καθυστερεί περισσότερο και όπως προαναφέραμε αυτό οφείλεται στην εφαρμογή του Αξιώματος 9.
- Η τέταρτη ράβδος αφορά την περίπτωση με τα πέντε άτομα εκ των οποίων είναι και τα πέντε ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας.
- Η πέμπτη ράβδος αντιπροσωπεύει την περίπτωση που δύο άτομα είναι ορισμένα στο πιο χαμηλό επίπεδο διαστάσεων που παράγουν δύο ιεραρχίες. Τα υπόλοιπα τρία άτομα δεν μας ενδιαφέρει σε τι ύψους επίπεδο είναι ορισμένα, γιατί περιέχουν τον τελεστή «!=» κι έτσι δεν παράγονται νέα άτομα με προγόνους των επιπέδων των ατόμων αυτών. Παρόλ' αυτά παρατηρούμε ότι η εκτέλεση του αλγορίθμου στην περίπτωση αυτή εκτελείται σχεδόν στον ίδιο χρόνο με την τέταρτη περίπτωση και συμπεραίνουμε, ότι δημιουργεί μεγάλη καθυστέρηση το γεγονός από μία διάσταση να προκύπτουν παραπάνω από μία ιεραρχίες. Επίσης βλέπουμε ότι το πλήθος των ατόμων της πέμπτης ράβδου είναι μεγαλύτερο από ό,τι της τέταρτης.
- Η έκτη ράβδος αφορά την περίπτωση που πέντε άτομα είναι ορισμένα στο χαμηλότερο επίπεδο διαστάσεων, που παράγουν δύο ιεραρχίες και το αποτέλεσμα είναι το αναμενόμενο, δηλαδή να καθυστερεί περισσότερο και από την τέταρτη περίπτωση που ναι μεν τα άτομα είναι ορισμένα στο χαμηλότερο επίπεδο, αλλά από τη διάσταση που αντιστοιχούν προκύπτει μόνο μία ιεραρχία, άρα και μικρότερο πλήθος ατόμων, άρα και λιγότερες διασχίσεις ιεραρχιών και από την Πέμπτη

περίπτωση στην οποία υπάρχουν μόνο δύο άτομα με επίπεδα που ανήκουν σε παραπάνω από μία ιεραρχίες, άρα περισσότερα άτομα και περισσότερες διασχίσεις ιεραρχιών.

- Η έβδομη ράβδος αφορά τον υπολογισμό της μεταβατικής κλειστότητας της συνθήκης επιλογής που έχει δέκα άτομα ορισμένα σε μεσαίο επίπεδο ιεραρχίας και πέντε από αυτά, αφορούν επίπεδα που ανήκουν σε δύο ιεραρχίες.
- Στην όγδοη ράβδο η εκτέλεση του αλγορίθμου καθυστερεί λίγο περισσότερο από την προηγούμενη περίπτωση. Η συνθήκη επιλογής έχει πέντε άτομα ορισμένα στο χαμηλότερο επίπεδο και άλλα πέντε άτομα ορισμένα σε αδιάφορο επίπεδο μιας κι αυτά περιέχουν τον τελεστή «!=» κι έτσι δεν ελέγχονται οι πρόγονοι των επιπέδων των ατόμων αυτών.
- Στην τελευταία ράβδο και τα δέκα άτομα είναι ορισμένα στο χαμηλότερο επίπεδο κι έχουν εννιά προγόνους με αποτέλεσμα να είναι το πιο αργό πείραμα απ'όλα.

Στη συνέχεια θα μελετηθεί το διάγραμμα απεικόνισης των χρόνων υπολογισμού της μεταβατικής κλειστότητας για ιεραρχίες ύψους δέκα. Σημειώνεται ξανά ότι το χαμηλότερο επίπεδο ιεραρχίας είναι το ένα και το υψηλότερο είναι το δέκα. Τα πειράματα που έγιναν είναι αυτά που αναγράφονται στον Πίνακα 4.2.



Σχήμα 4.6 Υπολογισμός την μεταβατικής κλειστότητας σε ιεραρχίες ύψους 10

- Η πρώτη μέτρηση που έγινε έχει δύο άτομα, τα οποία είναι και τα δύο ορισμένα στη μέση της ιεραρχίας.
- Η δεύτερη ράβδος αφορά την περίπτωση συνθήκης επιλογής με ένα άτομο, το οποίο είναι ορισμένο στο χαμηλότερο επίπεδο της ιεραρχίας και ένα άτομο το οποίο δεν μας ενδιαφέρει σε ποιό επίπεδο είναι ορισμένο γιατί περιέχει τον τελεστή «!=». Βλέπουμε ότι αυτό το πείραμα παρόλο που βρίσκει προγόνους από μόνο ένα άτομο (εννιά προγόνους) καθυστερεί περισσότερο από το προηγούμενο πείραμα το οποίο υπολόγιζε προγόνους από δύο άτομα τα οποία ήταν ορισμένα στη μέση της ιεραρχίας (συνολικά 4+4 προγόνους). Ενδιαφέρον παρουσιάζει το γεγονός, ότι στη δεύτερη μέτρηση παρόλο που παράγονται λιγότερα άτομα από ό,τι στην πρώτη (15 vs 18) ο χρόνος εκτέλεσης είναι μεγαλύτερος.
- Στην τρίτη ράβδο φαίνεται ο χρόνος που χρειάζεται ο αλγόριθμος για να υπολογίσει τη μεταβατική κλειστότητα συνθήκης επιλογής, η οποία περιλαμβάνει δύο άτομα ορισμένα και τα δύο στο χαμηλότερο επίπεδο της ιεραρχίας. Ο χρόνος εκτέλεσής του σε σχέση με τα υπόλοιπα είναι ο αναμενόμενος.
- Στην τέταρτη ράβδο αναπαρίσταται η μέτρηση στην οποία η συνθήκη επιλογής περιλαμβάνει δύο άτομα ορισμένα στο μεσαίο επίπεδο της ιεραρχίας και τρία άτομα τα οποία περιέχουν τον τελεστή «!=». Εδώ αν συγκρίνουμε με την πρώτη ράβδο που αφορά την περίπτωση δύο ατόμων ορισμένα στη μέση της ιεραρχίας, φαίνεται ο χρόνος που χρειάζεται για τον υπολογισμό νέων ατόμων από άτομα που περιέχουν τον τελεστή «!=» .
- Η πέμπτη ράβδος αφορά την περίπτωση που η συνθήκη επιλογής περιλαμβάνει δύο άτομα ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας και τα υπόλοιπα να συμπεριλαμβάνουν τον τελεστή «!=». Όπως περιμέναμε η πέμπτη μέτρηση έχει μεγαλύτερο χρόνο εκτέλεσης αφού τα δύο άτομά της είναι ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας.

- Η έκτο ράβδος αφορά περίπτωση συνθήκης επιλογής και με τα πέντε άτομα ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας και όπως ήταν αναμενόμενο ο χρόνος είναι πολύ μεγαλύτερος από τους προηγούμενους.
- Στην έβδομη μέτρηση τα πέντε άτομα είναι ορισμένα σε μεσαίο επίπεδο ιεραρχίας ενώ τα υπόλοιπα πέντε έχουν τον τελεστή «!=».
- Στην όγδοη μέτρηση όλα τα άτομα είναι ορισμένα στο μεσαίο επίπεδο της ιεραρχίας και βλέπουμε ότι ο χρόνος αυξάνεται αρκετά σε σχέση με την προηγούμενη περίπτωση.
- Στην ένατη μέτρηση και τα δέκα άτομα είναι ορισμένα στο χαμηλότερο επίπεδο της ιεραρχίας και ο χρόνος που χρειάζεται είναι μεγαλύτερος από όλους.

Εν τέλει καταλήγουμε στο συμπέρασμα, ότι οι παράγοντες που παίζουν ρόλο στη διαμόρφωση του χρόνου για τον υπολογισμό της μεταβατικής κλειστότητας είναι πρώτον σε τι ύψος στην ιεραρχία είναι ορισμένα τα άτομα της συνθήκης επιλογής και κατά συνέπεια τι ύψους είναι η ιεραρχία και δεύτερον το αν οι διαστάσεις των ατόμων παράγουν παραπάνω από μία ιεραρχίες. Επίσης παρατηρήσαμε ότι ο χρόνος υπολογισμού νέων ατόμων από τα άτομα που περιέχουν τον τελεστή «!=» είναι σταθερός. Επίσης το πλήθος των ατόμων που παράγονται κατά τον υπολογισμό της μεταβατικής κλειστότητας δεν παίζουν ρόλο όταν πρόκειται να συγκρίνουμε περιπτώσεις στις οποίες τα άτομα είναι ορισμένα σε διαφορετικό ύψος ιεραρχίας και παράγουν άτομα από το αξίωμα 9.

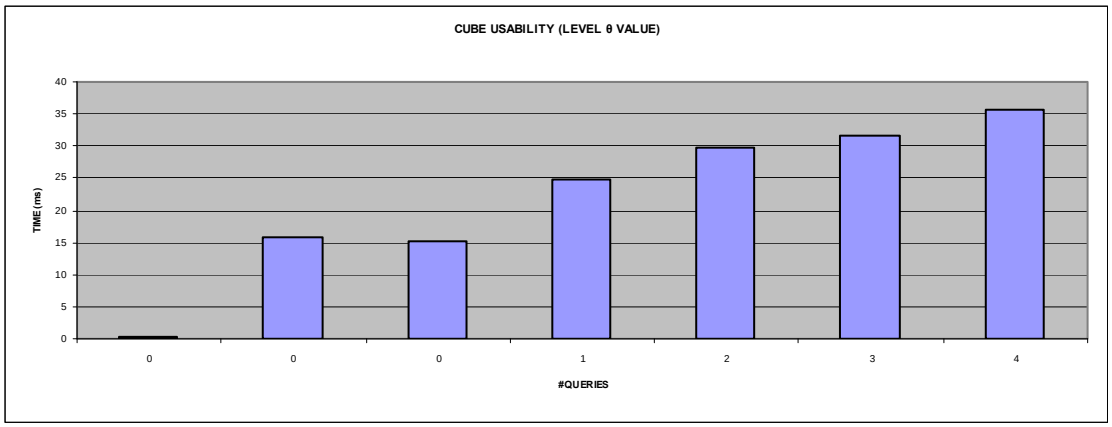
4.4 Έλεγχος καταλληλότητας κύβου για την περίπτωση Level θ Value

Έγιναν πειράματα για τη μελέτη του χρόνου που χρειάζεται για τον υπολογισμό της καταλληλότητας κύβου, δηλαδή για το αν ένας κύβος μπορεί να απαντήσει κάποιον άλλον. Στον παρακάτω πίνακα παρουσιάζονται τα πειράματα που έγιναν για τον σκοπό αυτό.

Πίνακας 4.3 Κύβοι για μελέτη χρόνου εκτέλεσης αλγορίθμου καταλληλότητας κύβων

1	c1=([date>=2415040,date<=2415057],[month,country,brand],sum)
	c2=([date>=2415040],[date,warehouse,brand],sum)
2	c3=([date>=2415040,date<=2415057],[date,warehouse,brand],sum)
	c4=([date>=2415040],[date,warehouse,brand],sum)
3	c5=([date>=2415040],[date,warehouse,brand],sum)
	c6=([date>=2415040,date<=2415057],[date,warehouse,brand],sum)
4	c7=([date>=2415040],[date,warehouse,brand],sum)
	c8=([date>=2415041],[year,country,brand],sum)
5	c9=([date>=2415040,date<=2415800],[date,warehouse,brand],sum)
	c10=([date>=2415386,date<=2415750],[year,country,brand],sum)
6	c11=([date>=2415040,date<=2415800,warehouse>=5],[date,warehouse,brand],sum)
	c12=([date>=2415386,date<=2415750,warehouse>=6],[year,county,brand],sum)
7	c13=([date>=2415040,date<=2415755,warehouse>=5,warehouse<=9],[date,warehouse,brand],sum)
	c14=([date>=2415386,date<=2415750,warehouse>=6,warehouse<=50][year,county,brand],sum)

Παρακάτω παρουσιάζεται η γραφική παράσταση για τον υπολογισμό καταλληλότητας κύβου. Στον άξονα x είναι το πλήθος των ερωτήσεων που χρειάστηκε να γίνουν στη βάση για το αν κάποιο άτομο ενός κύβου, περιλαμβάνεται στη συνθήκη επιλογής του
κύβου, ο οποίος καλείται να τον απαντήσει. Στον άξονα y είναι ο χρόνος που χρειάστηκε για τον υπολογισμό αυτό μετρημένος σε χιλιοστά του δευτερολέπτου.



Σχήμα 4.7 Χρόνος αλγορίθμου καταλληλότητας κύβων

- Η πρώτη μέτρηση αφορά το αν μπορεί ο κύβος `c1` να απαντήσει τον κύβο `c2`. Βλέπουμε ότι ο κύβος `c1` είναι πιο ψηλά ορισμένος απο τον `c2` και ο αλγόριθμος κατευθείαν σταματά κι επιστρέφει `false` χωρίς να χρειαστεί να ελέγξει τις υπόλοιπες συνθήκες γι αυτό εκτελείται και σε τόσο σύντομο χρόνο.
- Η δεύτερη ράβδος μας δείχνει πόσο χρόνο χρειάζεται για να ελέγξουμε αν μπορεί ο κύβος `c3` να απαντήσει τον κύβο `c4`. Ο `c3` έχει περισσότερα άτομα στην συνθήκη επιλογής από τον `c4` άρα δεν τον απαντάει. Ο λόγος για τον οποίο καθυστερεί περισσότερο από πριν είναι επειδή εκτελεί τη συνάρτηση η οποία μετατρέπει τα άτομα στο χαμηλότερο επίπεδο όπως επίσης παράγει ένα άτομο για κάθε διάσταση. Στην τρίτη περίπτωση ο κύβος `c5` απαντά τον `c6`. Αρχικά εκτελείται η συνάρτηση μετατροπής στο χαμηλότερο επίπεδο της κάθε ιεραρχίας, όπως στην προηγούμενη περίπτωση και στη συνέχεια δεν χρειάζεται να γίνει καμία ερώτηση στη βάση γιατί ο `c6` έχει για το ίδιο επίπεδο (`date`) και για τον ίδιο τελεστή (>=) ακριβώς την ίδια σταθερά στο άτομο, όπως επίσης ο `c5` δεν έχει κάποιον περιορισμό αντίστοιχο με το δεύτερο άτομο του `c6`.
- Στην τέταρτη εκτέλεση του αλγορίθμου ο κύβος `c7` δεν απαντά τον κύβο `c8`. Για να διαπιστωθεί αυτό όμως χρειάστηκε να γίνει μία ερώτηση στη βάση για να διαπιστωθεί αν από το `date`=2415041 αρχίζει νέος χρόνος μιας και ο κύβος `c8` θέλουμε να μας βγάλει αποτελέσματα ως προς το επίπεδο "`year`". Βλέπουμε ότι αμέσως ο χρόνος αυξήθηκε σε σχέση με πριν που σχεδόν σε σταθερό χρόνο υπολόγιζε τα λεπτομερή επίπεδα των διαστάσεων.
- Η πέμπτη ράβδος αφορά την περίπτωση που για να διαπιστωθεί αν ο κύβος `c9` απαντά τον `c10` χρειάζεται να γίνουν δύο ερωτήσεις στη βάση. Η μία ερώτηση είναι για να διαπιστωθεί αν για `date`=2415386 αρχίζει νέο έτος μιας και η ομαδοποίηση που θέλουμε να γίνει είναι ως προς το επίπεδο `year`. Πραγματικά για `date`=2415386 αρχίζει νέο έτος (1901) κι έτσι ο αλγόριθμος θα προχωρήσει για να εξετάσει αν το `date`=2415750 είναι η τελευταία μέρα του έτους. Αυτή είναι η δεύτερη

ερώτηση που θα γίνει στη βάση. Βλέπουμε ότι ο χρόνος που χρειάζεται για να διαπιστωθεί αυτό είναι μεγαλύτερος από όλους τους προηγούμενους.

- Στην έκτη περίπτωση χρειάζεται να γίνουν τρεις ερωτήσεις στη βάση. Οι δύο είναι οι ίδιες με του προηγούμενου κύβου και η τρίτη είναι για να διαπιστωθεί αν από το `warehouse`=6 αρχίζει νέα χώρα, αφού το επίπεδο που θέλουμε να ομαδοποιήσουμε είναι η χώρα. Όπως ήταν αναμενόμενο η εκτέλεση του αλγορίθμου αυτή καθυστερεί περισσότερο από όλα.
- Στην τελευταία ράβδο αναπαρίσταται η περίπτωση στην οποία γίνονται τέσσερις ερωτήσεις στη βάση, εκ των οποίων οι τρείς είναι οι ίδιες με του προηγούμενου κύβου και η τέταρτη αφορά το αν το `warehouse`=50 είναι το τελευταίο `warehouse` το οποίο ανήκει στη χώρα αυτή.

Συνολικά διαπιστώνουμε ότι ο υπολογισμός καταλληλότητας κύβου εξαρτάται απο τον αριθμό των ερωτήσεων που θα γίνουν στη βάση για να διαπιστωθεί αν όντως τα άτομα του κύβου, ο οποίος θέλουμε να απαντηθεί, περιέχονται ως προς το ζητούμενο επίπεδο στα αντίστοιχα άτομα του κύβου ο οποίος καλείται να απαντήσει.

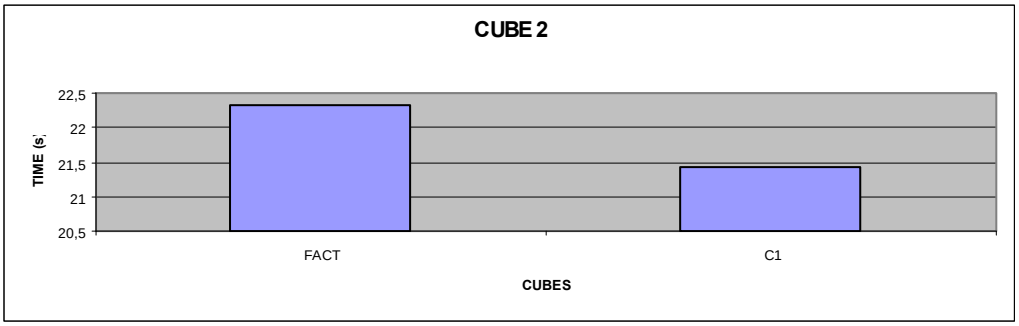
4.5 Μελέτη υπολογισμού κύβου από υλοποιημένο κύβο – LevelθLevel

Μετά τον έλεγχο για τη καταλληλότητα ενός κύβου, έγιναν πειράματα για να ελεγχθεί α) ο χρόνος που χρειάζεται για να απαντηθεί ένας κύβος από τον fact πίνακα και β) ο χρόνος που χρειάζεται για να απαντηθεί από κάποιον άλλον κύβο, ο οποίος τον απαντά, συν το χρόνο που χρειάζεται ο αλγόριθμος καταλληλότητας κύβου, για να αποφανθεί αν ο κύβος μπορεί να απαντηθεί από κάποιον άλλον συγκεκριμένο κύβο. Για το λόγο αυτό δημιουργήθηκαν δώδεκα κύβοι, που οι ορισμοί τους είναι στον παρακάτω πίνακα. Επίσης για την εξαγωγή συμπερασμάτων θα χρειαστεί και η πληροφορία του μεγέθους του κύβου δηλαδή του αριθμού των εγγραφών του.

Πίνακας 4.4 Κύβοι με συνθήκη επιλογής τύπου `Level θ Level`

	Κύβος	**# Εγγραφές**
1	c1=(date1>date2,[date,warehouse,brand],”sum”)	912033
2	c2=(date1>date2,[month,warehouse,brand],”sum”)	707831
3	c3=(date1>date2,[year,warehouse,brand],”sum”)	60938
4	c4=(date1>date2,[month,street,brand],”sum”)	707831
5	c5=(date1>date2,[year,country,brand],”sum”)	34874
6	c6=(month1>month2,[month,warehouse,brand],”sum”)	699710
7	c7=(year1>year2,[year,warehouse,brand],”sum”)	28123
8	c8=(year1>year2,[date,warehouse,brand],”sum”)	742028
9	c9=(date1>date2,[date,warehouse,ALL],”sum”)	127284
10	c10=(date1>date2,[date,ALL,ALL],”sum”)	25581
11	c11=(year1>year2,[ALL,ALL,ALL],”sum”)	1
12	c12=(year1>year2,[year,country,brand],”sum”)	25814

Στα διαγράμματα που ακολουθούν παρουσιάζεται ο χρόνος που χρειάστηκε κάποιος κύβος να απαντηθεί από τον πίνακα πληροφοριών και ο χρόνος που χρειάστηκε για να απαντηθούν από άλλους κύβους που τον απαντούν συν το χρόνο που χρειάζεται ο αλγόριθμος καταλληλότητας κύβων. Στον άξονα x είναι οι κύβοι που απαντούν τον κάθε κύβο και στον άξονα y είναι ο χρόνος που χρειάζεται για να απαντηθεί κάποιος κύβος μετρημένος σε δευτερόλεπτα.



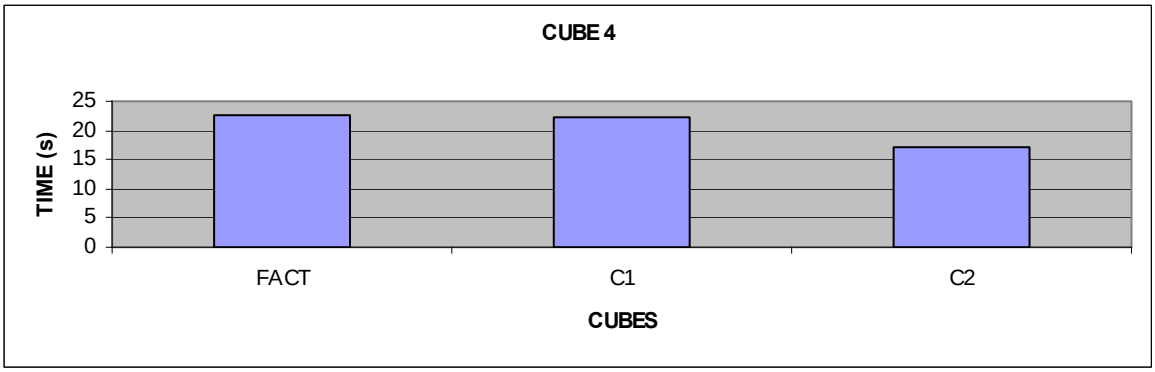
Σχήμα 4.8 Χρόνος για να απαντηθεί ο κύβος `c2`

Ο κύβος `c1` είναι ορισμένος στο λεπτομερές επίπεδο σαν τον πίνακα πληροφοριών δηλαδή και με τη συνθήκη επιλογής που έχει καταλήγουμε σε ένα μεγάλο υποσύνολο του πίνακα πληροφοριών, έναν κύβο με 912033 εγγραφές. Ο πίνακας πληροφοριών

να σημειωθεί ότι έχει 942933 εγγραφές. Παρατηρούμε ότι ο κύβος c1 απαντά πιο γρήγορα τον κύβο c2 απ' ό,τι τον απαντά ο πίνακας πληροφοριών. Αυτό οφείλεται στο ότι το πλήθος των εγγραφών του c1 είναι μικρότερο από του πίνακα πληροφοριών. Για να υπολογιστεί ο κύβος c2 χρειάζεται να γίνει ένα rollup στην ιεραρχία του χρόνου στο επίπεδο month. Ο κύβος c3 απαντιέται και απο τον c1 και τον c2. Στο διάγραμμα παρουσιάζονται οι χρόνοι που χρειάστηκε για να απαντηθεί.

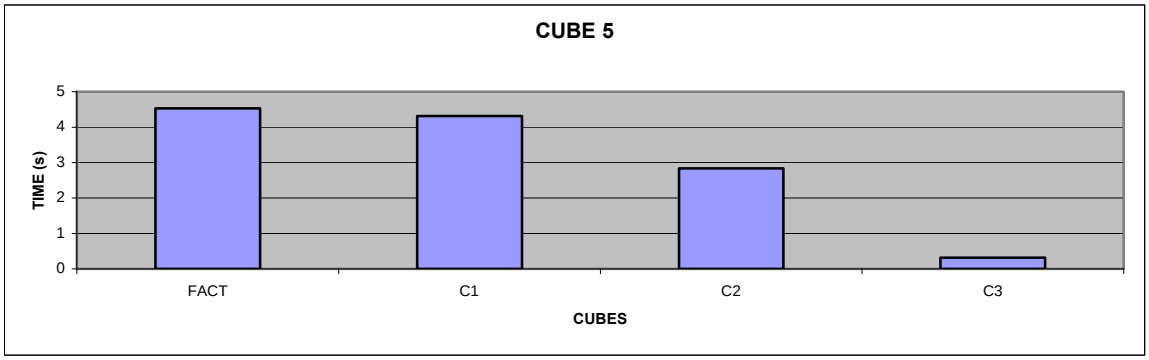
Σχήμα 4.9 Χρόνος για να απαντηθεί ο κύβος c3

Για να υπολογιστεί ο κύβος c3 πρέπει και από τον πίνακα πληροφοριών και από τον c1 να γίνει rollup δύο επιπέδων στην ιεραρχία του χρόνου από το επίπεδο date στο επίπεδο year, ενώ από τον c2 να γίνει rollup ενός επιπέδου και συγκεκριμένα από το επίπεδο month στο επίπεδο year. Βλέπουμε ότι ο c2 απαντά πιο γρήγορα γιατί είναι μικρότερου μεγέθους κι επειδή γίνεται rollup ενός επιπέδου. Ο κύβος c4 απαντιέται από τον c1 και από τον c2 .



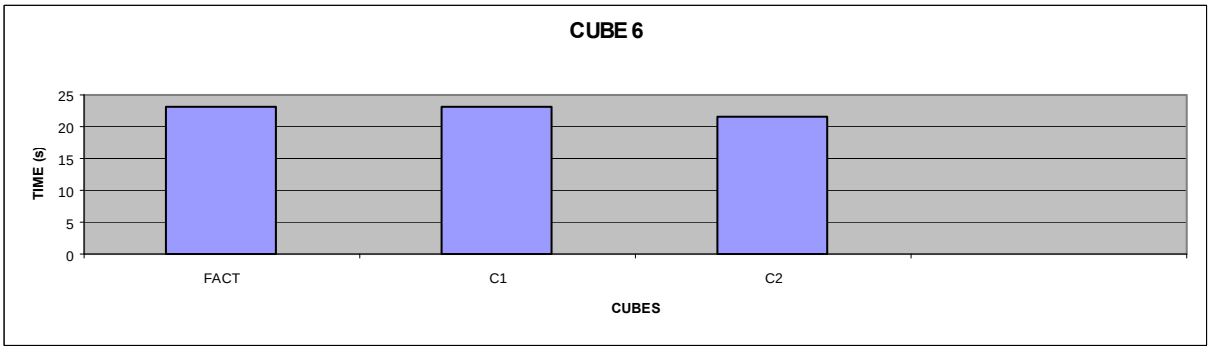
Σχήμα 4.10 Χρόνος για να απαντηθεί ο κύβος `c4`.

Ο κύβος `c2` τον απαντά πιο γρήγορα γιατί στη διάσταση του χρόνου είναι ορισμένοι στο ίδιο επίπεδο, ενώ στη διάσταση της τοποθεσίας χρειάζεται να γίνει ένα rollup από το επίπεδο `warehouse` στο επίπεδο `street`. Οι κύβος `c1` και ο πίνακας πληροφοριών είναι σίγουρα πιο αργοί και λόγω μεγέθους, αλλά και λόγω των rollup των δύο επιπέδων σε δύο ιεραρχίες.



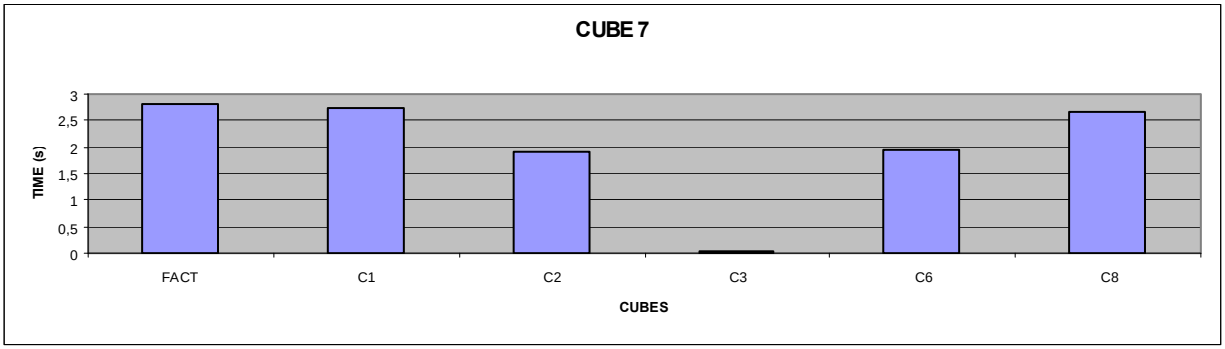
Σχήμα 4.11 Χρόνος για να απαντηθεί ο κύβος `c5`

Ο κύβος `c5` απαντιέται απο τον `c1`, `c2` και `c3`. Βλέπουμε ότι ο `c3` τον απαντά πιο γρήγορα από όλους και με μεγάλη διαφορά. Αυτό οφείλεται στο ότι ο `c3` είναι πολύ μικρότερος από τους υπόλοιπους κύβους. Ακόμη ο `c5` για να υπολογιστεί απο τον `c3` χρειάζεται μόνο στη διάσταση της τοποθεσίας να κάνει τέσσερα rollups από το επίπεδο `warehouse` στο επίπεδο `country`, ενώ από τους υπόλοιπους κύβους πρέπει να κάνει rollup και στη διάσταση του χρόνου.



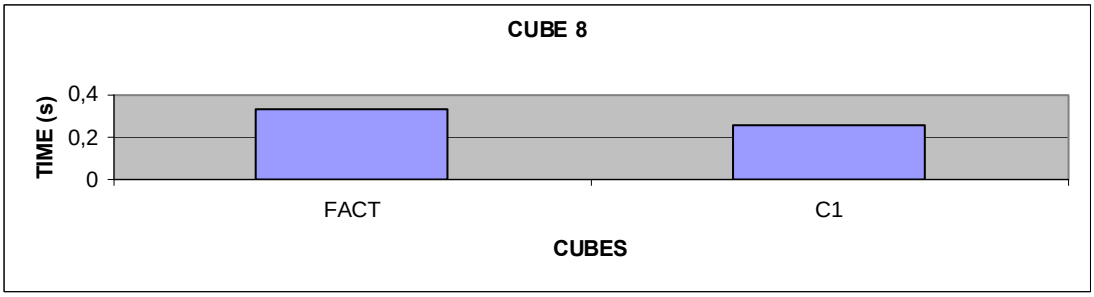
Σχήμα 4.12 Χρόνος για να απαντηθεί ο κύβος `c6`

Ο κύβος `c6` απαντιέται απο τους `c1` και `c2`, με τον κύβο `c2` να έχει καλύτερο χρόνο απο τον `c1`. Αυτό οφείλεται και στο μέγεθος των κύβων, αφού ο κύβος `c2` είναι μικρότερος απο τον `c1`, αλλά και στο ό,τι ο κύβος `c2` είναι ορισμένος στα ίδια επίπεδα με τον κύβο `c6`, άρα δεν χρειάζεται να κάνει συζεύξεις με άλλους πίνακες για να εφαρμόσει τη συνθήκη `month1>month2` αφού ο κύβος `c2` έχει και τα δύο αυτά πεδία. Ακόμη παρατηρούμε ότι οι χρόνοι είναι μεγαλύτεροι με τα προηγούμενα πειράματα και αυτό οφείλεται στο ότι ο κύβος `c6` είναι μεγάλος.



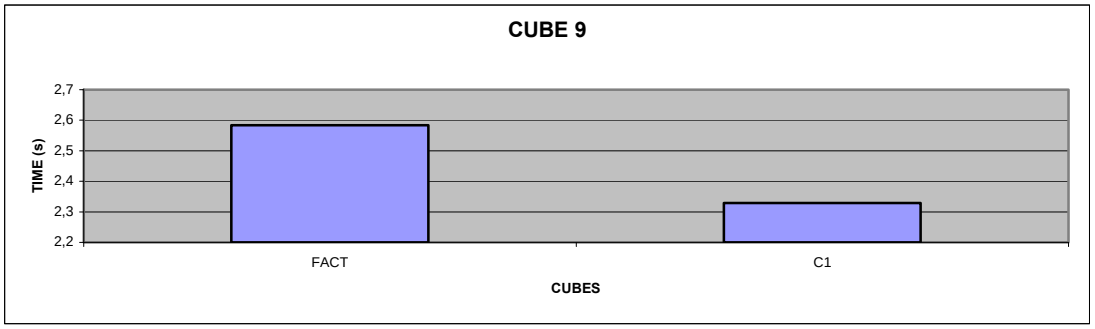
Σχήμα 4.13 Χρόνος για να απαντηθεί ο κύβος `c7`

Στο παραπάνω διάγραμμα παρουσιάζονται οι χρόνοι που απαιτούνται για να απαντηθεί ο κύβος `c7` από τους κύβους που απαντιέται. Βλέπουμε πολύ μεγάλη διαφορά στο χρόνο που απαιτείται από τον `c3`. Αυτό οφείλεται στο ότι ο `c3` είναι ορισμένος στα ίδια επίπεδα με τον `c7` και η συνθήκη επιλογής του `c7` μπορεί να εφαρμοστεί άμεσα στον κύβο `c3` χωρίς να χρειαστεί να γίνει σύζευξη με κανέναν άλλο πίνακα. Σε χρόνο ακολουθούν ο `c2` και ο `c6` οι οποίοι για να υπολογίσουν τον `c7` χρειάζεται να συζευχθούν με τον πίνακα `months`. Έπειτα ακολουθούν ο κύβος `c1` και c8, οι οποίοι για να υπολογίσουν τον κύβο `c7` πρέπει να συζευχθούν με τον πίνακα dates. Συνολικά με τον κύβο `c3` δεν χρειάστηκε καμία σύζευξη, δηλαδή κανένα rollup, με τον κύβο `c2` και `c6` χρειάστηκε ένα rollup και με τον κύβο `c1` και `c8` χρειάστηκαν δύο rollups.



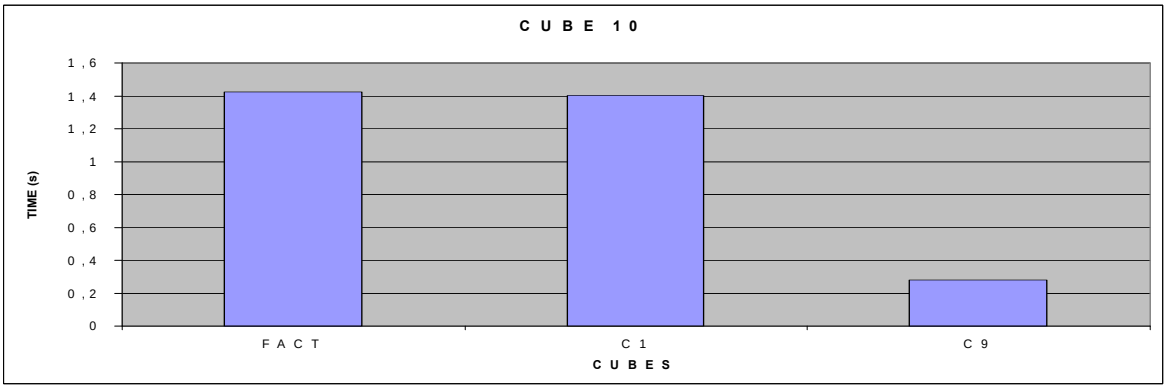
Σχήμα 4.14 Χρόνος για να απαντηθεί ο κύβος `c8`

Για τον κύβο `c8` ισχύουν τα ίδια που ισχύουν και για τον κύβο `c2`, δηλαδή ότι απαντιέται πιο γρήγορα από τον `c1` σε σχέση με τον πίνακα πληροφοριών λόγω μεγέθους. Για να απαντηθεί ο κύβος `c8` από τον `c1` αλλά και από τον πίνακα πληροφοριών, χρειάζεται να συζευχθεί με τον πίνακα `dates`.



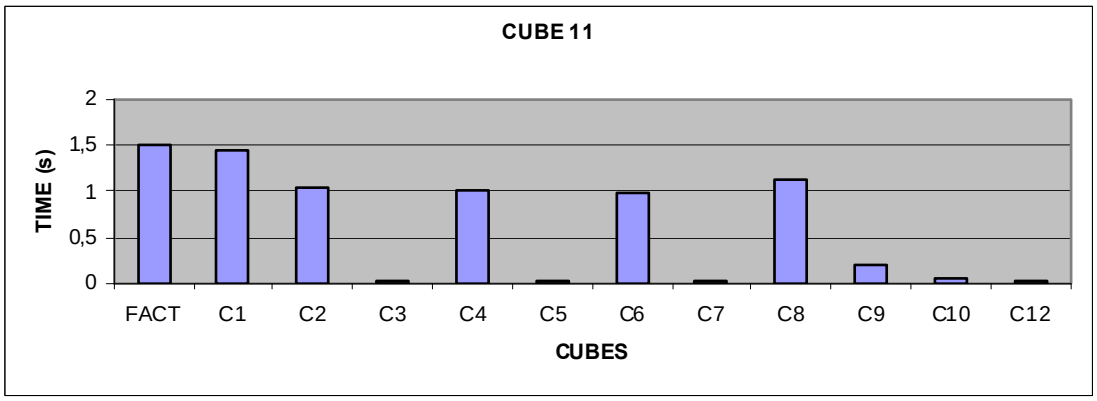
Σχήμα 4.15 Χρόνος για να απαντηθεί ο κύβος `c9`

Ο κύβος `c9` απαντιέται πιο γρήγορα από τον κύβο `c1` λόγω μεγέθους σε σχέση με τον πίνακα πληροφοριών.



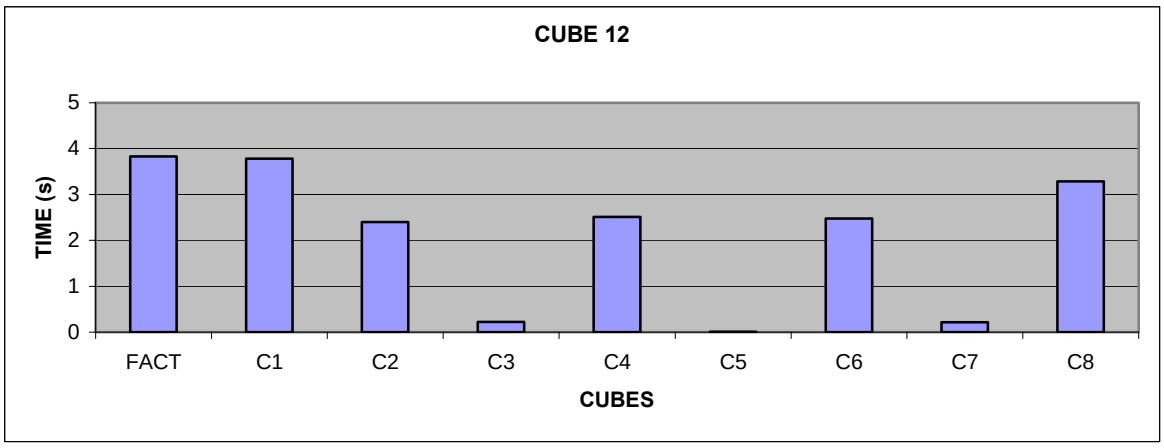
Σχήμα 4.16 Χρόνος για να απαντηθεί ο κύβος `c10`

Βλέπουμε ότι ο κύβος `c10` απαντιέται πολύ πιο γρήγορα από τον κύβο `c9` σε σχέση με τους υπόλοιπους κι αυτό οφείλεται μόνο στο μέγεθος των κύβων μιας και σε κανέναν πίνακα δεν κάνουμε κανένα rollup γιατί όλοι είναι ορισμένοι στο ίδιο επίπεδο.



Σχήμα 4.17 Χρόνος για να απαντηθεί ο κύβος `c11`

Ο κύβος `c11` απαντιέται από όλους τους κύβους. Πιο γρήγορα απαντιέται απο τον `c12` και ακολουθούν `c7`, `c5` και `c3`. Η διαφορά της ταχύτητας εκτέλεσης οφείλεται στο μέγεθος των κύβων αλλά και στο ότι οι συγκεκριμένοι κύβοι δεν χρειάζεται να συζευχθούν με άλλους πίνακες για να τον υπολογίσουν. Ακόμη για τον υπολογισμό από τον `c7` και από τον `c12` δεν χρειάζεται καν να υπάρχει η συνθήκη επιλογής του κύβου `c11`, αφού και αφού και αυτοί έχουν την ίδια συνθήκη επιλογής.



Σχήμα 4.18 Χρόνος για να απαντηθεί ο κύβος `c12`

Ο κύβος c12 είναι ο πιο ψηλά ορισμένος κύβος με συνθήκη επιλογής σε υψηλό επίπεδο. Βλέπουμε ότι απαντιέται από τον κύβο c5 σε ελάχιστο χρόνο. Αυτό οφείλεται στο γεγονός ότι είναι ορισμένοι στα ίδια επίπεδα, καθώς και η συνθήκη επιλογής του κύβου c12 μπορεί να εφαρμοστεί απευθείας στον κύβο c5 χωρίς να κάνει συζεύξεις με άλλους πίνακες. Ο c3 επίσης απαντά πολύ γρήγορα. Αυτός είναι ορισμένος στο ίδιο επίπεδο στη διάσταση του χρόνου, αλλά όχι στη διάσταση της τοποθεσίας. Για να υπολογιστεί ο c12 από τον c3 χρειάζεται να συζευχθεί με τον πίνακα ware. Ο πίνακας αυτός έχει όμως μόνο 100 εγγραφές γι αυτό γίνεται πολύ γρήγορα η σύζευξη. Αν έπρεπε η σύζευξη να γίνει με τον πίνακα dates θα είχαμε πολύ μεγαλύτερα νούμερα μιας και ο πίνακας dates είναι πολύ μεγαλύτερος. Ο κύβος c7 απαντάει επίσης τον c12 σε πολύ λίγο χρόνο. Αυτός πάλι χρειάζεται να συζευχθεί με τον πίνακα ware που όπως αναφέραμε παραπάνω , η σύζευξη αυτή γίνεται πολύ γρήγορα, αλλά επίσης να σημειωθεί ότι για να υπολογιστεί ο κύβος c12 δεν χρειάζεται στο "where" του sql ερωτήματος να μπει η συνθήκη επιλογής μιας και είναι η ίδια με του κύβου c7. Ο c4 και ο c6 ακολουθούν στο χρόνο υπολογισμού. Αυτοί για να υπολογίσουν τον c12 πρέπει να συζευχθούν με τον street και με τον ware αντίστοιχα, αλλά χρειάζεται να συζευχθούν και με τον months για να μπορέσουν να υπολογίσουν την συνθήκη επιλογής καθώς και τη συνθήκη ομαδοποίησης. Ο c1 και ο c8 αργούν περισσότερο από όλους τους προηγούμενους γιατί για να υπολογιστούν εκτός του ότι πρέπει να συζευχθούν με τον πίνακα ware, όπως και κάποιοι προηγούμενοι, πρέπει για τον υπολογισμό της συνθήκης επιλογής και της συνθήκης ομαδοποίησης να συζευχθούν με τον πίνακα dates, ο οποίος είναι αρκετά μεγάλος και κυρίως αρκετά μεγαλύτερος από τον months, γι αυτό παρατηρείται και η διαφορά αυτή με τους c2 και c6.

Συμπέρασμα: Από την μελέτη των παραπάνω πειραμάτων προκύπτει, ότι ο καλύτερος κύβος για να απαντήσει κάποιον νέο κύβο είναι αυτός ο οποίος έχει τις λιγότερες εγγραφές και δεν χρειάζεται να συζευχθεί για να απαντήσει.

Συγκεκριμένα σύζευξη δεν θα χρειαστεί αν οι κύβοι είναι ορισμένοι στα ίδια επίπεδα και ισχύει ένα απο τα παρακάτω:

- Να έχουν την ίδια συνθήκη επιλογής, έτσι ώστε να μην χρειαστεί να εφαρμοστεί

- Η συνθήκη επιλογής του κύβου που θέλουμε να απαντηθεί να είναι στο ίδιο επίπεδο με τον κύβο που καλείται να τον απαντήσει.

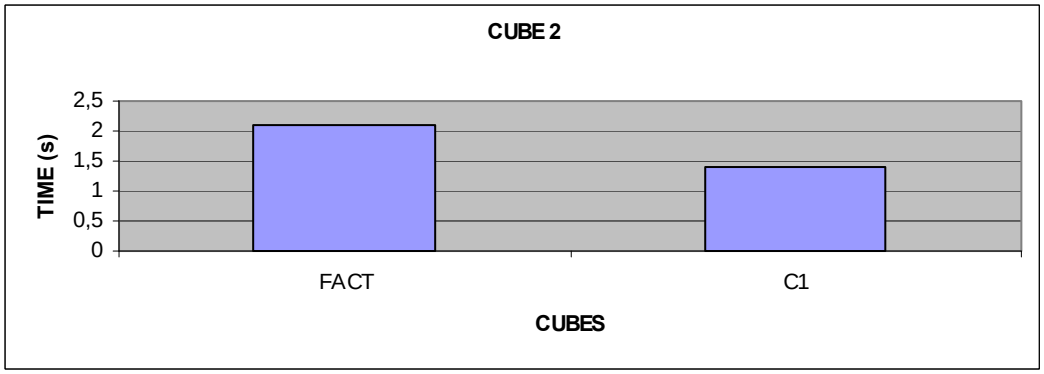
4.6 Μελέτη υπολογισμού κύβου από υλοποιημένο κύβο – LevelθValue

Για την μελέτη της περίπτωσης που ο κύβος έχει συνθήκη επιλογής τύπου `Level θ Value` δημιουργήθηκαν εννιά κύβοι, των οποίων οι ορισμοί περιέχονται στον πίνακα που ακολουθεί όπως επίσης και το πλήθος των εγγραφών τους. Εξετάστηκε πόσος χρόνος χρειάζεται για να απαντήσει ένας κύβος κάποιον άλλον, συμπεριλαμβανομένου και του χρόνου εκτέλεσης του αλγορίθμου καταλληλότητας κύβων.

Πίνακας 4.5 Κύβοι με συνθήκη επιλογής `Level θ Value`

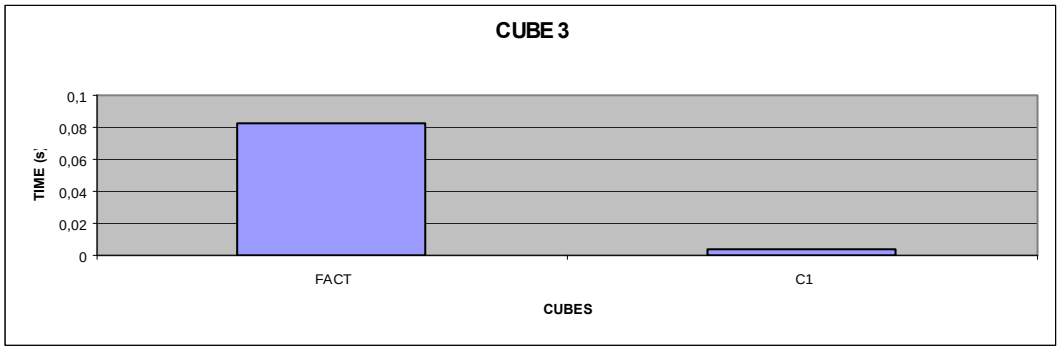
1	c1=(date>=2451180,[date,warehouse,brand],”sum”)	742028
2	c2=(date>=2451180,[year,country,brand],”sum”)	25874
3	c3=([date>=2451180,date<=2451276],[date,warehouse,brand],”sum”)	26888
4	c4=([date>=2451545,city>=1110,city<=1120],[month,city,brand],”sum”)	202018
5	c5=([date>=2451545,country=1000],[year,country,brand],”sum”)	16875
6	c6=([date>=2451545,country=1000,brand>=50,brand<=12302],[year,country,brand],”sum”)	11487
7	c7=([date>=2451911,city>=1110,city<=1120],[month,city,brand],”sum”)	134764
8	c8=([date>=2451911,date<=2452122,city>=1110,city<=1120],[month,city,brand],”sum”)	39273
9	c9=([date>=2451911,date<=2452122,country=1000,brand>=70,brand<=12302],[month,city,brand],”sum”)	26689

Ο κύβος c1 είναι ορισμένος στο λεπτομερές επίπεδο κι έχει μία συνθήκη επιλογής στο χαμηλότερο επίπεδο της ιεραρχίας του χρόνου. Ο κύβος c2 απαντιέται από τον κύβο c1 και φυσικά από τον πίνακα πληροφοριών.



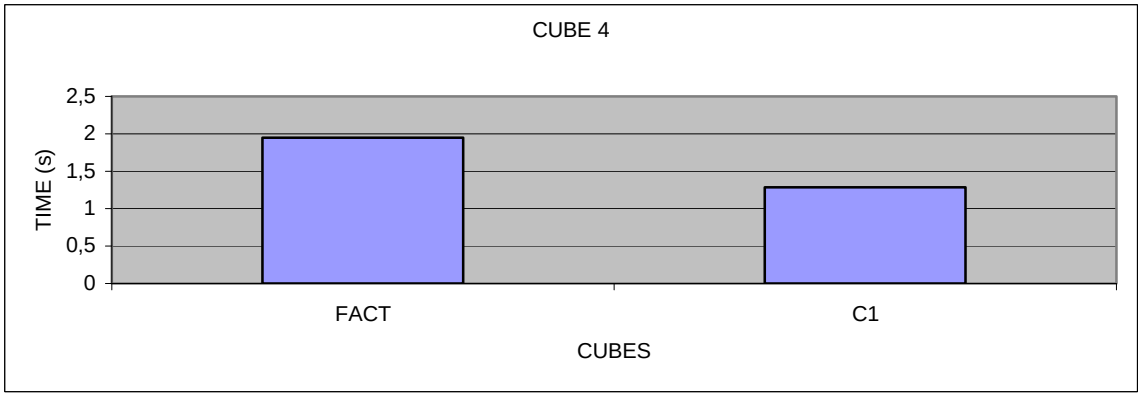
Σχήμα 4.19 Χρόνος για να απαντηθεί ο κύβος c2

Ο κύβος c1 απαντά αρκετά πιο γρήγορα τον κύβο c2 απ' ό,τι ο πίνακας πληροφοριών. Αυτό οφείλεται στο γεγονός ότι ο κύβος c1 είναι πιο μικρός από τον πίνακα πληροφοριών. Ακόμη να επισημανθεί ότι για να υπολογιστεί ο κύβος c2 από τον c1 δεν χρειάζεται η συνθήκη επιλογής του να συμπεριληφθεί στο "where" του sql ερωτήματος, αλλά πρέπει ο c1 να συζευχθεί με τον πίνακα dates, για να μπορέσει να γίνει το rollup από το επίπεδο date στο επίπεδο year που απαιτεί η συνθήκη ομαδοποίησης του κύβου c2.



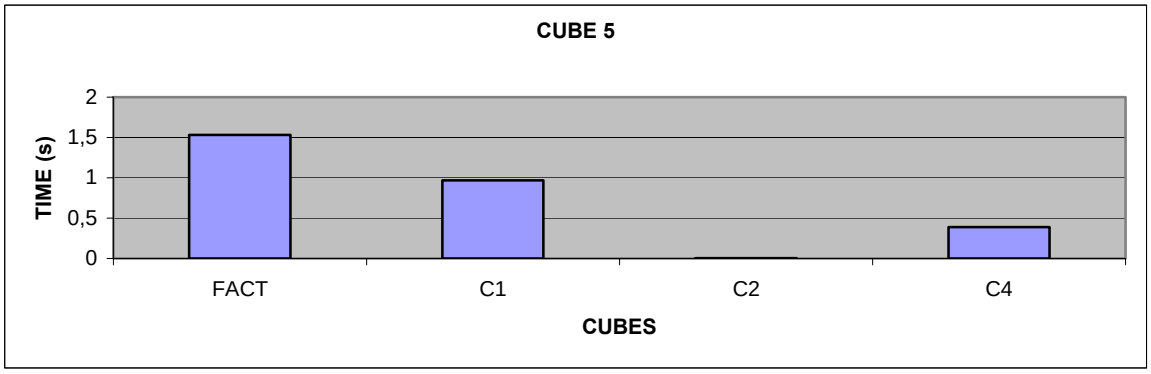
Σχήμα 4.20 Χρόνος για να απαντηθεί ο κύβος c3

Ο κύβος c3 απαντιέται πολύ πιο γρήγορα απο τον κύβο c1 σε σχέση με τον πίνακα πληροφοριών. Αυτό οφείλεται στο ότι ο κύβος c1 είναι πολύ μικρότερος του πίνακα πληροφοριών και είναι ορισμένος στο ίδιο επίπεδο με τον κύβο c3. Ακόμη δεν χρειάζεται στο "where" του sql ερωτήματος να συμπεριλάβουμε το πρώτο άτομο γιατί είναι ίδιο με του κύβου c1.



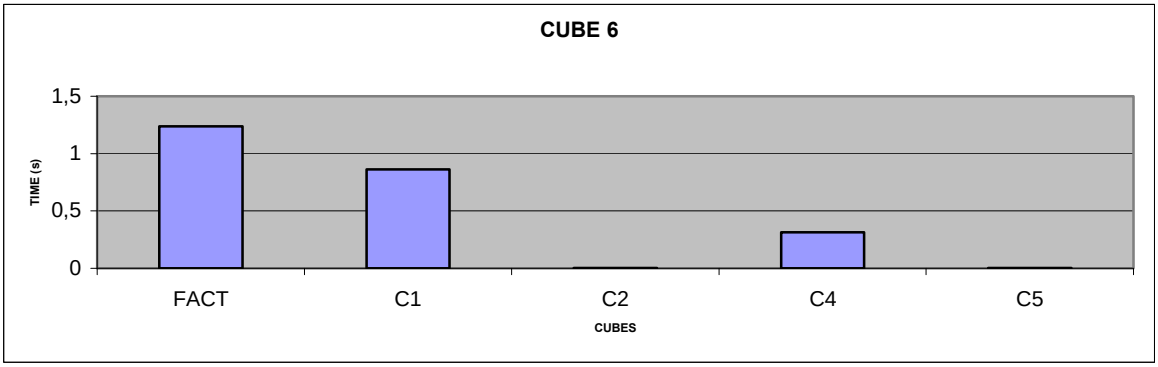
Σχήμα 4.21 Χρόνος για να απαντηθεί ο κύβος c4

Βλέπουμε ότι ο κύβος c4 απαντιέται πιο γρήγορα από τον κύβο c1 σε σχέση με τον πίνακα πληροφοριών κι αυτό οφείλεται στο μέγεθος των πινάκων. Για να υπολογιστεί από τον κύβο c1 πρέπει να συζευχθεί με τον πίνακα dates (73K εγγραφές) για να υπολογίσει τη συνθήκη ομαδοποίησης και με τον πίνακα ware για να υπολογίσει τη συνθήκη επιλογής. Το ίδιο ακριβώς πρέπει να γίνει και για να υπολογιστεί από τον πίνακα πληροφοριών, αλλά ο πίνακας αυτός είναι μεγαλύτερος απο τον c1 κατά 200K εγγραφές. Εδώ φαίνεται ξεκάθαρα το πόσο σημαντικός παράγοντας είναι το μέγεθος του πίνακα για τη διαμόρφωση χρόνου απάντησης των κύβων.



Σχήμα 4.22 Χρόνος για να απαντηθεί ο κύβος c5

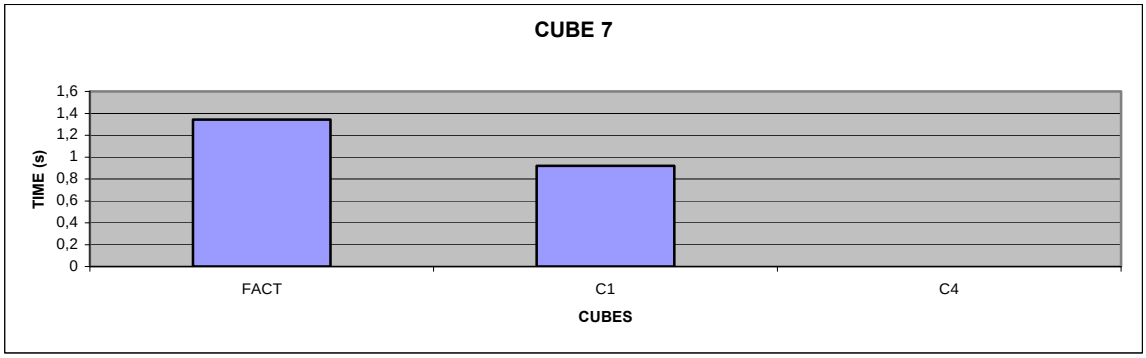
Ο κύβος c5 βλέπουμε ότι απαντιέται πολύ γρήγορα από τον κύβο c2 και με τον c4 να ακολουθεί. Αυτό συμβαίνει γιατί ο κύβος c2 για να τον απαντήσει δεν χρειάζεται να συζευχθεί με κανέναν άλλον πίνακα καθώς επίσης είναι και πολύ μικρός πίνακας σε σχέση με τους υπόλοιπους που τον απαντούν. Πρέπει να γίνει ένα rollup δύο επιπέδων στο άτομο date>=2451545 της συνθήκης επιλογής του , κι αυτό επιτυγχάνεται με τη δημιουργία ενός εμφωλευμένου query στο "where" του sql ερωτήματος. Ο κύβος c4 έχει ένα άτομο ίδιο με τον κύβο c5 κι έτσι δεν χρειάζεται να συμπεριληφθεί στην "where" πρόταση του sql ερωτήματος. Ο πίνακας c4 είναι αρκετά πιο μεγάλος από τον c2 κι επιπλέον κάνει σύζευξη με δύο πίνακες οι οποίοι όμως είναι σχετικά μικρού μεγέθους. Με τον πίνακα months κάνει σύζευξη για να υπολογίσει τη συνθήκη ομαδοποίησης και με τον πίνακα city για να υπολογίσει το αντίστοιχο άτομο της συνθήκης επιλογής. Ο c1 και ο πίνακας πληροφοριών για να απαντήσουν πρέπει να συζευχθούν με δύο πίνακες, εκ των οποίων ο ένας (dates) είναι μεγάλος (73000 εγγραφές), αλλά ο κυριότερος λόγος, για τον οποίο ο χρόνος είναι τόσο μεγαλύτερος, είναι επειδή είναι μεγαλύτεροι από τους κύβους c2 και c4



Σχήμα 4.23 Χρόνος για να απαντηθεί ο κύβος c6

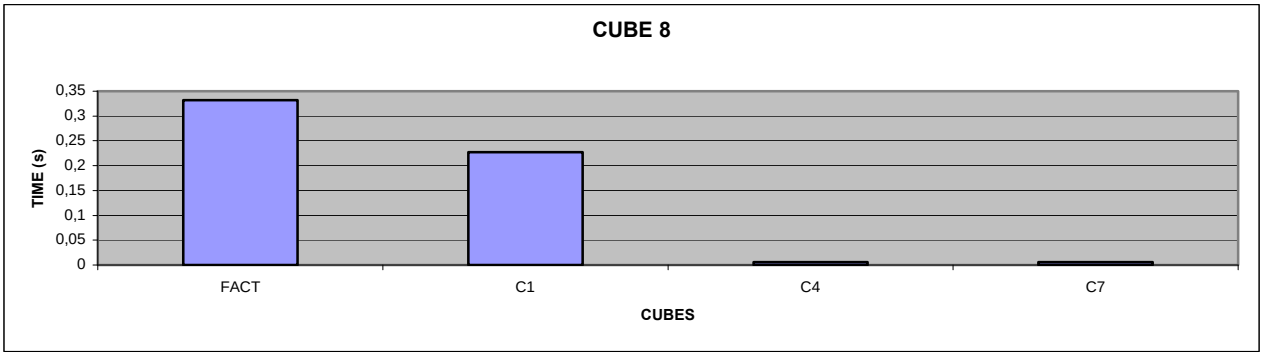
Ο c6 απαντιέται πολύ γρήγορα από τον c5 και αυτό συμβαίνει γιατί ο κύβος c5 είναι μικρός κύβος (16875 εγγραφές), είναι ορισμένος στα ίδια επίπεδα και θα εφαρμοστούν σε αυτόν μόνο τα άτομα της συνθήκης επιλογής brand>=50 και brand<=12302, μιας και τα υπόλοιπα είναι ίδια με του c5. Ο κύβος c2, ο οποίος είναι μικρού μεγέθους κι αυτός (25874) απαντά επίσης πολύ γρήγορα τον c6. Για να υπολογιστεί ο c6 από τον c2, επίσης δεν χρειάζεται να γίνει καμία σύζευξη με κανέναν πίνακα, όμως θα χρειαστούν περισσότερα άτομα στην πρόταση "where", ένα

εκ των οποίων θα είναι εμφωλευμένο query. Ακολουθεί ο κύβος c4, ο οποίος για να υπολογίσει τον c6 χρειάζεται να συζευχθεί με δύο πίνακες, οι οποίοι είναι και οι δύο μικρού μεγέθους. Συγκεκριμένα, ο πίνακας months θα χρειαστεί για τον υπολογισμό της συνθήκης ομαδοποίησης και ο πίνακας city για τον υπολογισμό της συνθήκης επιλογής. Ο c1 απαντά επίσης τον c6 με μεγαλύτερη καθυστέρηση σε σχέση με τους προηγούμενους αφού πρέπει να εκτελεστούν όλα τα άτομα της συνθήκης επιλογής και να γίνουν συζεύξεις με τον πίνακα dates (73K εγγραφές) και ware ώστε να εκτελεστούν οι συνθήκες ομαδοποίησης.



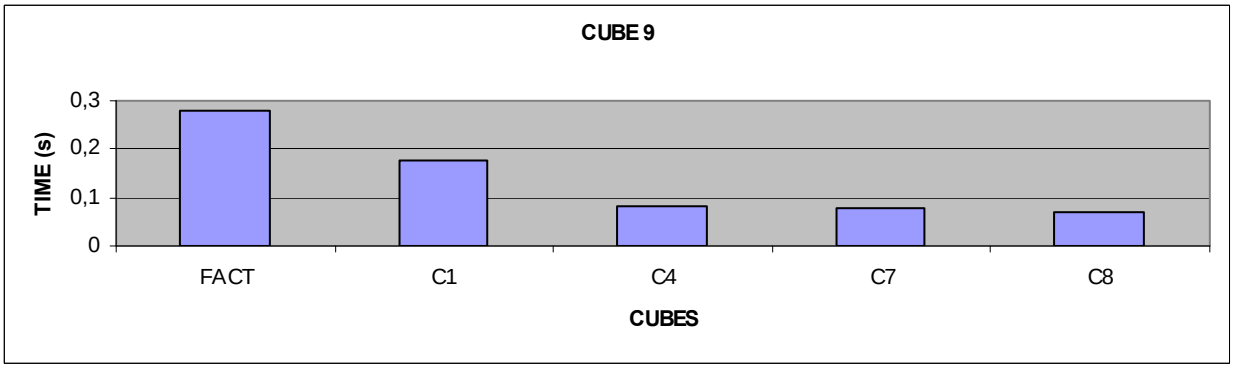
Σχήμα 4.24 Χρόνος για να απαντηθεί ο κύβος c7

Ο κύβος c7 απαντιέται πιο γρήγορα από τον κύβο c4 που είναι μικρότερος του πίνακα πληροφοριών και του κύβου c1 κατά 500K εγγραφές. Ο κύβος c4 για να απαντήσει τον κύβο c7 δεν χρειάζεται να συζευχθεί με κανέναν πίνακα, ενώ ο c1 πρέπει να συζευχθεί με τον πίνακα dates (73K) για τον υπολογισμό της συνθήκης ομαδοποίησης και με τον πίνακα ware για τον υπολογισμό και της συνθήκης επιλογής και της συνθήκης ομαδοποίησης.



Σχήμα 4.25 Χρόνος για να απαντηθεί ο κύβος `c8`

Ο κύβος `c8` απαντιέται γρήγορα από τον κύβο `c7` και από τον `c4`. Με τους κύβους αυτούς δεν χρειάζεται να κάνει καμία σύζευξη για να υπολογίσει τις συνθήκες επιλογής και ομαδοποίησης. Ακόμη οι κύβοι αυτοί είναι πολύ μικρότερου μεγέθους σε σχέση με τον κύβο `c1` (135K και 202K αντίστοιχα). Για να απαντηθεί από τον κύβο `c1` πρέπει να εφαρμόσει όλα τα άτομα της συνθήκης επιλογής και να συζευχθεί με τον πίνακα `dates` και τον πίνακα ware για να γίνουν τα rollups για τον υπολογισμό της συνθήκης επιλογής και της συνθήκης ομαδοποίησης.



Σχήμα 4.26 Χρόνος για να απαντηθεί ο κύβος `c9`

Ο κύβος `c9` απαντιέται πιο γρήγορα από τον κύβο `c8`. Γι να υπολογιστεί από τον `c8` πρέπει να γίνει σύζευξη με τον πίνακα `city` για τον υπολογισμό του κατάλληλου ατόμου της συνθήκης επιλογής. Το sql ερώτημα υπολογισμού του `c9` από τον `c8` έχει τα λιγότερα άτομα στη συνθήκη επιλογής και είναι ο μικρότερος από όλους. Ο υπολογισμός του επίσης από τον `c7` και τον `c4` είναι επίσης γρήγορος. Συγκεκριμένα για να υπολογιστεί από αυτούς πρέπει να κάνει μόνο μία σύζευξη με τον πίνακα `city`.

Οι πίνακες αυτοί είναι μεγαλύτεροι από τον `c8` αλλά αρκετά μικρότεροι (500K-600K αντίστοιχα) από τον κύβο `c1`. Ο `c1` κάνει σύζευξη με τους πίνακες `dates` και `ware` για τον υπολογισμό της συνθήκης ομαδοποίησης και της συνθήκης επιλογής.

Συνολικά παρατηρήσαμε ότι ο καλύτερος κύβος είναι αυτός που έχει τις λιγότερες εγγραφές κι αυτός που κάνει τις λιγότερες συζεύξεις για να απαντήσει τον νέο κύβο. Συγκεκριμένα, παρατηρήσαμε ότι η μεγαλύτερη καθυστέρηση προκαλείται, όταν πρόκειται να χρησιμοποιήσουμε μεγάλο κύβο για τον υπολογισμό του νέου κύβου και αυτός κάνει σύζευξη με πίνακα ο οποίος είναι μεγάλου μεγέθους.

Στο παράρτημα Α βρίσκονται οι ορισμοί των κύβων με σχεσιακά αιτήματα.

ΚΕΦΑΛΑΙΟ 5. ΑΛΟΓΟΡΙΘΜΟΣ ΕΠΙΛΟΓΗΣ ΚΥΒΟΥ

5.1 Ορισμός προβλήματος

Μετά την πειραματική μελέτη που παρουσιάστηκε στο προηγούμενο κεφάλαιο καταλήξαμε στο συμπέρασμα, ότι ο κύβος ο οποίος απαντάει πιο γρήγορα κάποιον άλλον νέο κύβο είναι συνάρτηση δύο παραγόντων:

- Μέγεθος παλιού κύβου
- Αριθμός συζεύξεων και μεγέθους πίνακα σύζευξης

Δεδομένου ότι οι χρήστες κάνουν OLAP ερωτήσεις και τα αποτελέσματα που προκύπτουν, δηλαδή νέοι κύβοι τους οποίους αποθηκεύουμε, μπορούν να χρησιμοποιηθούν για να απαντήσουν μία νέα ερώτηση, προκύπτει το ζήτημα της επιλογής του κύβου που θα δώσει πιο γρήγορα την απάντηση. Από τη μελέτη που διεξήχθη στο προηγούμενο κεφάλαιο και τα συμπεράσματα που διεξήχθησαν, καταλήξαμε στην πρόταση ενός αλγορίθμου ο οποίος επιλέγει ανάμεσα από κάποιους υποψήφιους κύβους, οι οποίοι απαντάνε τον νέο κύβο, τον κύβο ο οποίος θεωρείται αποτελεσματικότερος για τον υπολογισμό του νέου κύβου. Ο αλγόριθμος επιλογής κύβου αρχικά εντοπίζει τους υλοποιημένους κύβους, οι οποίοι μπορούν να απαντήσουν τον νέο κύβου χρησιμοποιώντας τον αλγόριθμο καταλληλότητας κύβου

και στη συνέχεια λαμβάνοντας υπόψη τα προαναφερθέντα συμπεράσματα και τον αλγόριθμο σύζευξης ο οποίος χρησιμοποιείται στο προγραμματιστικό περιβάλλον που χρησιμοποιούμε, επιλέγει τον κύβο με το χαμηλότερο κόστος. Μελετήθηκαν τέσσερις διαφορετικοί τρόποι εσωτερικής αναπαράστασης των υποψήφιων κύβων, δηλαδή των κύβων που μπορούν να απαντήσουν τον νέο προς απάντηση κύβο. Στην επόμενη παράγραφο παρουσιάζεται ο αλγόριθμος και η ανάλυσή του.

5.2 Μέθοδοι εντοπισμού υποψήφιων κύβων

Για τη δημιουργία του διανύσματος των υποψήφιων κύβων, δηλαδή τον κύβων οι οποίοι μπορούν να απαντήσουν έναν νέο προς απάντηση κύβο προτείνουμε τέσσερις τεχνικές με τις οποίες επιλέγουμε κάθε φορά τον προς εξέταση καταλληλότητας κύβο. Το γενικό πλαίσιο δημιουργίας του διανύσματος υποψήφιων κύβων (candidates) παρουσιάζεται στον παρακάτω πίνακα:

```
Αλγόριθμος δημιουργίας διανύσματος υποψήφιων κύβων
Είσοδος: Κύβος c_new, διάνυσμα cubes με όλους τους υλοποιημένους κύβους
Έξοδος: Διάνυσμα candidates με τους υποψήφιους κύβους
1. while (life condition)
2. {   pick a cube;
3.     decide usability;
4.     if (usable)
5.           add to candidates;
6. }
7. return candidates;
```

Πίνακας 5.1 Αλγόριθμος δημιουργίας διανύσματος υποψήφιων κύβων

Στον αλγόριθμο δίνεται ως είσοδος ο υποψήφιος προς απάντηση κύβος και ένα πλήθος υλοποιημένων κύβων. Το ζητούμενο είναι να μας επιστραφεί ένα διάνυσμα με υποψήφιους κύβους, οι οποίοι απαντούν τον κύβο μας c_{new} και οι κύβοι αυτοί να είναι στην κατάλληλη σειρά.

Συγκεκριμένα, ο αλγόριθμος αναφέρει ότι επιλέγουμε έναν κύβο ανάλογα με την τεχνική που χρησιμοποιούμε (γραμμή 2). Ελέγχουμε αν αυτός ο κύβος μπορεί να απαντήσει τον c_{new} χρησιμοποιώντας τον αλγόριθμο καταλληλότητας κύβων (γραμμή

3). Αν αυτός ο κύβος μπορεί να απαντήσει τον c_{new}, τότε τον προσθέτουμε στο διάνυσμα με τους `candidates` (γραμμές 4-5). Στη γραμμή 1 αναφέρεται η συνθήκη, η οποία πρέπει να ισχύει για να συνεχίζουμε να αναζητούμε κύβους. Η συνθήκη αυτή μπορεί να είναι α) ο αριθμός των υποψήφιων κύβων που έχουν ήδη βρεθεί ή β) ο αριθμός των προσπαθειών που έχουμε κάνει για να εντοπίσουμε υποψήφιο κύβο, δηλαδή το πλήθος των επαναλήψεων του βρόχου. Παρακάτω παρουσιάζονται οι τεχνικές με τις οποίες μπορούμε να προσπελάσουμε τους ήδη υλοποιημένους κύβους.

5.2.1 Επιλογή ανάλογα με την άφιξη (MRC –most recently created)

Σύμφωνα με τη μέθοδο αυτή, οι κύβοι προσπελαύνονται ανάλογα με το χρόνο δημιουργίας τους. Αυτός που θα εξεταστεί πρώτος αν απαντάει τον προς απάντηση κύβο c_{new} είναι αυτός ο οποίος έχει δημιουργηθεί τελευταίος.

Έτσι για παράδειγμα αν έχουμε δημιουργήσει δέκα κύβους με ονόματα `c1,…c10` ανάλογα με το χρόνο άφιξής τους και θέλουμε να βρούμε τους υποψήφιους για να απαντήσουν τον κύβο c_{new}, θα ελέγξουμε πρώτα τον `c10`, έπειτα τον `c9` κοκ.

5.2.2 Επιλογή του πιο πρόσφατα χρησιμοποιημένου (MRU- most recently used)

Σύμφωνα με τη μέθοδο αυτή, οι κύβοι που ελέγχονται πρώτοι είναι αυτοί που πιο πρόσφατα απάντησαν κάποιον άλλον κύβο. Για παράδειγμα έστω ότι θέλουμε να απαντήσουμε τον κύβο `c6`, αυτό που θα κάνουμε είναι να βρούμε ποιός απάντησε τον κύβο `c5`, αν αυτός που απάντησε τον κύβο `c5` απαντάει και τον `c6` τότε τον προσθέτουμε στο διάνυσμα. Στη συνέχεια ελέγχουμε το ποιος απάντησε τον κύβο `c4`. Σε περίπτωση που αυτός που τον απάντησε απαντάει και τον `c6` τότε τον προσθέτουμε στη λίστα. Η διαδικασία συνεχίζεται μέχρι να τελειώσουν οι κύβοι, είτε μέχρι να έχει ξεπεραστεί το κατώφλι το οποίο έχουμε καθορίσει.

5.2.3 Επιλογή του μικρότερου (SF-smallest first)

Σύμφωνα με τη μέθοδο αυτή, ελέγχεται πρώτα ο μικρότερος σε μέγεθος κύβος. Αν αυτός ο κύβος απαντά τον κύβο που θέλουμε να απαντήσουμε τότε αυτός εισάγεται στο διάνυσμα `candidates`. Για να επιτευχθεί ο έλεγχος των κύβων από τον μικρότερο

προς το μεγαλύτερο, αυτό που κάνουμε είναι να ταξινομούμε με τον αλγόριθμο γρήγορης ταξινόμησης το διάνυσμα των ήδη υλοποιημένων κύβων και στη συνέχεια να τους παίρνουμε με τη σειρά, είτε μέχρι να μην υπάρχουν άλλοι προς έλεγχο κύβοι, είτε μέχρι να ξεπεραστεί το κατώφλι, το οποίο έχουμε καθορίσει.

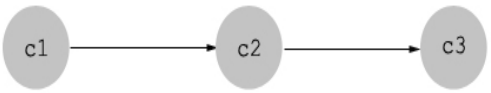
5.2.4 Επιλογή από το γράφημα με κατά πλάτος διάσχιση (SFG- select from graph)

Με τη μέθοδο αυτή παράγουμε ένα γράφημα και το διασχίζουμε από τον τελευταίο κόμβο με κατά πλάτος διάσχιση αφού κάνουμε αντιστροφή των ακμών του. Στο γράφημα οι κύβοι αναπαρίστανται με τους κόμβους.

Δύο κόμβοι `c1`, `c2` ενώνονται μεταξύ τους αν

- ο `c1` απαντά τον `c2` και
- ο `c1` δεν απαντά κάποιον άλλον τρίτο `c` κόμβο ο οποίος απαντά τον `c2`.

Έστω για παράδειγμα ότι έχουμε τους κόμβους `c1, c2` και `c3` και δημιουργούν το παρακάτω γράφημα.



Σχήμα 5.1 Παράδειγμα δημιουργίας γραφήματος με κύβους

Στο παραπάνω γράφημα ο κύβος `c3` απαντιέται και από τον `c1` και από τον `c2` αλλά ενώνεται μόνο με τον `c2` ο οποίος δεν απαντά κάποιον άλλον, ο οποίος απαντά τον `c3` σε αντίθεση με τον κύβο `c1`, ο οποίος απαντά τον κύβο `c3`, αλλά απαντά και τον `c2`, ο οποίος απαντά τον `c3`. Μετά τη δημιουργία του γραφήματος των υλοποιημένων κύβων μαζί με το νέο κύβο, ξεκινάμε να εξετάζουμε τους κόμβους, κάνοντας αντιστροφή των ακμών και διασχίζοντάς τους κατά πλάτος ξεκινώντας από τον νέο κύβο.

5.3 Εύρεση κόστους υπολογισμού κύβου από άλλον κύβο

Για τον υπολογισμό του κόστους από κάθε υποψήφιο κύβο χρησιμοποιούμε τη φόρμουλα κόστους του αλγορίθμου `block nested loops`, που είναι ο αλγόριθμος που

χρησιμοποιεί η MySql για να κάνει σύζευξη. Συγκεκριμένα η MySql έχει διαθέσιμο χώρο αποθήκευσης στη μνήμη 128 KB από προεπιλογή, το μέγεθος σελίδας του InnoDB είναι 16KB άρα έχουμε 8 σελίδες διαθέσιμες για χρήση. Η μία θα αποτελεί είσοδο για τον εξωτερικό πίνακα, η άλλη για την έξοδο και μένουν 6 διαθέσιμες σελίδες για τον εσωτερικό πίνακα. Για τον υπολογισμό του κόστους χρειάζεται να γνωρίζουμε το πλήθος των σελίδων που διαιρείται κάθε πίνακας. Για να το υπολογίσουμε αυτό, αρκεί να βρούμε το μέγεθός του σε bytes και να διαιρέσουμε με το 2^{14} που είναι το μέγεθος σελίδας. Έπειτα υπολογίζουμε το κόστος από τον τύπο |outer|+|inner|*outer|/6. Να σημειωθεί ότι για κάθε υποψήφιο κύβο θα υπολογίσουμε δύο φορές αυτό τον τύπο, εκ των οποίων την πρώτη ο κύβος θα αποτελεί τον εξωτερικό πίνακα και τη δεύτερη θα αποτελεί τον εσωτερικό. Τελικά κόστος για τον υποψήφιο αυτό κύβο θα αποτελεί το μικρότερο από τα κόστη που υπολογίστηκαν με τους δύο αυτούς τρόπους.

5.4 Αλγόριθμος επιλογής κύβου

Στον παρακάτω πίνακα παρουσιάζεται σε ψευδογλώσσα ο αλγόριθμος επιλογής του κύβου που θα χρησιμοποιηθεί για να απαντήσουμε τον νέο κύβο. Ακολουθεί ανάλυση του εν λόγω αλγορίθμου κατά γραμμή.

Πίνακας 5.2 Αλγόριθμος Επιλογής Κύβου

```
Αλγόριθμος Επιλογής Κύβου
Είσοδος: Κύβος προς απάντηση c_new, και ένα διάνυσμα candidates με υποψήφιους
υλοποιημένους κύβους
Εξοδος: Ο βέλτιστος κύβος
1. needsjoin=false;
2. for all c_iold (φ_iold,[L_i1old,L_i2old,…L_inold],agg) (i=1,..,candidates.size())
3. {   for all atoms anew in φnew
4.     {      if φ_iold.contains(a_new)
5.                φ_new.remove(a_new);
6.            else
7.             { if (a_new(Level).isAncestor(L_ijold)) //όπου j πρέπει να είναι το
αντίστοιχο επίπεδο
8.                   needsjoin=true;
9.            }
10.    }
11.    if(!needsjoin)
12.    {     for all levels L_jnew
13.          { if (L_jnew.isAncestor(L_ijold))
14.              {        needsjoin=true;
15.                       break;
16.              }
17.          }
18.    }
19. }
20. if(!needsjoin)
21.    return c_iold with the min(size);
22. else
23. {    for all c_iold in candidates
24.         compute_cost_join;
25.      return c_iold with the min(cost_join);
26.}
```

Ο αλγόριθμος επιλογής κύβου δέχεται ως είσοδο έναν κύβο c_{new}, και ένα διάνυσμα με κάποιους υποψήφιους υλοποιημένους κύβους οι οποίοι τον απαντούν. Ο αλγόριθμος επιστρέφει ως έξοδο τον κύβο, ο οποίος απαντά τον κύβο c_{new} με το λιγότερο κόστος.
Η γενική ιδέα του αλγορίθμου είναι ότι αν από όλους τους υποψήφιους κύβους δεν χρειάζεται από κανέναν να γίνει σύζευξη με κάποιον πίνακα τότε ο αλγόριθμος να επιστρέφει τον μικρότερο από τους υποψήφιους αυτούς κύβο. Αν όμως έστω κι από

έναν χρειάζεται να γίνει σύζευξη για να υπολογιστεί ο νέος αυτός κύβος τότε με κάποια φόρμουλα κόστους να υπολογιστεί ένα κόστος για κάθε υποψήφιο κύβο και στη συνέχεια να επιλεχθεί ο κύβος με το χαμηλότερο κόστος.

Στις γραμμές 1-18 ο αλγόριθμος προσπαθεί να αποφασίσει για κάθε κύβο αν χρειάζεται να γίνει σύζευξη. Συγκεκριμένα, στις γραμμές 2-5 ο αλγόριθμος ελέγχει για κάθε άτομο που υπάρχει στη συνθήκη επιλογής του c_{new} αν υπάρχει στη συνθήκη επιλογής του υποψήφιου κύβου και αν ναι το αφαιρεί. Αυτό το κάνει γιατί όταν έχουμε κάποιο ίδιο άτομο στη συνθήκη επιλογής δεν χρειάζεται να το υπολογίσουμε στην πρόταση "where" του sql ερωτήματος, άρα δεν πρόκειται αυτό να αποτελέσει λόγο για κάποια σύζευξη. Στη συνέχεια (γραμμές 7-9), ο αλγόριθμος ελέγχει αν το επίπεδο που είναι ορισμένο κάθε άτομο της συνθήκης επιλογής του c_{new} είναι πιο λεπτομερές από το επίπεδο που είναι ορισμένος ο υποψήφιος κύβος. Αν δεν είναι πιο λεπτομερές τότε χρειάζεται να γίνει σύζευξη. Αφού τελειώσει με τον έλεγχο των ατόμων στη συνθήκη επιλογής του c_{new}, αν προς το παρόν δεν χρειάζεται από τα προηγούμενα να γίνει σύζευξη, ο αλγόριθμος ελέγχει μήπως χρειάζεται να γίνει σύζευξη εξαιτίας των επιπέδων που είναι ορισμένοι οι κύβοι. Στις γραμμές 12-15 ελέγχει αν τα αντίστοιχα επίπεδα του κύβου cnew είναι πιο ψηλά ορισμένα στην ιεραρχία από του υποψήφιου κύβου. Αν ναι, αυτό σημαίνει ότι χρειάζεται να γίνει σύζευξη και πλέον ο αλγόριθμος σταματά να κάνει έλεγχο για κάθε κύβο. Σε περίπτωση που δεν βρεθεί κάποιος κύβος ο οποίος να χρειάζεται να συζευχθεί με κάποιον πίνακα για να υπολογίσει τον νέο κύβο τότε επιστρέφεται ο κύβος με το μικρότερο μέγεθος (γραμμή 20). Αν όμως είχε βρεθεί κύβος ο οποίος να χρειάζεται σύζευξη για τον υπολογισμό του c_{new}, τότε ο αλγόριθμος υπολογίζει το κόστος για τον κάθε κύβο και στη συνέχεια επιστρέφει τον κύβο με το μικρότερο κόστος (γραμμές 22-24).

5.5 Πειραματική μελέτη του αλγορίθμου επιλογής κύβου

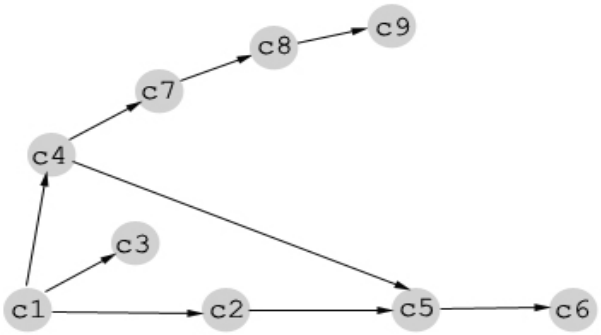
Για την μελέτη των αποτελεσμάτων του αλγορίθμου επιλογής κύβου χρησιμοποιήσαμε του κύβους του προηγούμενου κεφαλαίου της παραγράφου μελέτης καταλληλότητας κύβου για την περίπτωση `Level0Value`. Οι ορισμοί τους αναφέρονται ξανά στον παρακάτω πίνακα.

Πίνακας 5.3 Ορισμοί κύβων για μελέτη αλγορίθμου επιλογής κύβου

1	c1=(date>=2451180,[date,warehouse,brand],"sum")	742028
2	c2=(date>=2451180,[year,country,brand],"sum")	25874
3	c3=([date>=2451180,date<=2451276],[date,warehouse,brand],"sum")	26888
4	c4=([date>=2451545,city>=1110,city<=1120],[month,city,brand],"sum")	16875
5	c5=([date>=2451545,country=1000],[year,country,brand],"sum")	202018
6	c6=([date>=2451545,country=1000,brand>=50,brand<=12302],[year,country,brand],"sum")	11487
7	c7=([date>=2451911,city>=1110,city<=1120],[month,city,brand],"sum")	134764
8	c8=([date>=2451911,date<=2452122,city>=1110,city<=1120],[month,city,brand],"sum")	39273
9	c9=([date>=2451911,date<=2452122,country=1000,brand>=70,brand<=12302],[month,city,brand],"sum")	26689

5.5.1 Μελέτη χρόνου αλγορίθμου επιλογής κύβου

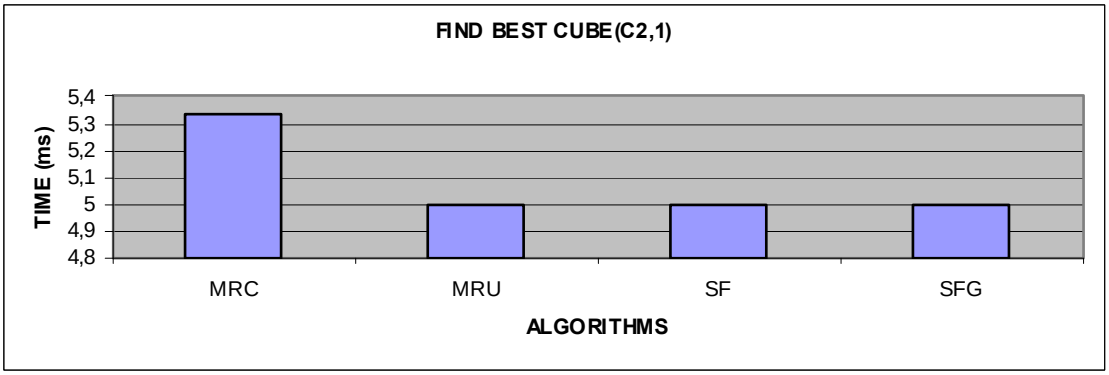
Ακολουθεί η ανάλυση χρόνου εκτέλεσης του αλγορίθμου επιλογής κύβου χρησιμοποιώντας και τις τέσσερις προαναφερθείσες μεθόδους δημιουργίας του διανύσματος υποψήφιων κύβων. Ως κατώφλι χρησιμοποιήθηκε το πλήθος των υποψήφιων κόμβων, δηλαδή το μέγεθος του διανύσματος candidates. Κατά τη δημιουργία των κύβων αυτών παράγεται το γράφημα του σχήματος 5.2 και είναι απαραίτητο για τη μελέτη της τελευταίας μεθόδου δημιουργίας διανύσματος υποψήφιων κόμβων.



Σχήμα 5.2 Γράφημα κύβων

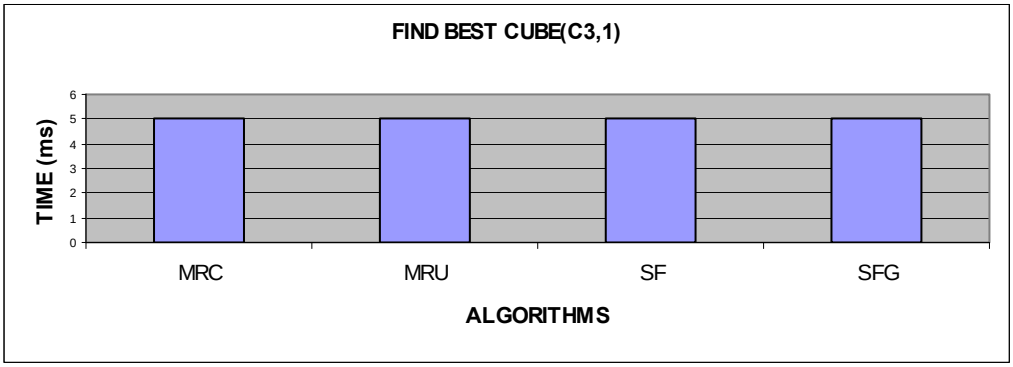
Στον άξονα τον x βρίσκονται οι μέθοδοι επιλογής των υποψήφιων κύβων, MRC, MRU, SF και SFG και στον άξονα τον y αναπαρίσταται ο χρόνος σε ms. Στο Σχήμα

5.3 βλέπουμε το χρόνο επιλογής κύβου για τον κύβο **c2**. Οι μετρήσεις έγιναν και με τις τέσσερις τεχνικές οι οποίες δεν παρουσιάζουν διαφορά και για τιμή κατωφλιού 1. Δεν εκτελέσαμε τον αλγόριθμο για άλλες τιμές κατωφλιού μιας και ο κύβος **c2** έχει μόνο έναν υλοποιημένο κύβο πριν από αυτόν. Ο βέλτιστος κύβος για να απαντήσει τον κύβο **c2** είναι φυσικά ο κύβος c1, εφόσον δεν υπάρχει άλλος πριν από αυτόν.



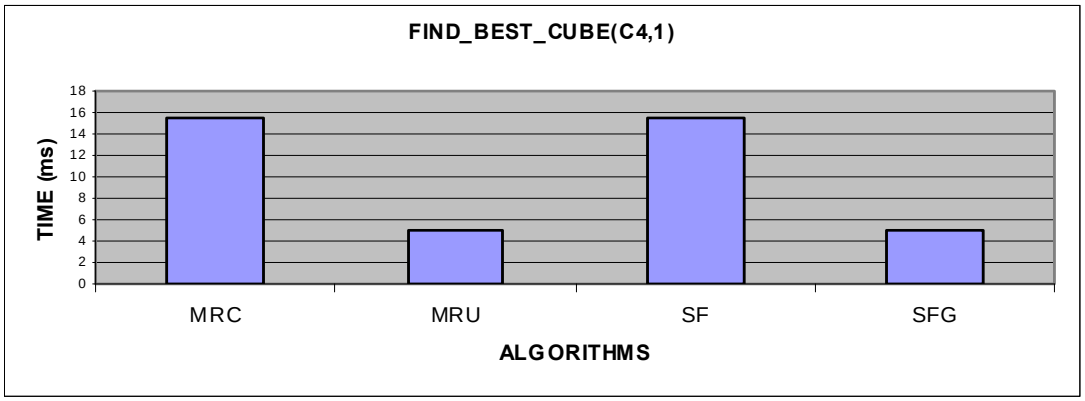
Σχήμα 5.3 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου c2

Στο Σχήμα 5.4 παρουσιάζονται οι χρόνοι εκτέλεσης του αλγορίθμου επιλογής κύβου για την απάντηση του κύβου **c3**. Χρησιμοποιήθηκε τιμή κατωφλιού 1 μιας και ο αλγόριθμος **c3** απαντιέται μόνο από τον c1 και ο c1 είναι ο τελευταίος κύβος ο οποίος εξετάζεται από τους αλγορίθμους διάσχισης κύβων, αν όχι ο μοναδικός στην περίπτωση της επιλογής από το γράφημα. Δεν παρουσιάζονται διαφορές στον χρόνο εκτέλεσης του αλγορίθμου, ανάμεσα στις διάφορες τεχνικές.



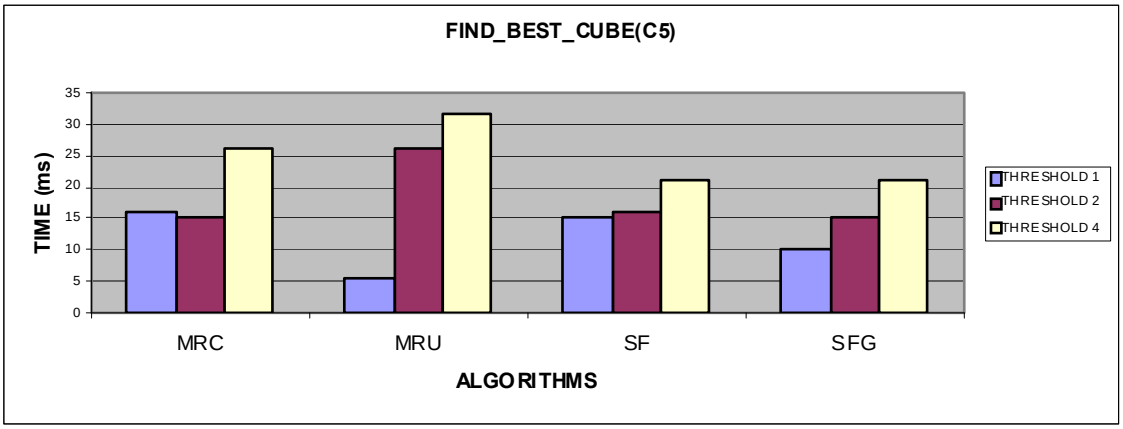
Σχήμα 5.4 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου c3

Στο Σχήμα 5.5 παρουσιάζεται ο χρόνος εκτέλεσης του αλγορίθμου με τιμή κατωφλιού 1 για απάντηση του κύβου **c4.** Ο λόγος για τον οποίο δεν εξετάζονται άλλες τιμές κατωφλιού είναι ο ίδιος με παραπάνω.



Σχήμα 5.5 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου c4

Δεν βλέπουμε κάποια σημαντική διαφορά στους χρόνους εφόσον ο χρόνος μέτρησης είναι σε χιλιοστά του δευτερολέπτου. Και με τις τέσσερις τεχνικές εύρεσης υποψήφιων κύβων, ο αλγόριθμος βρίσκει το σωστό βέλτιστο κύβο τον c1, ο οποίος είναι και ο μοναδικός.



Σχήμα 5.6 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου c5

Ο αλγόριθμος επιλογής κύβου εκτελέστηκε για τιμές κατωφλιού 1,2 και 4 για τον κύβο **c5**.

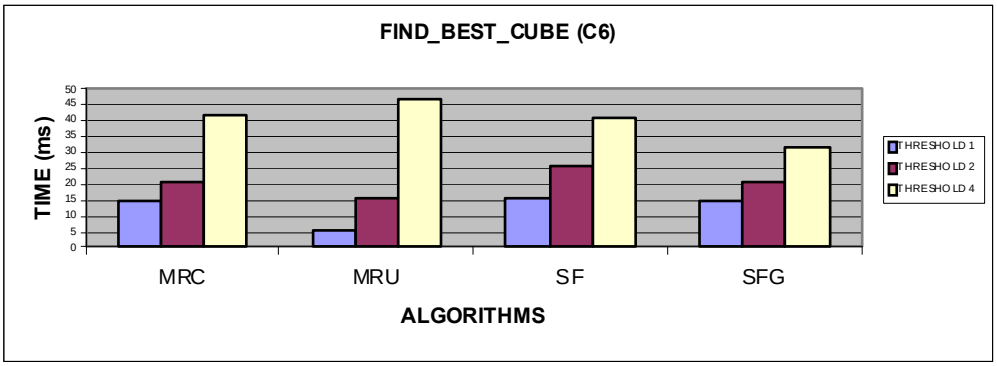
- Για τιμή κατωφλιού 1 ο αλγόριθμος MRC θα υπολογίσει αμέσως αν ο c4 απαντά τον **c5** που αυτό ισχύει και αμέσως θα σταματήσει. Ο MRU

θα ελέγξει το ποιος απάντησε τον c4, θα βρει ότι τον απάντησε ο c1 και θα σταματήσει. Ο SF θα ελέγξει τον c4 που είναι ο πιο μικρός από τους υπάρχοντες και θα τον επιστρέψει. Ο SFG θα επιστρέψει τον c4. Η σειρά επιλογής είναι τυχαία στην περίπτωση αυτή.

- Για τιμή κατωφλιού 2 ο MRC θα έχει ως υποψήφιους κύβους τον c4 και τον c2 για τους οποίους πρέπει να υπολογίσει το κόστος από την φόρμουλα, αφού ο **c5** για να υπολογιστεί από τον c4 πρέπει να κάνει σύζευξη. Τελικά θα επιστρέψει τον κύβο c2. Ο MRU θα εισάγει πρώτο στο διάνυσμα υποψήφιων κύβων, τον κύβο c1, ο οποίος είναι αυτός που απάντησε τον c4. Στη συνέχεια θα τους εισάγει από το τέλος προς την αρχή, σύμφωνα με την εμφάνισή τους. Άρα θα εισάγει τον κύβο c4 εφόσον έχουμε τιμή κατωφλιού 2. Από τους δύο κύβους αυτούς ο αλγόριθμος επιλέγει τον κύβο c4 ως βέλτιστο. Αυτό έχει λογική εφόσον και για τους δύο χρειάζονται συζεύξεις για να υπολογίσουν τον κύβο c5, αλλά ο κύβος c4 είναι πιο μικρός από τον c1. Ο SF θα εισάγει στο διάνυσμα τον κύβο c4 και τον κύβο c2 που είναι οι μικρότεροι που απαντούν τον κύβο **c5** και ο αλγόριθμος θα επιστρέψει τον κύβο c2. Ο αλγόριθμος SFG θα εισάγει κι αυτός στο διάνυσμα υποψήφιων κύβων τους c2 και c4 και από τους δύο θα επιστρέψει τον c2, που η φόρμουλα υπολογισμού κόστους επέστρεψε το μικρότερο κόστος.
- Για τιμή κατωφλιού 4, βλέπουμε ότι ο αλγόριθμος εξακολουθεί και επιστρέφει το ίδιο αποτέλεσμα και με τις τέσσερις τεχνικές, αλλά παρατηρούμε μία μικρή αύξηση στο χρόνο κι αυτό είναι λογικό γιατί ο αλγόριθμος, με το που θα βρει δύο υποψήφιους κύβους, δεν θα σταματήσει αλλά θα συνεχίσει να αναζητά και να ελέγχει αν υπάρχουν κάποιοι άλλοι κύβοι που να απαντούν τον **c5**.

Συνολικά για τις μετρήσεις που έγιναν για τον κύβο **c5** παρατηρούμε ότι για τιμή κατωφλιού 1, ο αλγόριθμος δεν μας βγάζει τον καλύτερο κύβο και κυρίως η τεχνική MRU επιστρέφει τον κύβο, ο οποίος θα φέρει το χειρότερο αποτέλεσμα. Από τιμή κατωφλιού 2 και άνω το αποτέλεσμα σταθεροποιείται, με όποια τεχνική κι αν εκτελεστεί ο αλγόριθμος και για τιμή 4 έχουμε αύξηση της καθυστέρησης επιλογής

του κύβου. Σύμφωνα και με τις μετρήσεις που έγιναν στο προηγούμενο κεφάλαιο, βλέπουμε ότι ο αλγόριθμος επιλογής κύβου σωστά επιστρέφει τον κύβο c2, ως τον καλύτερο κύβο για να απαντήσει τον κύβο **c5**.

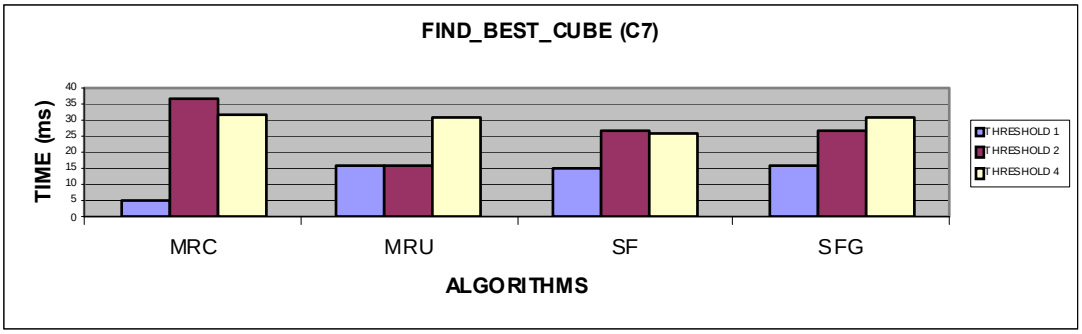


Σχήμα 5.7 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου c6

Στο Σχήμα 5.7 φαίνονται οι χρόνοι εκτέλεσης του αλγορίθμου επιλογής κύβου, για τον κύβο **c6** χρησιμοποιώντας και τις τέσσερις τεχνικές για κατώφλι 1,2 και 4.

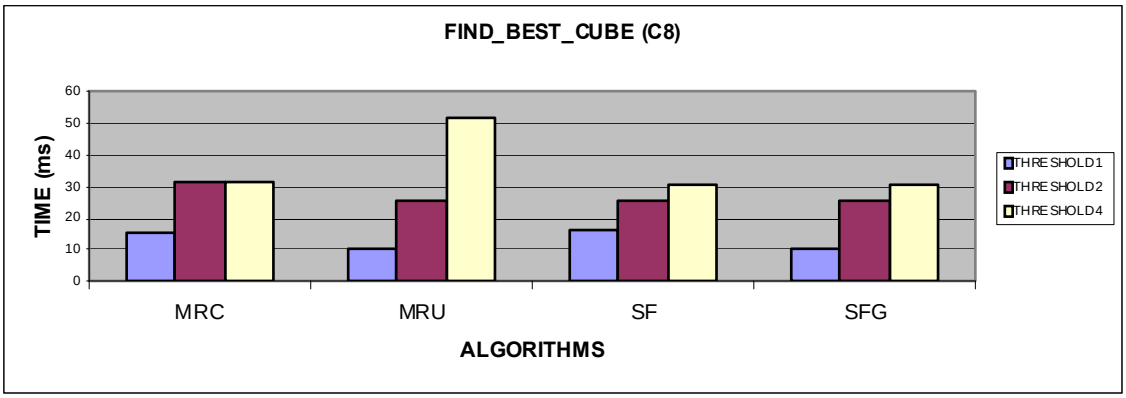
- Για τιμή κατωφλιού 1, ο MRC επιστρέφει τον κόμβο c5, ο MRU τον c4, τον οποίο τον επέλεξε τυχαία ανάμεσα σε αυτόν και τον c2, ο SF επιστρέφει τον c4, ο οποίος είναι ο πιο μικρός κύβος μέχρι στιγμής και ο SFG επιστρέφει τον c5.
- Για τιμή κατωφλιού 2, ο SFG έχει εισάγει στο διάνυσμα candidates τους κύβους c5 και c4, στη συνέχεια υπολογίζει τα κόστη τους από την φόρμουλα κόστους και επιστρέφει τον κύβο c5, ο οποίος είναι και αυτός που δεν χρειάζεται να γίνει καμία σύζευξη για να υπολογιστεί ο κύβος **c6**. Ο SF εισάγει στο διάνυσμα τους κύβους c4 και c2 και από τους δύο αυτούς επιστρέφει τον c2. Ο SFG εισάγει στο διάνυσμα τους κύβους c5 και c4 και επιστρέφει τον c5.
- Τέλος για τιμή κατωφλιού 4, όλοι επιστρέφουν τον c2.

Βλέπουμε ότι ο αλγόριθμος σωστά υπολογίζει ξανά ως βέλτιστο κύβο τον σωστό που είναι ο c2 για τιμή κατωφλιού 2 και άνω, εκτός από όταν χρησιμοποιήθηκε ο SFG για τιμή 2 που επέστρεψε τον c5 που δίνει επίσης γρήγορη απάντηση για τον **c6**. Παρατηρούμε ακόμη ότι οι χρόνοι έχουν αυξηθεί κι αυτό οφείλεται στο ότι ο αλγόριθμος κάνει αναζήτηση σε μεγαλύτερο πεδίο τιμών. Να σημειωθεί ότι σε περίπτωση που υπολογίσει για δύο κύβους ίδιο κόστος επιστρέφει κάποιον τυχαία.



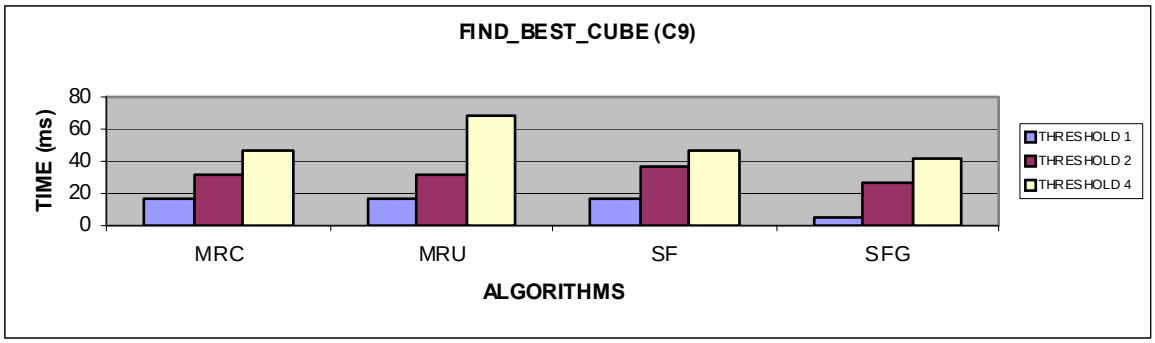
Σχήμα 5.8 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου `c7`

Στο Σχήμα 5.8 παρουσιάζονται οι μετρήσεις του αλγορίθμου επιλογής κύβου για τον υπολογισμό του κύβου **c7**, χρησιμοποιώντας και τις τέσσερις τεχνικές με τιμές κατωφλιού 1,2 και 4. Για όλες τις τιμές κατωφλιού με τη χρήση όλων των τεχνικών ο αλγόριθμος επιστρέφει σε όλες τις περιπτώσεις τον κύβο `c4`. Το διάνυσμα candidates στην περίπτωση αυτή είχε μόνο δύο υποψήφιους κύβους, τον `c4` και τον `c1`. Σωστά ξανά ο αλγόριθμος επέστρεψε τον `c4`.



Σχήμα 5.9 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου `c8`

Ο αλγόριθμος υπολογισμού επιλογής κύβου στην περίπτωση του κύβου **c8** επιστρέφει τον κύβο `c7` ή τον κύβο `c4`. Η επιλογή είναι ανάλογα με το πως θα τους συναντήσει μέσα στο διάνυσμα. Το κόστος που υπολόγισε για τους δύο αυτούς κύβους είναι το ίδιο και όπως βλέπουμε και από τις μετρήσεις του προηγούμενου κεφαλαίου, οι δύο αυτοί κύβοι απαντούν σχεδόν στον ίδιο χρόνο τον κύβο **c8**. Επίσης να αναφέρουμε, ότι το διάνυσμα υποψήφιων κύβων περιείχε τους κύβους `c1`, `c4` και `c7`.



Σχήμα 5.10 Χρόνος εκτέλεσης αλγορίθμου επιλογής κύβου για απάντηση του κύβου `c9`

Ο αλγόριθμος σε όλες τις περιπτώσεις υπολογίζει σωστά τον κύβο `c8` ως τον καλύτερο που πρόκειται να απαντήσει τον **`c9`** πιο γρήγορα, εκτός από τις περιπτώσεις χρήσης του MRU για τιμή κατωφλιού 1 και 2 και του SF για τιμή κατωφλιού 1, που δεν συμπεριλαμβάνεται στο διάνυσμα υποψήφιων κύβων ο κύβος `c8`.

Συμπέρασμα: Ο αλγόριθμος σε όλες τις περιπτώσεις που μελετήσαμε για τιμή κατωφλιού 4 υπολογίζει κι επιστρέφει τον βέλτιστο κύβο. Δεν διακρίνεται κάποια συγκεκριμένη τεχνική για την ταχύτητά της ή για την εγκυρότητα των αποτελεσμάτων της. Το μόνο που θα μπορούσαμε να πούμε είναι, ότι η τεχνική MRU για μικρές τιμές κατωφλιού δεν υπολογίζει τον βέλτιστο κύβο τις περισσότερες φορές. Αυτό οφείλεται στο γεγονός, ότι συνήθως βέλτιστος κύβος είναι ο ακριβώς προηγούμενος στο γράφημα από τον νέο κύβο που θέλουμε να υπολογίσουμε και η τεχνική αυτή για να τον συμπεριλάβει πρέπει να έχει πιο μεγάλες τιμές κατωφλιού μιας και είναι ο κύβος συνήθως που υπολογίσαμε τελευταίο. Για τον χρόνο εκτέλεσης του αλγορίθμου έχουμε να πούμε, ότι όσο αυξάνεται το κατώφλι αυξάνεται και ο χρόνος εκτέλεσης αυτού κι αυτό είναι λογικό, αφού η αναζήτηση για νέους υποψήφιους κύβους και η κλήση του αλγορίθμου καταλληλότητας κύβων συνεχίζει να εκτελείται.

5.5.2 Μελέτη βελτίωσης εκτέλεσης ερωτήματος μετά τη χρήση του αλγορίθμου επιλογής κύβου.

5.5.2.1Μελέτη περίπτωσης για εκτέλεση του ερωτήματος με χρήση της τεχνικής MRC

Στους παρακάτω πίνακες παρουσιάζονται οι χρόνοι υπολογισμού των κύβων από τον πίνακα πληροφοριών, από τον βέλτιστο κύβο, ποιος είναι ο βέλτιστος κύβος, και για τιμές κατωφλιού 1,2 και 4 παρουσιάζονται, οι κύβοι c που επέστρεψε ο αλγόριθμος επιλογής κύβου, σε πόσο χρόνο τον επέστρεψε, πόσος χρόνος χρειάζεται για να υπολογιστεί ο κύβος από τον κύβο που επέστρεψε ο αλγόριθμος, την ποιότητα της λύσης σε σχέση με τον βέλτιστο (χειροτέρευση) και την ποιότητα της λύσης σε σχέση με τον πίνακα πληροφοριών (fact table).

Πίνακας 5.4 Μετρήσεις με τεχνική MRC και κατώφλι 1

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5,333333	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	15,66667	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C4	16	0,38976	141,28	0,745415
C6	1,23828	0,00372	C2	C5	15	0,00666	0,790323	0,994622
C7	1,34432	0,00044	C4	C4	5	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	15,66667	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	15,66667	0,06896	0	0,752885

Ο παραπάνω πίνακας ήταν για τιμή κατωφλιού 1. Παρατηρούμε ότι για τον κύβο `c5`, ο αλγόριθμος επιστρέφει ως βέλτιστο τον `c4` που είναι ο ακριβώς προηγούμενός του. Αυτό είναι λογικό, γιατί εφόσον η τιμή του κατωφλιού είναι 1 και η τεχνική αυτή διασχίζει τους κύβους σύμφωνα με το χρόνο άφιξής τους θα ελέγξει πρώτα τον κύβο `c4`, ο οποίος είναι ο τελευταίος κύβος που έχει φτάσει και θα τον επιστρέψει. Ο χρόνος υπολογισμού είναι κατά 14128% μεγαλύτερος από του βέλτιστου και η βελτίωση σε σχέση με τον πίνακα πληροφοριών είναι 79%. Ο αλγόριθμος για τον

κύβο c6 επιστρέφει τον c5 για τον ίδιο λόγο με την προηγούμενη περίπτωση, ο χρόνος υπολογισμού σε σχέση με τον βέλτιστο είναι 79 φορές μεγαλύτερος και έχουμε βελτίωση υπολογισμού σε σχέση με τον πίνακα πληροφοριών 99,4%. Ο πίνακας που ακολουθεί περιέχει τις μετρήσεις για τιμή κατωφλιού 2.

Πίνακας 5.5 Μετρήσεις με τεχνική MRC και κατώφλι 2

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	16	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	15,33333	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C5	20,66667	0,00666	0,790323	0,994622
C7	1,34432	0,00044	C4	C4	36,66667	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	31,33333	0,0061	0	0,983318
C9	0,27906	0,06896	C8	C8	31,66667	0,06896	0	0,752885

Για τιμή κατωφλιού 2 ο αλγόριθμος υπολογίζει μόνο στην περίπτωση του κύβου c6 διαφορετικό κύβο από τον βέλτιστο. Στην περίπτωση αυτή στο διάνυσμα υποψήφιων κύβων υπάρχουν οι κύβοι c4 και c5. Έχουμε 99,4% βελτίωση σε σχέση με τον πίνακα πληροφοριών και υπολογίζεται από τον κύβο c5 κατά 79% πιο αργά σε σχέση με τον βέλτιστο.

Για τιμή κατωφλιού 4 έγιναν οι μετρήσεις που παρουσιάζονται παρακάτω:

Πίνακας 5.6 Μετρήσεις με τεχνική MRC και κατώφλι 4

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5,333333	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5,333333	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	15,66667	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	26	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C2	42	0,00372	0	0,996996
C7	1,34432	0,00044	C4	C4	31,66667	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	31,33333	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	47	0,06896	0	0,752885

Παρατηρούμε ότι σε όλες τις περιπτώσεις ο αλγόριθμος επιστρέφει πάντα τον βέλτιστο κύβο.

Συμπέρασμα: Συνολικά παρατηρούμε ότι στις περισσότερες περιπτώσεις υπολογίζεται ο βέλτιστος κύβος. Στην περίπτωση για τιμή κατωφλιού 1 κυρίως ο αλγόριθμος δεν υπολογίζει πάντα τον βέλτιστο κύβο. Στις έξι περιπτώσεις δηλαδή ο βέλτιστος κύβος είναι ακριβώς ο προηγούμενος που δημιουργήθηκε. Τέλος για τιμή κατωφλιού 4 υπολογίζει πάντα τον βέλτιστο κύβο.

5.5.2.2 Μελέτη περίπτωσης για εκτέλεση του ερωτήματος με χρήση της τεχνικής MRU

Στους παρακάτω πίνακες παρουσιάζονται τα αποτελέσματα για την εκτέλεση του αλγορίθμου με την τεχνική προσπέλασης κύβων MRU. Οι μετρήσεις στους πίνακες είναι με τιμές κατωφλιού 1,2 και 4.

Πίνακας 5.7 Μετρήσεις με τεχνική MRU και κατώφλι 1

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	5	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C1	5,333333	0,96896	352,635	0,36709
C6	1,23828	0,00372	C2	C4	5,333333	0,31386	83,37097	0,746536
C7	1,34432	0,00044	C4	C4	16	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C4	10,33333	0,0061	0,101083	0,981632
C9	0,27906	0,06896	C8	C4	15,66667	0,08306	0,204466	0,702358

Παρατηρούμε ότι για τιμή κατωφλιού 1, ο αλγόριθμος με χρήση της τεχνικής MRU στις 4 από τις 8 περιπτώσεις εξέτασης δεν επέστρεψε τον βέλτιστο κύβο. Στην περίπτωση του κύβου `c5` επιστρέφει τον κύβο `c1` που είναι 35263 φορές πιο αργός από τον βέλτιστο και επιφέρει 36% βελτίωση υπολογισμού σε σύγκριση με τον πίνακα πληροφοριών. Για τον κύβο `c6` επιστρέφει τον κύβο `c4` που είναι 8337% πιο

αργός του βέλτιστου και επιφέρει 74% βελτίωση σε σχέση με τον πίνακα πληροφοριών. Για τους κύβους c8 και c9, επιστρέφει τον κύβο c4 που είναι 10% και 20% πιο αργός των βέλτιστων κι επιφέρει 98% και 70% αντίστοιχα βελτίωση.
Στον παρακάτω πίνακα παρουσιάζονται οι μετρήσεις που έγιναν για τον αλγόριθμο εύρεσης βέλτιστου κύβου για τιμή κατωφλιού 2.

Πίνακας 5.8 Μετρήσεις με τεχνική MRU και κατώφλι 2

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5,333333	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	10,33333	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	26	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C2	15,33333	0,00372	0	0,996996
C7	1,34432	0,00044	C4	C4	16	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C4	26	0,0061	0,101083	0,981632
C9	0,27906	0,06896	C8	C4	31,33333	0,08306	0,204466	0,702358

Παρατηρούμε ότι σε όλες τις περιπτώσεις ο αλγόριθμος επιστρέφει τον βέλτιστο κύβο εκτός από τους κύβους c8 και c9 που επιστρέφει τον c4. Παρόλ' αυτά στην περίπτωση του κύβου c8 έχουμε βελτίωση 98% σε σχέση με τον χρόνο υπολογισμού του κύβου c8 από τον πίνακα πληροφοριών. Στην περίπτωση για τον κύβο c9 έχουμε βελτίωση 70%.
Παρακάτω παρουσιάζονται οι μετρήσεις που έγιναν για τιμή κατωφλιού 4.

Πίνακας 5.9 Μετρήσεις με τεχνική MRU και κατώφλι 4

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5,333333	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5,333333	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	20,66667	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	31,66667	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C2	47	0,00372	0	0,996996
C7	1,34432	0,00044	C4	C4	31	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	52	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	67,66667	0,06896	0	0,752885

Στην περίπτωση εκτέλεσης του αλγορίθμου με τιμή κατωφλιού 4 υπολογίζεται σε όλες τις περιπτώσεις ο βέλτιστος κύβος.

Συμπέρασμα: Ο αλγόριθμος επιλογής κύβου με κατώφλι 1 κατά 50% των περιπτώσεων δεν υπολόγισε τον βέλτιστο αλγόριθμο, παρόλ'αυτά οι κύβοι που επέστρεψε επέφεραν αρκετά μεγάλη βελτίωση (70% και άνω) σε σχέση με τον πίνακα πληροφοριών. Για μεγάλες τιμές κατωφλιού ο αλγόριθμος επέστρεψε σε όλες τις περιπτώσεις τον βέλτιστο στο σύνολο των υποψηφίων κύβο, ο οποίος επιφέρει σημαντικές βελτιώσεις στον χρόνο υπολογισμού κάποιου κύβου σε σχέση με τον πίνακα πληροφοριών. Ακόμη ο χρόνος υπολογισμού του βέλτιστου στο σύνολο κύβου από τον αλγόριθμο επιλογής κύβου δεν είναι σημαντικά, άρα με επιφύλαξη θα μπορούσαμε να πούμε ότι προτιμάμε μεγάλες τιμές κατωφλιού εφόσον ο αλγόριθμος δεν έχει σημαντική καθυστέρηση σε σύγκριση με μικρές τιμές κατωφλιού.

5.5.2.3Μελέτη περίπτωσης για εκτέλεση του ερωτήματος με χρήση της τεχνικής SF

Οι πίνακες που ακολουθούν περιλαμβάνουν τις μετρήσεις για την εκτέλεση του αλγορίθμου επιλογής κύβου με χρήση της τεχνικής SF με τιμές κατωφλιού 1,2 και 4.

Πίνακας 5.10 Μετρήσεις με τεχνική SF και κατώφλι 1

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	15,66667	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C1	15	0,96896	352,635	0,36709
C6	1,23828	0,00372	C2	C4	16	0,31386	83,37097	0,746536
C7	1,34432	0,00044	C4	C4	15	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C4	16	0,0061	0,101083	0,981632
C9	0,27906	0,06896	C8	C4	16	0,08306	0,204466	0,702358

Βλέπουμε ότι ο αλγόριθμος με χρήση της τεχνικής SF και τιμή κατωφλιού 1, επιστρέφει τους ίδιους κύβους με τα αποτελέσματα που έδωσε με την τεχνική MRU με τιμή κατωφλιού 1. Ο αλγόριθμος με την τεχνική αυτή, ταξινομεί τους υποψήφιους κύβους ανάλογα με το μέγεθός τους και τους εξετάζει από τον μικρότερο προς τοπ μεγαλύτερο. Εφόσον η τιμή κατωφλιού είναι 1 επιστρέφει τον μικρότερο.

Πίνακας 5.11 Μετρήσεις με τεχνική SF και κατώφλι 2

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	10,33333	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	16	0,00274	0	0,36709
C6	1,23828	0,00372	C2	C2	26	0,00372	0	0,996996
C7	1,34432	0,00044	C4	C4	26,33333	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C4	26	0,0061	0,101083	0,981632
C9	0,27906	0,06896	C8	C8	36,33333	0,06896	0	0,752885

Στον παραπάνω πίνακα παρουσιάζονται οι μετρήσεις για τιμή κατωφλιού 2. Ο αλγόριθμος με τις παραμέτρους αυτές επιστρέφει σε όλες εκτός από μία περίπτωση τον βέλτιστο κύβο. Ο κύβος που επιστρέφει στην περίπτωση αυτή όμως επιφέρει 98%

βελτίωση σε σχέση με τον πίνακα πληροφοριών που είναι πολύ μεγάλη βελτίωση και ελάχιστα μικρότερη (0,4% περίπου χειρότερα) από τον βέλτιστο.

Πίνακας 5.12 Μετρήσεις με τεχνική SF και κατώφλι 4

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	5,333333	1,40334	0	0,3268
C3	0,0826	0,0033	C1	5,333333	0,0033	0	0,960048
C4	1,94676	1,28222	C1	10,33333	1,28222	0	0,341357
C5	1,53096	0,00274	C2	21	0,00274	0	0,99821
C6	1,23828	0,00372	C2	41,33333	0,00372	0	0,996996
C7	1,34432	0,00044	C4	25,66667	0,00044	0	0,999673
C8	0,3321	0,00554	C7	31	0,00554	0	0,983318
C9	0,27906	0,06896	C8	46,33333	0,06896	0	0,752885

Στον παραπάνω πίνακα παρουσιάζονται οι μετρήσεις που έγιναν με τιμή κατωφλιού 4. Σε όλες τις περιπτώσεις ο αλγόριθμος επέστρεψε τον βέλτιστο κύβο.

Συμπέρασμα: Παρατηρούμε ότι ο χρόνος εκτέλεσης του αλγορίθμου στην περίπτωση αυτή είναι μικρότερος σε σχέση με τις υπόλοιπες τεχνικές. Για τιμή κατωφλιού 1 δεν υπολόγισε σε δύο περιπτώσεις τον βέλτιστο κύβο, εκ των οποίων στη μία είχε 98% βελτίωση σε σχέση με τον πίνακα πληροφοριών και στην άλλη 74%. Για τιμή κατωφλιού 2 υπολογίζεται σε όλες εκτός από μία περίπτωση ο βέλτιστος κύβος, που και πάλι όμως στην περίπτωση αυτή υπολογίζεται ένας κύβος, ο οποίος επιφέρει 98% βελτίωση.

5.5.2.4 Μελέτη περίπτωσης για εκτέλεση του ερωτήματος με χρήση της τεχνικής SFG

Στους πίνακες αυτής της παραγράφου παρουσιάζονται οι μετρήσεις εκτέλεσης του αλγορίθμου επιλογής κύβου με χρήση της τεχνικής SFG για τιμές κατωφλιού 1,2 και 4. Ο πίνακας που ακολουθεί αφορά την εκτέλεση του αλγορίθμου με τιμή κατωφλιού 1.

Πίνακας 5.13 Μετρήσεις με τεχνική SFG και κατώφλι 1

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	5	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	10	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C5	15	0,00666	0,790323	0,994622
C7	1,34432	0,00044	C4	C4	15,66667	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	10	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	5	0,06896	0	0,752885

Σε όλες τις περιπτώσεις εκτός από την περίπτωση του κύβου `c6` υπολογίζει σωστά τον βέλτιστο κύβο. Παρόλ'αυτά έχουμε βελτίωση στην περίπτωση του κύβου `c6` 99,4%. Ακόμη οι χρόνοι εκτέλεσης του αλγορίθμου παρουσιάζονται να είναι μικρότεροι από τους χρόνους εκτέλεσης με τις υπόλοιπες τεχνικές. Στον παρακάτω πίνακα παρουσιάζονται οι μετρήσεις που έγιναν για τιμή κατωφλιού 2.

Πίνακας 5.14 Μετρήσεις με τεχνική SFG και κατώφλι 2

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**c**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	5	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	15,33333	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C5	20,66667	0,00666	0,790323	0,994622
C7	1,34432	0,00044	C4	C4	26,33333	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	26	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	26	0,06896	0	0,752885

Τα αποτελέσματα εκτέλεσης του αλγορίθμου με τιμή κατωφλιού 2 είναι τα ίδια ακριβώς με τιμή κατωφλιού 1. Βλέπουμε μόνο ότι ο αλγόριθμος εκτελείται σε

μεγαλύτερο χρόνο από ό,τι για τιμή κατωφλιού 1. Αυτό είναι λογικό, εφόσον ψάχνει για περισσότερους υποψήφιους κύβους.

Πίνακας 5.15 Μετρήσεις με τεχνική SFG και κατώφλι 4

Κύβοι	**Χρόνος fact**	**Χρόνος βέλτιστου**	**Βέλτιστος**	**C**	**Χρόνος εύρεσης c (ms)**	**Χρόνος από c**	**Ποιότητα c vs βέλτιστου**	**Βελτίωση**
C2	2,08458	1,40334	C1	C1	5,333333	1,40334	0	0,3268
C3	0,0826	0,0033	C1	C1	5	0,0033	0	0,960048
C4	1,94676	1,28222	C1	C1	5	1,28222	0	0,341357
C5	1,53096	0,00274	C2	C2	21	0,00274	0	0,99821
C6	1,23828	0,00372	C2	C2	31,33333	0,00372	0	0,996996
C7	1,34432	0,00044	C4	C4	31	0,00044	0	0,999673
C8	0,3321	0,00554	C7	C7	31	0,00554	0	0,983318
C9	0,27906	0,06896	C8	C8	40,66667	0,06896	0	0,752885

Τέλος για τιμή κατωφλιού 4 υπολογίζει πάντα τον βέλτιστο κύβο, όπως και με τις προηγούμενες τεχνικές.

Συμπέρασμα: Παρατηρήσαμε συνολικά ότι με χρήση οποιασδήποτε τεχνικής ο αλγόριθμος για τιμή κατωφλιού 4, υπολογίζει πάντα τον βέλτιστο κύβο. Ακόμη, η τεχνική SFG είχε τα καλύτερα αποτελέσματα από άποψη χρόνου εκτέλεσης αλγορίθμου και πλήθος σωστών απαντήσεων με τιμή κατωφλιού 1. Τα χειρότερα αποτελέσματα προέκυψαν με τη χρήση της τεχνικής MRU για τιμή κατωφλιού 1. Μετά τα αποτελέσματα αυτά, επόμενο βήμα είναι να αναρωτηθούμε αν αυτός ο αλγόριθμος όντως πάντα υπολογίζει τον βέλτιστο κύβο. Ας σκεφτούμε την περίπτωση όπου ένας κύβος για να υπολογιστεί από έναν μικρό σε μέγεθος πίνακα χρειάζεται συζεύξεις και για να υπολογιστεί από έναν μεγάλο σε μέγεθος πίνακα δεν χρειάζεται συζεύξεις. Το κόστος υπολογισμού από τον μικρό πίνακα θα είναι μεγαλύτερο του μηδενός ενώ από τον μεγάλο πίνακα θα είναι μηδέν, εφόσον η φόρμουλα κόστους υπολογίζει το κόστος συζεύξεων. Στην περίπτωση αυτή ο αλγόριθμος θα επιστρέψει ως βέλτιστο κύβο τον μεγάλο κύβο.

ΚΕΦΑΛΑΙΟ 6. ΣΥΜΠΕΡΑΣΜΑΤΑ

Στην εργασία αυτή μελετήθηκε η αποδοτική εκτέλεση OLAP ερωτημάτων με χρήση υλοποιημένων κύβων. Στη σχετική εργασία που μελετήθηκε, όλοι οι αλγόριθμοι και οι ορισμοί που αναφέρονται αφορούν το σχεσιακό σχήμα και κανένας αλγόριθμος δεν υποστηρίζει ιεραρχίες και ερωτήσεις OLAP.

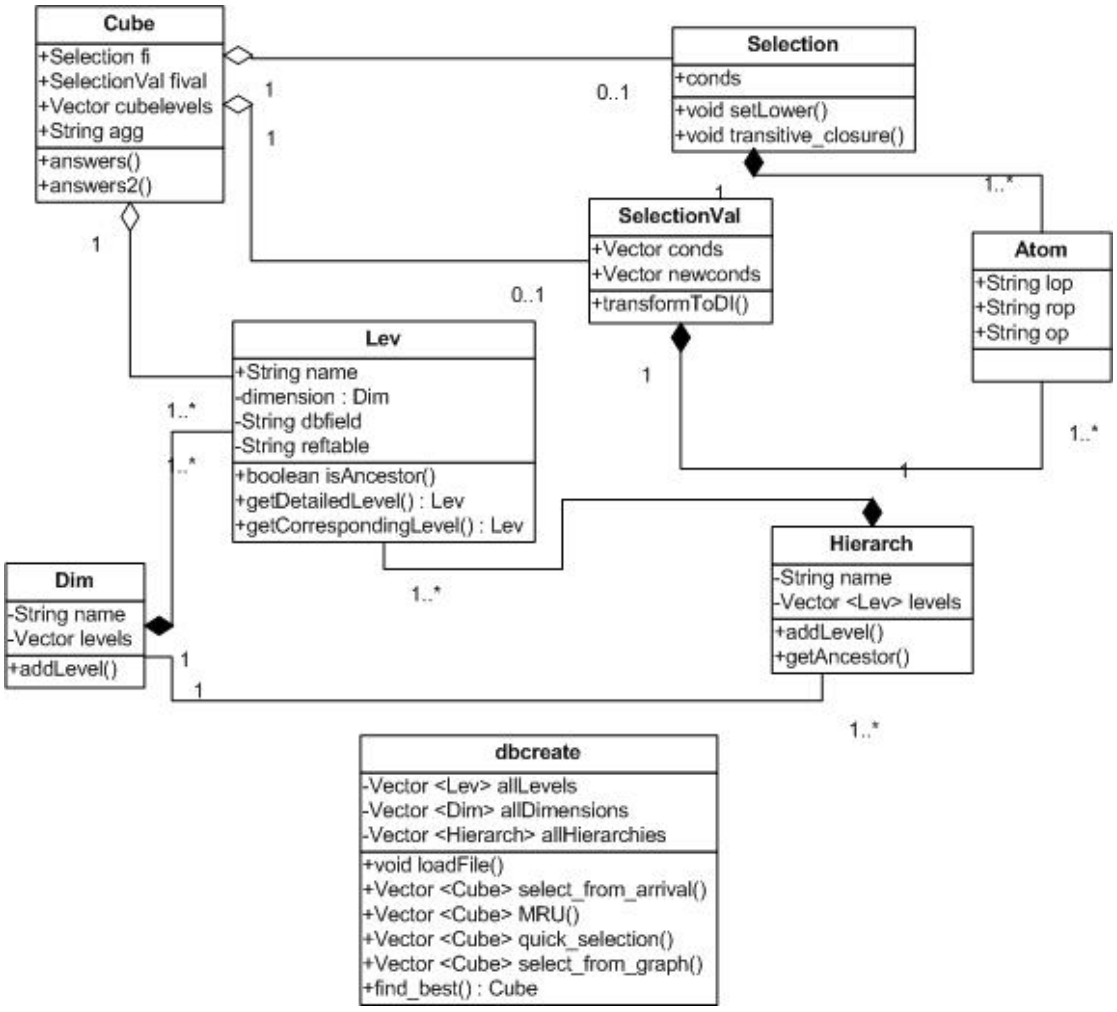
Συγκεκριμένα μελετήθηκαν οι αλγόριθμοι καταλληλότητας κύβου και επαναδιατύπωσης ερωτήματος σύμφωνα με το [VaSk00]. Από την πειραματική μελέτη που διεξήχθη και παρουσιάστηκε αναλυτικά στο κεφάλαιο 4, προέκυψε ο αλγόριθμος εύρεσης βέλτιστου κύβου, ο οποίος δοθέντος ενός διανύσματος κύβων κι ενός ερωτήματος προς απάντηση, επιστρέφει το βέλτιστο κύβο για την απάντηση του ερωτήματος. Ο αλγόριθμος για να αποφανθεί χρησιμοποιεί ένα μοντέλο κόστους, το οποίο λαμβάνει υπόψη το μέγεθος των κύβων και το κόστος των συζεύξεων που γίνονται. Ο αλγόριθμος επιφέρει μεγάλη βελτίωση στο χρόνο εκτέλεσης των ερωτημάτων και τις περισσότερες φορές επιστρέφει και τον μέγιστο κύβο.

Σαν μελλοντική δουλειά μπορεί να μελετηθεί η βελτίωση του αλγορίθμου αυτού ως προς το μοντέλο κόστους που θα χρησιμοποιεί. Συγκεκριμένα, ο αλγόριθμος μπορεί να στηρίζεται σε μία φόρμουλα κόστους, η οποία να υπολογίζει το κόστος συμπεριλαμβάνοντας και το μέγεθος και το κόστος των συζεύξεων.

ΠΑΡΑΡΤΗΜΑ

Υλοποίηση

Οι αλγόριθμοι υλοποιήθηκαν σε γλώσσα προγραμματισμού Java και το DBMS που χρησιμοποιήθηκε είναι ο MySql Server 5.0. Παρακάτω παρουσιάζεται το διάγραμμα κλάσεων σε γλώσσα μοντελοποίησης UML.



Σχήμα 6.1 Διάγραμμα UML

Cube: Η κλάση κύβος. Το αντικείμενο κύβος αποτελείται από τρία ορίσματα. Πρώτο όρισμα είναι η συνθήκη επιλογής η οποία είναι είτε τύπου SelectionVal, είτε τύπου Selection. Το δεύτερο όρισμα είναι ένα διάνυσμα που περιέχει αντικείμενα τύπου Lev, δηλαδή είναι τα επίπεδα στα οποία είναι ορισμένος ο κύβος και θέλουμε να εφαρμόσουμε τη συνθήκη ομαδοποίησης. Τελευταίο όρισμα είναι η συνάρτηση συνάθροισης που είναι τύπου String. Οι κυριότερες μέθοδοι της κλάσης είναι η συνάρτηση answers (Cube c): Boolean και η συνάρτηση answers2 (Cube c): Boolean. Η πρώτη είναι η υλοποίηση του αλγορίθμου καταλληλότητας κύβου για την περίπτωση που η συνθήκη επιλογής είναι τύπου Selection δηλαδή της μορφής LevelθLevel και η δεύτερη για την περίπτωση που η συνθήκη επιλογής είναι τύπου SelectionVal δηλαδή τύπου LevelθValue.

SelectionVal: Η κλάση η οποία ορίζει συνθήκη επιλογής τύπου LevelθValue. Τα αντικείμενα της κλάσης αυτής αποτελούνται από ένα διάνυσμα που περιέχει αντικείμενα τύπου Atom. Η κυριότερη μέθοδος της κλάσης αυτής είναι η transformToDI(), η οποία υπολογίζει τα διαστήματα διάστασης της συνθήκης επιλογής.

Selection: Η κλάση η οποία ορίζει συνθήκη επιλογής τύπου LevelθLevel. Τα αντικείμενα της κλάσης αυτής αποτελούνται από ένα διάνυσμα που περιέχει αντικείμενα τύπου Atom. Η κυριότερη μέθοδος της κλάσης αυτής είναι η transitive_closure(), με την οποία υπολογίζεται η μεταβατική κλειστότητα της συνθήκης επιλογής.

Atom: Η κλάση που ορίζει αντικείμενα άτομα. Αποτελείται από ένα String που χωρίζεται σε τρία μέρη, τον αριστερό τελεσταίο, τον δεξί τελεσταίο και τον τελεστή.

Lev: Η κλάση που ορίζει τα επίπεδα Η κλάση αυτή παράγει αντικείμενα δίνοντας το όνομα του επιπέδου, την διάσταση στην οποία ανήκει το επίπεδο τον πίνακα στον οποίο αναφέρεται και το πεδίο του πίνακα στο οποίο αναφέρεται. Η μέθοδος isAncestor(Lev l): Boolean επιστρέφει αν ένα επίπεδο είναι πρόγονος του επιπέδου l. Η μέθοδος getDetailedLevel() επιστρέφει το λεπτομερές επίπεδο της διάστασης στην οποία ανήκει το επίπεδο το οποίο καλεί την συνάρτηση.

Dim: Η κλάση που ορίζει τις διαστάσεις. Μία διάσταση έχει όνομα και ένα διάνυσμα με επίπεδα.

Hierarch: Η κλάση που ορίζει τις ιεραρχίες. Για να οριστεί μία ιεραρχία χρειάζεται ένα όνομα και ένα διάνυσμα με επίπεδα. Η μέθοδος getAncestor(Lev l) επιστρέφει τον πρόγονο του επιπέδου l που βρίσκεται στη συγκεκριμένη ιεραρχία.
Dbcreate: Η κλάση που περιέχει τη main, τον αλγόριθμος επιλογής κύβου και τις τεχνικές εύρεσης υποψήφιων κύβων.

Ορισμοί κύβων (LevelθLevel)

Κύβος 1:

```
select
inv_date_sk,inv_item_sk,inv_warehouse_sk,inv_date_kati,sum(inv_quantity_on_han
d)
from inventory
where inv_date_sk>inv_date_kati
group by inv_date_sk,inv_item_sk,inv_warehouse_sk,inv_date_kati;
```

Κύβος 2:

```
select D1.mids ,inv_item_sk,inv_warehouse_sk,D2.mids ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.mids,inv_item_sk,inv_warehouse_sk,D2.mids;
```

```
select D1.mids ,inv_item_sk,inv_warehouse_sk,D2.mids ,sum(agg)
from cube1,dates D1,dates D2
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.mids,inv_item_sk,inv_warehouse_sk,D2.mids;
```

Κύβος 3:

```
select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
from cube1,dates D1,dates D2
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
from cube2,months D1,months D2
where d_month_seq1=D1.mids
and d_month_seq2=D2.mids
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;
```

Κύβος 4:

```
select D1.mids ,inv_item_sk,streetid,D2.mids ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2,ware
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
and inv_warehouse_sk=wareid
group by D1.mids,inv_item_sk,streetid,D2.mids;

select D1.mids ,inv_item_sk,streetid,D2.mids ,sum(agg)
from cube1,dates D1,dates D2,ware
where inv_date_sk>inv_date_kati
```

```
and inv_date_sk=D1.did
and inv_date_kati=D2.did
and inv_warehouse_sk=wareid
group by D1.mids,inv_item_sk,streetid,D2.mids;

select d_month_seq1 ,inv_item_sk,streetid,d_month_seq2,sum(agg)
from cube2,ware
where inv_warehouse_sk=wareid
group by d_month_seq1 ,inv_item_sk,streetid,d_month_seq2;
```

Κύβος 5:

```
select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2,ware
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
and wareid=inv_warehouse_sk
group by D1.yid,inv_item_sk,countryid,D2.yid;

select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(agg)
from cube1,dates D1,dates D2,ware
where inv_date_sk>inv_date_kati
and inv_date_sk=D1.did
and inv_date_kati=D2.did
and inv_warehouse_sk=wareid
group by D1.yid,inv_item_sk,countryid,D2.yid;

select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(agg)
from cube2,months D1,months D2,ware
where d_month_seq1=D1.mids
and d_month_seq2=D2.mids
and wareid=inv_warehouse_sk
```

```
group by D1.yid,inv_item_sk,countryid,D2.yid;

select d_year1 ,inv_item_sk,countryid,d_year2 ,sum(agg)
from cube3,ware
where wareid=inv_warehouse_sk
group by d_year1,inv_item_sk,countryid,d_year2;
```

Κύβος 6:

```
select D1.mids ,inv_item_sk,inv_warehouse_sk,D2.mids ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
where D1.mids>D2.mids
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.mids,inv_item_sk,inv_warehouse_sk,D2.mids;

select D1.mids ,inv_item_sk,inv_warehouse_sk,D2.mids ,sum(agg)
from cube1,dates D1,dates D2
where D1.mids>D2.mids
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.mids,inv_item_sk,inv_warehouse_sk,D2.mids;

select d_month_seq1 ,inv_item_sk,inv_warehouse_sk,d_month_seq2 ,sum(agg)
from cube2
where d_month_seq1>d_month_seq2
group by d_month_seq1,inv_item_sk,inv_warehouse_sk,d_month_seq2;
```

Κύβος 7:

```
select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
```

```
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
from cube1,dates D1,dates D2
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
from cube2,months D1,months D2
where  D1.yid>D2.yid
and d_month_seq2=D2.mids
and d_month_seq1=D1.mids
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select d_year1 ,inv_item_sk,inv_warehouse_sk,d_year2 ,sum(agg)
from cube3,ware
where d_year1>d_year2
group by d_year1,inv_item_sk,inv_warehouse_sk,d_year2;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
from cube6,months D1,months D2
where D1.yid>D2.yid
and D1.mids=d_month1
and D2.mids=d_month2
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;

select D1.yid ,inv_item_sk,inv_warehouse_sk,D2.yid ,sum(agg)
```

```
from cube8,dates D1,dates D2
where D1.did=inv_date_sk
and D2.did=inv_date_kati
group by D1.yid,inv_item_sk,inv_warehouse_sk,D2.yid;
```

Κύβος 8:

```
select inv_date_sk ,inv_item_sk,inv_warehouse_sk,inv_date_kati
,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by inv_date_sk,inv_item_sk,inv_warehouse_sk,inv_date_kati;

select inv_date_sk ,inv_item_sk,inv_warehouse_sk,inv_date_kati ,sum(agg)
from cube1,dates D1,dates D2
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did
group by inv_date_sk,inv_item_sk,inv_warehouse_sk,inv_date_kati;
```

Κύβος 9:

```
select inv_date_sk ,inv_warehouse_sk,inv_date_kati ,sum(inv_quantity_on_hand)
from inventory
where inv_date_sk>inv_date_kati
group by inv_date_sk,inv_warehouse_sk,inv_date_kati;

select inv_date_sk ,inv_warehouse_sk,inv_date_kati ,sum(agg)
from cube1
group by inv_date_sk,inv_warehouse_sk,inv_date_kati;
```

Κύβος 10:

```
select inv_date_sk ,inv_date_kati ,sum(inv_quantity_on_hand)
from inventory
where inv_date_sk>inv_date_kati
group by inv_date_sk,inv_date_kati;

select inv_date_sk ,inv_date_kati ,sum(agg)
from cube1
group by inv_date_sk,inv_date_kati;

select inv_date_sk,inv_date_kati,sum(agg)
from cube9
group by inv_date_sk,inv_date_kati;
```

Κύβος 11:

```
select sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did;

select sum(agg)
from cube1,dates D1,dates D2
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did;

select sum(agg)
```

```
from cube2,months D1,months D2
where  D1.yid>D2.yid
and d_month_seq2=D2.mids
and d_month_seq1=D1.mids;

select sum(agg)
from cube3
where d_year1>d_year2;

select sum(agg)
from cube4,months D1,months D2
where D1.yid>D2.yid
and d_month1=D1.mids
and d_month2=D2.mids;

select sum(agg)
from cube5
where d_year1>d_year2;

select sum(agg)
from cube6,months D1,months D2
where D1.yid>D2.yid
and D1.mids=d_month1
and D2.mids=d_month2;

select sum(agg)
from cube7;

select sum(agg)
from cube8,dates D1,dates D2
where D1.did=inv_date_sk
and D2.did=inv_date_kati
```

```
and D1.yid>D2.yid;

select sum(agg)
from cube9,dates D1,dates D2
where inv_date_sk=D1.did
and inv_date_kati=D2.did
and D1.yid>D2.yid;

select sum(agg)
from cube10,dates D1,dates D2
where inv_date_sk=D1.did
and inv_date_kati=D2.did
and D1.yid>D2.yid;

select sum(agg)
from cube12;
```

Κύβος 12:

```
select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(inv_quantity_on_hand)
from inventory,dates D1,dates D2,ware
where D1.yid>D2.yid
and inv_date_sk=D1.did
and inv_date_kati=D2.did
and inv_warehouse_sk=wareid
group by D1.yid,inv_item_sk,countryid,D2.yid;

select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(agg)
from cube1,dates D1,dates D2,ware
where D1.yid>D2.yid
and inv_date_sk=D1.did
```

```
and inv_date_kati=D2.did
and inv_warehouse_sk=wareid
group by D1.yid,inv_item_sk,countryid,D2.yid;

select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(agg)
from cube2,months D1,months D2,ware
where D1.yid>D2.yid
and d_month_seq1=D1.mids
and d_month_seq2=D2.mids
and wareid=inv_warehouse_sk
group by D1.yid,inv_item_sk,countryid,D2.yid;

select d_year1,inv_item_sk,countryid,d_year2 ,sum(agg)
from cube3,ware
where d_year1>d_year2
and wareid=inv_warehouse_sk
group by d_year1,inv_item_sk,countryid,d_year2;

select D1.yid ,inv_item_sk,countryid,D2.yid ,sum(agg)
from cube4,months D1,months D2,street
where D1.yid>D2.yid
and d_month1=D1.mids
and d_month2=D2.mids
and cube4.inv_hier_street=street.streetid
group by D1.yid,inv_item_sk,countryid,D2.yid;

select d_year1,inv_item_sk,countryid,d_year2 ,sum(agg)
from cube5
where d_year1>d_year2
group by d_year1,inv_item_sk,countryid,d_year2;

select D1.yid,inv_item_sk,countryid,D2.yid ,sum(agg)
```

```
from cube6,ware,months D1,months D2
where wareid=inv_warehouse_sk
and D1.mids=d_month1
and D2.mids=d_month2
and D1.yid>D2.yid
group by D1.yid,inv_item_sk,countryid,D2.yid;

select d_year1,inv_item_sk,countryid,d_year2,sum(agg)
from cube7,ware
where wareid=inv_warehouse_sk
group by d_year1,inv_item_sk,countryid,d_year2;

select D1.yid,inv_item_sk,countryid,D2.yid,sum(agg)
from cube8,ware,dates D1,dates D2
where wareid=inv_warehouse_sk
and D1.did=inv_date_sk
and D2.did=inv_date_kati
group by D1.yid,inv_item_sk,countryid,D2.yid;
```

Ορισμοί Κύβων (LevelθValue)

Κύβος 1:

```
insert into c1(ddate,wware,iitem,agg)
SELECT inv_date_sk,inv_warehouse_sk,inv_item_sk,sum(inv_quantity_on_hand)
from inventory where inv_date_sk>=2451180
group by inv_date_sk,inv_warehouse_sk,inv_item_sk;
```

Κύβος 2:

```
insert into c2(dyear,wcountry,iitem,agg)
SELECT yid,countryid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
```

```
where inv_date_sk>=2451180
and inv_date_sk=did
and inv_warehouse_sk=wareid
group by yid,countryid,inv_item_sk;

SELECT yid,countryid,iitem,sum(agg)
 from c1,dates,ware
 where did=ddate
 and wareid=wware
group by yid,countryid,iitem;
```

Κύβος 3:

```
insert into c3(ddate,wware,iitem,agg)
SELECT inv_date_sk,inv_warehouse_sk,inv_item_sk,sum(inv_quantity_on_hand)
from inventory
where inv_date_sk>=2451180
and inv_date_sk<=2451276
group by inv_date_sk,inv_warehouse_sk,inv_item_sk;


SELECT ddate,wware,iitem,sum(agg)
from c1
where ddate<=2451276
group by ddate,wware,iitem;
```

Κύβος 4:

```
insert into c4(dmonth,wcity,iitem,agg)
select mids,cityid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451545
and inv_date_sk=did
and inv_warehouse_sk=wareid
```

```
and cityid>=1110
and cityid<=1120
group by mids,cityid,inv_item_sk;

SELECT mids,cityid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451545
and cityid>=1110
and cityid<=1120
and wware=wareid
and ddate=did
group by mids,cityid,iitem;
```

Κύβος 5:

```
insert into c5(dyear,wcountry,iitem,agg)
SELECT yid,countryid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451545
and inv_date_sk=did
and inv_warehouse_sk=wareid
and countryid=1000
group by yid,countryid,inv_item_sk;

SELECT yid,countryid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451545
and countryid=1000
and ddate=did
and wware=wareid
group by yid,countryid,iitem;

SELECT dyear,wcountry,iitem,sum(agg)
```

```
from c2
where wcountry=1000
and dyear>=(select yid from dates where did=2451545)
group by dyear,wcountry,iitem;

SELECT yid,countryid,iitem,sum(agg)
from c4,city,dates
where countryid=1000
and cityid=wcity
and dmonth=mids
group by yid,countryid,iitem;
```

Κύβος 6:

```
insert into c6(dyear,wcountry,iitem,agg)
select yid,countryid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451545
and inv_date_sk=did
and inv_warehouse_sk=wareid
and countryid=1000
and brand>=50
and brand<=12302
group by yid,countryid,inv_item_sk;

SELECT dyear,wcountry,iitem,sum(agg)
from c2
where dyear>=(select yid from dates where did=2451545)
and wcountry=1000
and iitem>=50
and iitem<=12302
group by dyear,wcountry,iitem;
```

```
SELECT dyear,wcountry,iitem,sum(agg)
from c5
where iitem>=50
and iitem<=12302
group by dyear,wcountry,iitem;


SELECT yid,countryid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451545
and countryid=1000
and ddate=did
and wware=wareid
and iitem>=50
and iitem<=12302
group by yid,countryid,iitem;

SELECT yid,countryid,iitem,sum(agg)
from c4,months,city
where dmonth=mids
and countryid=1000
and wcity=cityid
and iitem>=50
and iitem<=12302
group by yid,countryid,iitem;
```

Κύβος 7:

```
insert into c7(dmonth,wcity,iitem,agg)
select mids,cityid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451911
```

```
and inv_date_sk=did
and inv_warehouse_sk=wareid
and cityid>=1110
and cityid<=1120
group by mids,cityid,inv_item_sk;

SELECT dmonth,wcity,iitem,sum(agg)
from c4
where dmonth>=select mids from dates where date=51911
group by dmonth,wcity,iitem;

SELECT mids,cityid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451911
and cityid>=1110
and   cityid<=1120
and wware=wareid
and ddate=did
group by mids,cityid,iitem;
```

Κύβος 8:

```
insert into c8(dmonth,wcity,iitem,agg)
select mids,cityid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451911
and inv_date_sk<=2452122
and inv_date_sk=did
and inv_warehouse_sk=wareid
and cityid>=1110
and cityid<=1120
group by mids,cityid,inv_item_sk;
```

```
SELECT mids,cityid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451911
and ddate<=2452122
and cityid>=1110
and cityid<=1120
and wware=wareid
and ddate=did
group by mids,cityid,iitem;

select dmonth,wcity,iitem,sum(agg)
from c4,dates
where dmonth>=(select mids from dates where did=2451911)
and dmonth<=(select mids from dates where did=2452122)
group by dmonth,wcity,iitem;

select dmonth,wcity,iitem,sum(agg)
from c7
where dmonth<=(select mids from dates where did=2452122)
group by dmonth,wcity,iitem;
```

Κύβος 9:

```
insert into c9(dmonth,wcountry,iitem,agg)
select mids,countryid,inv_item_sk,sum(inv_quantity_on_hand)
from inventory,dates,ware
where inv_date_sk>=2451911
and inv_date_sk<=2452122
and inv_date_sk=did
and inv_warehouse_sk=wareid
and countryid=1000
```

```
and inv_item_sk>=70
and inv_item_sk<=12302
group by mids,cityid,inv_item_sk;

SELECT mids,countryid,iitem,sum(agg)
from c1,dates,ware
where ddate>=2451911
and ddate<=2452122
and countryid=1000
and ddate=did
and wware=wareid
and iitem>=70
and iitem<=12302
group by mids,countryid,iitem;

select dmonth,countryid,iitem,sum(agg)
from c4,city
where dmonth>=(select mids from dates where did=2451911)
and dmonth<=(select mids from dates where did=2452122)
and wcity-cityid
and iitem>=70
and iitem<=12302
group by dmonth,countryid,iitem;

select dmonth,countryid,iitem,sum(agg)
from c7,city
where dmonth>=(select mids from dates where did=2451911)
and dmonth<=(select mids from dates where did=2452122)
and iitem>=70
and iitem<=12302
and cityid=wcity
and countryid=1000
```

```
group by dmonth,countryid,iitem;

select dmonth,countryid,iitem,sum(agg)
from c8,city
where iitem>=70
and iitem<=12302
and cityid=wcity
and countryid=1000
group by dmonth,countryid,iitem;
```

ΑΝΑΦΟΡΕΣ

[AgGS95] R. Agrawal, A. Gupta, S. Sarawagi. Modeling Multidimensional Databases. IBM Research Report, IBM Almaden Research Center, September 1995.

[Arbo96] Arbor Software Corporation. Arbor Essbase.http://www.arborsoft.com/essbase.html, 1996.

[BaSa98] F. Baader and U. Sattler. Description Logics with Concrete Domains and Aggregation. Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98), pp. 336-340, 1998.

[BaPT97] E. Baralis, S. Paraboschi, E. Teniente. Materialized View Selection in a Multidimensional Database. In Proceedings of the 23rd International Conference on Very Large Databases (VLDB), Athens, August 1997.

[ChSh96] S. Chaudhuri, K. Shim: Optimizing Queries with Aggregate Views. In Proc. of EDBT 1996. Proceedings of the 5th International Conference on Extending Database Technology (EDBT-96), Avignon, France, March 25-29, 1996.

[CKPS95] S. Chaudhuri, S. Krishnamurthy, S. Potamianos, and K. Shim. Optimizing queries with materialized views. Proceedings of the 11th International Conference on Data Engineering (ICDE), IEEE Computer Society, pp. 190-200, Taipei, March 1995.

[Co05] Sara Cohen, Containment of Aggregate Queries, SIGMOD Rec. Vol 34 2005

[Coll96] G. Colliat. OLAP, Relational, and Multidimensional Database Systems. SIGMOD Record, vol. 25, No.3, September 1996.

[CoNS99] S. Cohen, W. Nutt, A. Serebrenik: Rewriting Aggregate Queries Using Views. Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), Philadelphia, Pennsylvania. ACM Press, 1999.

[CaTo97] L. Cabbibo and R. Torlone. Querying Multidimesional Databases. 6th International Workshop on Database Programming Languages (DBPL6), 1997.

[CaTo98a] L. Cabbibo, R. Torlone. A Logical Approach to Multidimensional Databases. In 6th EDBT, 1998.

[CaTo98b] L. Cabibbo, R. Torlone. From a Procedural to a Visual Query Language for OLAP. Proceedings of 10th International Conference on Scientific and Statistical Database Management (SSDBM), Capri, Italy, July 1998.

[DJLS96] S. Dar, H.V.Jagadish, A. Levy, D. Srivastava. Answering queries with aggregation using views. In Proc. of 22nd International Conference on Very Large Data Bases (VLDB), Mumbai, India, 1996.

[Ende72] H.B. Enderton, A Mathematical Introduction to Logic. Academic Press, 1972.

[GBLP96] J. Gray, A. Bosworth, A. Layman, H. Pirahesh. Data Cube. A Relational Aggregation Operator Generalizing Group-By, Cross-Tabs, and Sub-Totals. Proceedings of the 12th International Conference on Data Engineering (ICDE '96), New Orleans, February 1996. Also Microsoft Technical Report, MSR-TR-95-22, available at http://www.research.microsoft.com/~gray.

[GuHQ95] A. Gupta, V. Harinarayan, and D. Quass. Aggregate query processing in data warehouses. Proceedings of the 21st International Conference on Very Large Data Bases (VLDB), Zurich, Switzerland, Morgan Kaufmann Publishers, August 1995.

[GeJJ97] M. Gebhardt, M Jarke and S. Jacobs. A toolkit for negotiation support on multi-dimensional data. Proceedings of ACM SIGMOD International Conference on Management of Data. Tucson, Arizona, 1997.

[GyLa97] M. Gyssens, L.V.S. Lakshmanan. A Foundation for Multi-Dimensional Databases. Proceedings of the 23rd International Conference on Very Large Databases (VLDB), Athens, August 1997.

[GiLa98] F. Gingras, L. Lakshmanan. nD-SQL: A Multi-dimensional Language for Interoperability and OLAP. Proceedings of the 24th International Conference on Very Large Databases (VLDB), N. York, August 1998.

[GT03] Stephane Grumbach, Leonardo Tininini: On the content of materialized aggregate views. J. Comput. Syst. Sci. 66(1): 133-168 (2003)

[Gupt97] H. Gupta. Selection of Views to Materialize in a Data Warehouse. In Proceedings of the 6th International Conference on Database Theory (ICDT-97), Delfi, Greece, 1997

[Ha01] Survey:Halevy Ansering Queries using Views

[Info97] Informix, Inc.: The INFORMIX-MetaCube Product Suite. http://www.informix.com/informix/products/new_plo/metabro/metabro2.htm, 1997.

[JLVV00] M. Jarke, M. Lenzerini, Y. Vassiliou, P. Vassiliadis (eds.). Fundamentals of Data Warehouses. Springer-Verlag, 2000.

[Kimb96] R. Kimball. The Data Warehouse Toolkit: Practical techniques for building dimensional data warehouses. John Wiley. 1996.

[LeAW98] W. Lehner, J. Albrect, H. Wedekind. Normal Forms for Multidimensional Databases. Proceedings of 10th International Conference on Scientific and Statistical Database Management (SSDBM), Capri, Italy, July 1998.

[Lehn98] W. Lehner. Modeling Large Scale OLAP Scenarios. Proceedings of the 6th International Conference of Extending Database Technology (EDBT-98), 1998.

[LMSS95] A.Y. Levy, A.O. Mendelzon, Y. Sagiv, and D. Srivastava. Answering queries using views. Proceedings of the 14th Symposium on Principles of Database Systems (PODS), pp. 95-104, San Jose (California, USA), May 1995. ACM Press.

[LeSh97] H. Lenz, A. Shoshani. Summarizability in OLAP and Statistical databases. In Proceedings of 9th International Conference on Scientific and Statistical Database Management (SSDBM), 1997.

[LSTV99] S.Ligoudistianos, T.Sellis, D.Theodoratos, and Y.Vassiliou. Heuristic Algorithms for Designing the Data Warehouse with SPJ Views. Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery, (DaWaK), Lecture Notes in Computer Science, Vol. 1676, Springer, 1999.

[LiWa96] C. Li, X. Sean Wang. A Data Model for Supporting On-Line Analytical Processing. Proceedings of the International Conference on Information and Knowledge Management (CIKM), 1996.

[Meta97] Metadata Coalition. Metadata Interchange Specification (MDIS v. 1.1). http://www.metadata.org/standards/toc.html 1997.

[Micr98] Microsoft Corp. OLEDB for OLAP February 1998. Available at http://www.microsoft.com/data/oledb/olap/

[MStr97] MicroStrategy, Inc. MicroStrategy's 4.0 Product Line. http://www.strategy.com/launch/4_0_arc1.htm, 1997.

[NuSS98] W.Nutt, Y.Sagiv, S. Surin. Deciding Equivalences among Aggregate Queries. In Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on

Principles of Database Systems (PODS), Seattle, USA, 1998

[OLAP97] OLAP Council. OLAP AND OLAP Server Definitions. 1997 Available at http://www.olapcouncil.org/research/glossaryly.htm

[OLAP97a] OLAP Council. The APB-1 Benchmark. 1997. Available at http://www.olapcouncil.org/research/bmarkly.htm

[RBSI97] Red Brick Systems, Inc. Red Brick Warehouse 5.0. http://www.redbrick.com/rbs-g/html/whouse50.html, 1997.

[Sara97] Sunita Sarawagi. Indexing OLAP Data. Data Engineering Bulletin 20(1): 36-43 (1997).

[Shos97] A. Shoshani. OLAP and Statistical Databases: Similarities and Differences. Tutorials of 16th Symposium on Principles Of Database Systems (PODS), 1997.

[STGI96] Stanford Technology Group, Inc. Designing the Data Warehouse on Relational Databases. http://www.informix.com/informix/corpinfo/zines/whitpprs/stg/metacube.htm, 1996.

[ThLS99] D.Theodoratos, S.Ligoudistianos, and T.Sellis. Designing the Global DW with SPJ Queries. In Proc. of the 11th Conference on Advanced Information Systems Engineering (CAiSE), pages 180--194, June 1999.

[TPC99] TPC: TPC Benchmark H and TPC Benchmark R. Transcation Processing Council. June 1999. Available at http://www.tpc.org/

[Ullm89] J. Ullman. "Principles of Database and Knowledge-Base Systems. Volume II: The New Technologies. Computer Science Press. 1989.

[Ullm97] J.D. Ullman. Information integration using logical views. Proceedings of the 6th International Conference on Database Theory (ICDT-97), Lecture Notes in Computer Science, pp. 19-40. Springer-Verlag, 1997.

[VaSe99] P. Vassiliadis, T. Sellis. “A Survey on Logical Models for OLAP Databases”. SIGMOD Record, vol. 28, no. 4, December 1999.

[VaSk00] P. Vassiliadis, S. Skiadopoulos. Modelling and Optimization Issues for Multidimensional Databases. In Proc. 12th Conference on Advanced Information Systems Engineering (CAiSE '00), pp. 482-497, Stockholm, Sweden, 5-9 June 2000. Lecture Notes in Computer Science, Vol. 1789, Springer, 2000.

[VaSk99] P. Vassiliadis, S. Skiadopoulos. Modeling and Optimization Issues for Multidimensional Databases, Extended version, Technical Report, KDBSL 1999. Available at

http://www.dblab.ece.ntua.gr/~pvassil/publications/cube99.ps.gz

[Vass98] P. Vassiliadis. Modeling Multidimensional Databases, Cubes and Cube Operations. Proceedings of 10th International Conference on Scientific and Statistical Database Management (SSDBM), Capri, Italy, July 1998.

[YaLa85] P. Larson, H. Z. Yang: Computing Queries from Derived Relations. In Proc. of VLDB 1985. Proceedings of the 11th International Conference on Very Large Data Bases (VLDB), Stockholm, Sweden, Morgan Kaufmann Publishers, August 1985.

ΣΥΝΤΟΜΟ ΒΙΟΓΡΑΦΙΚΟ

Η Χαρά Παπαγεωργίου γεννήθηκε στην Πρέβεζα το 1984. Αποφοίτησε από το Λύκειο το 2001 και είσήχθησε στο τμήμα Εφαρμοσμένης Πληροφορικής του Πανεπιστημίου Μακεδονίας, από το οποίο αποφοίτησε το 2005. Τον Σεπτέμβριο του 2006 εισήχθησε στο Πρόγραμμα Μεταπτυχιακών Σπουδών του Τμήματος Πληροφορικής του Πανεπιστημίου Ιωαννίνων. Τα ερευνητικά της ενδιαφέροντα εντοπίζονται στις περιοχές Βάσεων Δεδομένων, Αποθηκών δεδομένων και στην OLAP επεξεργασία.