ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

ΚΩΝΣΤΑΝΤΙΝΟΥ ΒΑΣΙΛΑΚΗ

ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΒΕΛΤΙΣΤΟΠΟΙΗΜΕΝΗ ΥΛΟΠΟΙΗΣΗ ΕΝΟΣ ΣΥΣΤΗΜΑΤΟΣ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ ΙΣΤΟΡΙΚΟΥ ΤΥΠΟΥ

ΟΚΤΩΒΡΙΟΣ 1995

ΤΡΙΜΕΛΗΣ ΕΠΙΤΡΟΠΗ

Καθηγητής Γ. Φιλοκύπρου

Καθηγητής Κ. Χαλάτσης

Αν. Καθηγητής Π. Γεωργιάδης

ΜΕΛΗ ΕΠΤΑΜΕΛΟΥΣ ΕΠΙΤΡΟΠΗΣ

Καθηγητής Μ. Χατζόπουλος

Αν. Καθηγητής Τ. Σελλής

Επ. Καθηγητής Ν. Λορέντζος

Επ. Καθηγητής Γ. Μανωλόπουλος

Ο Κ. Βασιλάκης διετέλεσε υποψήφιος του Ιδρύματος Κρατικών Υποτροφιών (Ι. Κ. Υ.) κατά την περίοδο Νοέμβριος 1991 έως Οκτώβριος 1994.

Πρόλογος

Η διατριβή αυτή είχε ως αφετηρία το ερευνητικό πρόγραμμα ESPRIT III 7224 - ORES, το οποίο αποσκοπούσε στην ανάπτυξη ενός χρονολογικού συστήματος διαχείρισης βάσεων δεδομένων πρώτης γενιάς. Στα πλαίσια του προγράμματος αυτού σχεδιάστηκε και αναπτύχθηκε ένα χρονολογικό ΣΔΒΔ, γραφικά εργαλεία για σχεδιασμό χρονολογικών βάσεων δεδομένων και για υποβολή ερωτήσεων σε βάσεις αυτού του τύπου και μία εφαρμογή διαχείρισης ιατρικών δεδομένων για τη μονάδα μεταμοσχεύσεων ήπατος του νοσοκομείου της Μαδρίτης Clinica Puerta de Hiero. Η ομάδα του Πανεπιστημίου Αθηνών, τις εργασίες της οποίας διηύθυνε ο αναπληρωτής καθηγητής κ. Γεωργιάδης, ανέπτυξε το χρονολογικό ΣΔΒΔ και την ιατρική εφαρμογή.

Μετά το πέρας του ερευνητικού προγράμματος ORES, σχεδιάστηκαν βελτιστοποιημένοι αλγόριθμοι για την υλοποίηση των εντολών διαχείρισης δεδομένων του χρονολογικού ΣΔΒΔ. Οι βελτιστοποιημένοι αλγόριθμοι υλοποιήθηκαν και η υπεροχή τους, από άποψη απόδοσης, έναντι των αλγορίθμων που είχαν χρησιμοποιηθεί στην πρώτη υλοποίηση τεκμηριώθηκε τόσο βάσει ενός θεωρητικού μοντέλου κόστους, όσο και από τα αποτελέσματα πειραματικών μετρήσεων. Σχεδιάστηκαν επίσης τεχνικές για υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών στο χρονολογικό ΣΔΒΔ, οι οποίες ενσωματώθηκαν στη βελτιστοποιημένη υλοποίηση.

Στο σημείο αυτό θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν στην ολοκλήρωση της παρούσας διατριβής.

Τα μέλη της τριμελούς συμβουλευτικής επιτροπής, αποτελούμενη από τον καθηγητή κ. Γ. Φιλοκύπρου, τον καθηγητή κ. Κ. Χαλάτση και τον αναπληρωτή καθηγητή κ. Π. Γεωργιάδη, συνέβαλαν στην ανάληψη του προγράμματος ORES από το Πανεπιστήμιο Αθηνών και επέδειξαν συνεχές επιστημονικό ενδιαφέρον για την ολοκλήρωση του ερευνητικού προγράμματος. Ειδικότερα ο κ. Γεωργιάδης παρείχε πολύτιμη βοήθεια στη διαμόρφωση του τελικού κειμένου της διατριβής και προσέφερε αμέριστη συμπαράσταση κατά τις πιο δύσκολες φάσεις της πορείας προς την ολοκλήρωσή της.

Επίσης, θα ήθελα να ευχαριστήσω τον καθηγητή κ. Μ. Χατζόπουλο, για τις εύστοχες υποδείξεις του και τη γενικότερη συμβολή του στο ερευνητικό πρόγραμμα ORES.

Η ομάδα του Πανεπιστημίου Αθηνών που συμμετείχε στο πρόγραμμα ORES, διευθυνόμενη από τον κ. Γεωργιάδη και αποτελούμενη πλην εμού, από τους Δ. Αναγνωστόπουλο, Γ. Λασκαρίδη, Κ. Μπουκουβάλα, Μ. Νικολαΐδου και Α. Σωτηροπούλου συνεισέφερε τα μέγιστα στην υλοποίηση της πρώτης έκδοσης του χρονολογικού ΣΔΒΔ.

Ο επίκουρος καθηγητής του Γεωργικού Πανεπιστημίου Αθηνών κ. Ν. Λορέντζος είναι ο εμπνευστής του μοντέλου δεδομένων στο οποίο στηρίχτηκε αυτή η ερευνητική εργασία. Ο κ. Ν. Λορέντζος είχε σημαντική συνεισφορά στη διαδικασία σχεδιασμού και ανάπτυξης του χρονολογικού ΣΔΒΔ. Η εποικοδομητική συνεργασία με τον κ. Ν. Λορέντζο συνεχίστηκε και μετά το πέρας του προγράμματος ORES, για τη διαμόρφωση εργασιών, οι οποίες υποβλήθηκαν για δημοσίευση.

Τέλος, ο κ. Γαζέπης, της Information Dynamics επέκτεινε τις δυνατότητες του πυρήνα του Ingres, διευκολύνοντας την υλοποίηση του χρονολογικού ΣΔΒΔ.

Αύγουστος 1995

Κ. Βασιλάκης

Περίληψη

Τα χρονολογικά συστήματα διαχείρισης βάσεων δεδομένων έχουν συγκεντρώσει το ενδιαφέρον πολλών ερευνητών κατά την τελευταία δεκαπενταετία. Πολλοί ερευνητές έχουν υποστηρίξει την αναγκαιότητα υποστήριξης της χρονικής διάστασης στα συστήματα διαχείρισης βάσεων δεδομένων και έχουν αποδείξει ότι αυτή η υποστήριξη θα επέφερε σημαντικές ωφέλειες σε πολλούς τομείς εφαρμογών.

Κατά την περίοδο αυτή, έχουν προταθεί πολυάριθμα μοντέλα δεδομένων για την αναπαράσταση χρονολογικών δεδομένων, ενώ έχουν επίσης σχεδιαστεί και γλώσσες που επιτρέπουν τον ορισμό και τη διαχείριση τους. Παρ' όλα αυτά, τα εμπορικά συστήματα διαχείρισης βάσεων δεδομένων δεν προσφέρουν, μέχρι σήμερα, υποστήριξη για τα χρονολογικά δεδομένα, ενώ και στην ακαδημαϊκή κοινότητα , ο αριθμός των υλοποιήσεων είναι σημαντικά περιορισμένος.

Στην παρούσα εργασία, παρουσιάζουμε τον σχεδιασμό και την υλοποίηση ενός ΣΔΒΔ ιστορικού τύπου, δηλ. ενός ΣΔΒΔ που υποστηρίζει δεδομένα με καθορισμένο χρόνο εγκυρότητας. Το ΣΔΒΔ χρησιμοποιεί την SQL-XE, μία συντακτικά και σημασιολογικά συνεπή επέκταση της SQL89, ως γλώσσα ορισμού και διαχείρισης δεδομένων. Η SQL-XE επιτρέπει τον ορισμό πινάκων που περιέχουν δεδομένα με καθορισμένο χρόνο εγκυρότητας και διαχειρίζεται τα δεδομένα αυτά, λαμβάνοντας υπόψη τη χρονική σημασιολογία τους.

Το ΣΔΒΔ ιστορικού τύπου έχει υλοποιηθεί ως ένας εξωτερικός φλοιός στο σχεσιακό ΣΔΒΔ Ingres. Το Ingres χρησιμοποιείται για την αποθήκευση και την ανάκτηση δεδομένων, σε φυσικό επίπεδο, ενώ ο φλοιός υποστηρίζει το συντακτικό της SQL-XE και υλοποιεί τα χρονολογικά σημασιολογικά χαρακτηριστικά.

Ιδιαίτερο βάρος έχει δοθεί στην απόδοση του ΣΔΒΔ ιστορικού τύπου, έτσι ώστε η υποστήριξη των επιπρόσθετων σημασιολογικών χαρακτηριστικών να μην οδηγεί σε υποβάθμιση της απόδοσης. Για τον σκοπό αυτό αναπτύχθηκαν αλγόριθμοι που εκμεταλλεύονται τις δυνατότητες που προσφέρει η εμφυτευμένη SQL και επιτυγχάνουν χρόνους εκτέλεσης σημαντικά καλύτερους από τους αλγόριθμους που είχαν χρησιμοποιηθεί σε παλαιότερες υλοποιήσεις του ΣΔΒΔ ιστορικού τύπου.

Τέλος, το ΣΔΒΔ ιστορικού τύπου εμπλουτίστηκε με δυνατότητες υποστήριξης δοσοληψιών και ταυτόχρονους χρήστες, που θεωρούνται ιδιαίτερα σημαντικά χαρακτηριστικά για ένα ΣΔΒΔ.

Design and Optimised Implementation of an Historical Database Management System

Abstract

Temporal database systems have received a lot of the researchers' attention during the last 15 years. Many researchers have argued for the need of supporting the time dimension in database management systems, and have proved that such a support would greatly enhance the functionality of many applications.

During this period, numerous data models have been proposed for the representation of temporal data and languages have been designed for the definition and manipulation of temporal data. However, commercial database management systems have not incorporated support for the management of temporal data and the number of implementations within the academic community is very limited.

In this study, we present the design and implementation of an historical DBMS, ie. a DBMS supporting valid time data. The DBMS uses VT-SQL, a syntactically and semantically consistent extension to SQL89 as its data definition and manipulation language. VT-SQL allows for the creation of tables containing valid time data and manipulates these data taking into account their temporal semantics.

The historical DBMS has been built as a shell to the Ingres RDBMS. Ingres is used for the storage and retrieval of data, at the physical level, while the shell supports the VT-SQL syntax and implements the temporal semantics.

Emphasis has been put in the performance aspects of the historical DBMS, so that the support of the additional semantics does not lead to poor performance. In order to provide high performance, new algorithms have been designed, exploiting the potential offered by embedded SQL. These algorithms deliver considerably better performance than the ones that had been used in previous implementations of the historical DBMS.

Finally, the historical DBMS has been enhanced with support for transactions and concurrent users, which are important features for a DBMS.

Περιεχόμενα

Ευρετήριο σχημάτων.

1. Εισαγωγή.

Η ανάγκη για υποστήριξη *χρονολογικών δεδομένων* (temporal data-[Jensen92]), δεδομένων δηλαδή που μεταβάλλονται με τον χρόνο, έχει διαπιστωθεί από πολλούς ερευνητές. Οι πρώτες αναφορές χρονολογούνται στο 1977 ([Schueler77]) και έκτοτε η μοντελοποίηση, η διαχείριση και οι εφαρμογές των χρονολογικών δεδομένων έχουν μελετηθεί σε πολλές εργασίες (εκτενείς βιβλιογραφικές αναφορές μπορούν να βρεθούν στα [Bolour83], [McKenzie86], [Stam88], [Soo91], [Taha93], [Kline93]). Οι πιο χαρακτηριστικές εφαρμογές που θα ωφελούνταν από τη χρήση χρονολογικών δεδομένων μπορούν να εντοπισθούν στους τομείς της υγείας, του περιβάλλοντος, της διοίκησης, της οικονομίας και της άμυνας.

Τα χρονολογικά δεδομένα που βρίσκονται καταχωρισμένα σε μία βάση δεδομένων μπορούν να συσχετίζονται με τρεις *χρονικές διαστάσεις* (time dimensions-[Snodgrass87]): τον *χρόνο εγκυρότητας* (valid time), τον *χρόνο δοσοληψίας* (transaction time) και τον *οριζόμενο από τον χρήστη χρόνο* (user-defined time). Για κάθε χρονική διάσταση, αποθηκεύεται στη βάση δεδομένων μία *χρονική ένδειξη* (timestamp), η οποία σχετίζεται με ένα σύνολο χρονολογικών δεδομένων, τα οποία και χαρακτηρίζει.

Ο χρόνος εγκυρότητας αντιπροσωπεύει το χρονικό διάστημα κατά το οποίο το γεγονός ήταν αληθές στον πραγματικό κόσμο. Ο χρόνος εγκυρότητας μπορεί να είναι μία περίοδος στον παρελθόν, ή ακόμη και ένα μελλοντικό διάστημα, αν προβλέπεται ότι η πληροφορία με την οποία σχετίζεται θα είναι αληθής κατά το διάστημα αυτό. Σύμφωνα με τα ανωτέρω, το διάστημα 330 μ.Χ. έως 1453 μ.Χ. είναι ο χρόνος εγκυρότητας της πληροφορίας *ύπαρξη Βυζαντινής Αυτοκρατορίας*, ενώ το έτος 1997 είναι ο χρόνος εγκυρότητας του γεγονότος *ολοκλήρωση της ευρωπαϊκής ένωσης*.

Ο χρόνος δοσοληψίας αντικατοπτρίζει την περίοδο κατά την οποία η σχετική πληροφορία ήταν καταχωρισμένη στη βάση δεδομένων. Ο χρόνος δοσοληψίας που συσχετίζεται με κάποια πληροφορία, έχει ως αφετηρία τη χρονική στιγμή που η πληροφορία εισάγεται στη βάση δεδομένων, ενώ το πέρας του εντοπίζεται στη χρονική στιγμή κατά την οποία η πληροφορία διαγράφεται.

Τέλος, τα σημασιολογικά χαρακτηριστικά του οριζόμενου από τον χρήστη χρόνου είναι γνωστά μόνο στην εφαρμογή που διαχειρίζεται τα δεδομένα.

Τα ΣΔΒΔ κατατάσσονται σε τέσσερις κατηγορίες ([Snodgrass87], [Jensen93]), ανάλογα με το αν παρέχουν υποστήριξη για τον χρόνο εγκυρότητας και τον χρόνο δοσοληψίας

(δεδομένου ότι ο οριζόμενος από τον χρήστη χρόνος δεν έχει καθορισμένη σημασιολογία, σε ό,τι αφορά το σύστημα, το ΣΔΒΔ δεν μπορεί να ερμηνεύσει ή να διαχειριστεί κατά ιδιαίτερο τρόπο τα δεδομένα αυτά). Τα ΣΔΒΔ που δεν υποστηρίζουν καμία από τις δύο χρονικές διαστάσεις καλούνται *ΣΔΒΔ στιγμιοτύπου* (snapshot DBMS), καθώς περιέχουν πάντα την πληροφορία που περιγράφει τον πραγματικό κόσμο (το τμήμα που έχει μοντελοποιηθεί) για μία μόνο χρονική στιγμή (κατά κανόνα την παρούσα). Τα ΣΔΒΔ που υποστηρίζουν μόνο τον χρόνο εγκυρότητας ονομάζονται *ΣΔΒΔ ιστορικού τύπου* (historical DBMSs) ή *ΣΔΒΔ χρόνου εγκυρότητας*, ενώ τα ΣΔΒΔ που υποστηρίζουν μόνο τον χρόνο δοσοληψίας ονομάζονται *ΣΔΒΔ επαναφοράς* (rollback DBMSs). Τέλος, τα ΣΔΒΔ που υποστηρίζουν και τις δύο χρονικές διαστάσεις ονομάζονται *αμφιχρονολογικά ΣΔΒΔ* (bitemporal DBMSs).

Για την αναπαράσταση των χρονολογικών δεδομένων, έχουν προταθεί πολλές επεκτάσεις τόσο του σχεσιακού, όσο και του αντικειμενοστρεφούς μοντέλου ([Jones79], [Ben-Zvi82], [Clifford83], [Ariav86], [Snodgrass86], [Tansel86], [Clifford87], [Segev87], [Snodgrass87], [Stonebreaker87], [Beech88], [Caruso88], [Gadia88], [Lorentzos88], [Navathe89], [Elmarsi90], [Sarda90], [Jensen91], [Rose91], [Sciore91], [Su91], [Gadia92], [Wuu92], [Rose93]). Οι κυριότερες διαφορές μεταξύ των μοντέλων δεδομένων εντοπίζονται στα ακόλουθα σημεία:

1. *υποστήριξη χρονικών διαστάσεων*. Τα περισσότερα από τα προτεινόμενα μοντέλα δεδομένων υποστηρίζουν μόνο τη μία χρονική διάσταση, ενώ λίγα είναι αυτά που παρέχουν πλήρη υποστήριξη και για τις δύο χρονικές διαστάσεις. Σημειώνουμε εδώ ότι, επειδή η σημασιολογία του χρόνου εγκυρότητας είναι διαφορετική από τη σημασιολογία του χρόνου δοσοληψίας, η υποστήριξη της κάθε χρονικής διάστασης απαιτεί ξεχωριστές συντακτικές και σημασιολογικές επεκτάσεις στη γλώσσα ορισμού και διαχείρισης δεδομένων.
2. *είδος χρονικών ενδείξεων*. Οι χρονικές ενδείξεις που συσχετίζονται με τα αντικείμενα μπορούν να περιέχουν μία ή δύο τιμές, ανάλογα με το πλήθος των χρονικών διαστάσεων που υποστηρίζονται από το μοντέλο. Κάθε στιγμή, με τη σειρά της, μπορεί να αντιπροσωπεύει ένα μόνο χρονικό κβάντο (chronon), ένα σύνολο διαδοχικών χρονικών κβάντων (που συνήθως καλείται *χρονικό διάστημα*- time interval) ή ένα σύνολο μη γειτονικών και μη επικαλυπτομένων χρονικών διαστημάτων, το οποίο καλείται *χρονολογικό στοιχείο* (temporal element).
3. *χρονολογική ομοιογένεια* (temporal homogeneity). Όταν οι χρονικές ενδείξεις συσχετίζονται με το σύνολο μίας πληροφοριακής ενότητας (μιας *πλειάδας* για τις

επεκτάσεις του σχεσιακού μοντέλου ή ενός *αντικειμένου* για τα αντικειμενοστρεφή ΣΔΒΔ), τότε η πληροφοριακή ενότητα καλείται *χρονολογικά ομογενής*. Αντίθετα, όταν χρονικές ενδείξεις συσχετίζονται με *τμήματα* της πληροφοριακής ενότητας (ένα ιδιοχαρακτηριστικό -attribute- ή μια ομάδα ιδιοχαρακτηριστικών), η πληροφοριακή ενότητα καλείται *χρονολογικά ετερογενής*. Μία βάση δεδομένων καλείται *χρονολογικά ομογενής* όταν όλες οι πληροφοριακές ενότητές της είναι χρονολογικά ομογενείς. Η χρονολογική ομοιογένεια ορίζεται τόσο για τον χρόνο εγκυρότητας, όσο και για τον χρόνο δοσοληψίας.

Από τις ανωτέρω κατηγορίες μοντέλων δεδομένων, ιδιαίτερο ενδιαφέρον παρουσιάζουν οι επεκτάσεις του σχεσιακού μοντέλου που είναι χρονολογικά ομογενείς και χρησιμοποιούν διαστήματα για την αναπαράσταση των χρονικών ενδείξεων, μια και αποτελούν φυσική επέκταση του σχεσιακού μοντέλου, στην οποία με κάθε πλειάδα συσχετίζεται μία χρονική ένδειξη. Το σχεσιακό μοντέλο κρίνεται προτιμότερο του αντικειμενοστρεφούς, για τους ακόλουθους λόγους:

1. οι προτεινόμενες επεκτάσεις του σχεσιακού μοντέλου έχουν ισχυρή θεωρητική τεκμηρίωση, σε αντίθεση με τις αντικειμενοστρεφείς προσεγγίσεις, από τις οποίες λίγες μόνο βασίζονται σε κάποια άλγεβρα.
2. οι αντικειμενοστρεφείς προσεγγίσεις είναι στενά συνυφασμένες με το μοντέλο δεδομένων και τη γλώσσα ορισμού και διαχείρισης δεδομένων κάποιου αντικειμενοστρεφούς ΣΔΒΔ, σε αντίθεση με τις επεκτάσεις του σχεσιακού μοντέλου, οι οποίες εμπλουτίζουν τη σχεσιακή άλγεβρα ([Codd70]) και την SQL.

Ένα επιπρόσθετο πλεονέκτημα της προαναφερθείσας κατηγορίας μοντέλων δεδομένων, είναι ότι τα δεδομένα βρίσκονται σε πρώτη κανονική μορφή ([Date82]), καθιστώντας έτσι δυνατή τη χρήση των τεχνικών που χρησιμοποιούνται στα σχεσιακά ΣΔΒΔ, στο φυσικό επίπεδο αποθήκευσης. Οι τεχνικές αυτές αποτελούν το αντικείμενο μελέτης για περισσότερα από 20 χρόνια και στο διάστημα αυτό έχει πιστοποιηθεί η ορθότητά τους και έχουν βελτιστοποιηθεί.

Αντικείμενο αυτής της εργασίας είναι η παρουσίαση και η υλοποίηση ενός ΣΔΒΔ ιστορικού τύπου, το οποίο δίνει ιδιαίτερη έμφαση στην απόδοση. Το θεωρητικό υπόβαθρο του χρονολογικού ΣΔΒΔ είναι η σχεσιακή άλγεβρα χρόνου εγκυρότητας (ΣΑΧΕ-VTRA([ESPRIT93α])), ενώ ως γλώσσα ορισμού και διαχείρισης δεδομένων χρησιμοποιείται η SQL-XE (VT-SQL([ESPRIT93β])).

Η παρούσα εργασία έχει την ακόλουθη δομή: στο δεύτερο κεφάλαιο παρουσιάζεται η ΣΑΧΕ, η οποία αποτελεί τη θεωρητική βάση του χρονολογικού ΣΔΒΔ που αναπτύχθηκε.

Στο τρίτο κεφάλαιο περιγράφεται το συντακτικό και η σημασιολογία της SQL-XE, μιας επέκτασης του προτύπου SQL89 ([Lans88]), η οποία επιτρέπει τη δημιουργία και τη διαχείριση πινάκων που περιέχουν δεδομένα με καθορισμένο χρόνο εγκυρότητας.

Το τέταρτο κεφάλαιο είναι αφιερωμένο στην υλοποίηση ενός χρονολογικού ΣΔΒΔ, το οποίο βασίζεται στη ΣΑΧΕ και χρησιμοποιεί την SQL-XE ως γλώσσα ορισμού και διαχείρισης δεδομένων. Στο κεφάλαιο αυτό παρουσιάζεται η συνολική αρχιτεκτονική του χρονολογικού ΣΔΒΔ, αναλύεται η λειτουργικότητα των επί μέρους τμημάτων του και περιγράφονται οι αλγόριθμοι που χρησιμοποιήθηκαν για την εκτέλεση των εντολών της SQL-XE. Ιδιαίτερη έμφαση δίνεται στην απόδοση του χρονολογικού ΣΔΒΔ και πιο συγκεκριμένα στις επιδόσεις των αλγορίθμων εκτέλεσης των εντολών που επιφέρουν τροποποίηση στην κατάσταση της βάσης δεδομένων (INSERT, DELETE και UPDATE). Το κόστος εκτέλεσης των εντολών αυτών, σύμφωνα με τους βελτιστοποιημένους αλγόριθμους, εκτιμάται, βάσει ενός μαθηματικού μοντέλου και συγκρίνεται με το κόστος εκτέλεσης των ίδιων εντολών, σύμφωνα με μη βελτιστοποιημένους αλγόριθμους που έχουν χρησιμοποιηθεί στο παρελθόν. Η βελτίωση που επέρχεται στις επιδόσεις του χρονολογικού ΣΔΒΔ, τεκμηριώνεται και βάσει χαρακτηριστικών διαγραμμάτων απόδοσης.

Στο κεφάλαιο 5 παρουσιάζονται τεχνικές για τον εμπλουτισμό του χρονολογικού ΣΔΒΔ, που περιγράφηκε στο κεφάλαιο 4, με δυνατότητες υποστήριξης δοσοληψιών (transactions) και ταυτοχρόνων χρηστών (concurrent users). Έμφαση δίνεται στο να εισάγονται οι επιπρόσθετες δυνατότητες, κατά τρόπο ώστε να μην επέρχεται δυσανάλογη επιβάρυνση στις επιδόσεις του χρονολογικού ΣΔΒΔ. Η αποδοτικότητα του προτεινόμενου σχήματος τεκμηριώνεται τόσο βάσει του μαθηματικού μοντέλου, που εισάγεται στο κεφάλαιο 4, όσο και βάσει διαγραμμάτων απόδοσης.

Τέλος, στο κεφάλαιο 6 συνοψίζονται τα συμπεράσματα της παρούσας εργασίας και αναδεικνύονται ορισμένες κατευθύνσεις για μελλοντική έρευνα.

2. Η Σχεσιακή Άλγεβρα Χρόνου Εγκυρότητας.

Η σχεσιακή άλγεβρα χρόνου εγκυρότητας (ΣΑΧΕ-valid time algebra (VT-RA) [ESPRIT93α]) αποτελεί μία συνεπή επέκταση της σχεσιακής άλγεβρας ([Codd70]), διατηρώντας τη βασική της δομή, τη *σχέση* ή *πίνακα* (relation ή table). Κάθε σχέση μπορεί να αναπαρασταθεί με έναν δισδιάστατο πίνακα, οι γραμμές του οποίου καλούνται *πλειάδες*. Κάθε στήλη μιας σχέσης έχει διαφορετικό όνομα από τις υπόλοιπες και όλες οι τιμές μιας στήλης ανήκουν στο ίδιο *πεδίο ορισμού*. Το διατεταγμένο σύνολο των ονομάτων των στηλών μιας σχέσης καλείται *σχήμα* της σχέσης.

Στα πλαίσια της ΣΑΧΕ ορίζονται δύο νέοι τύποι δεδομένων ([ESPRIT93α]) για την αναπαράσταση του χρόνου, το *χρονικό σημείο* (time point) και το *χρονικό διάστημα* (time interval), σχέσεις πάνω σε στιγμιότυπα που ανήκουν στους τύπους αυτούς, καθώς και τελεστές που εφαρμόζονται σε σχέσεις που περιέχουν δεδομένα αυτών των τύπων. Στις επόμενες παραγράφους περιγράφονται οι δύο νέοι τύποι δεδομένων, οι σχέσεις ανάμεσα σε δεδομένα των τύπων αυτών, καθώς και οι νέοι τελεστές.

2.1. Οι τύποι δεδομένων χρονικό σημείο και χρονικό διάστημα.

Ο τύπος δεδομένων *χρονικό σημείο* αναπαριστά μία στιγμή του χρόνου. Κάθε χρονικό σημείο έχει ορισμένη *διακριτότητα* (granularity), η οποία καθορίζει την ακρίβεια με την οποία είναι γνωστή η χρονική στιγμή που αναπαριστά (ή την ακρίβεια με την οποία επιθυμούμε να καταχωρήσουμε αυτή τη γνώση). Για παράδειγμα, η χρονική στιγμή κατασκευής ενός αγγείου που βρέθηκε σε κάποια αρχαιολογική ανασκαφή προσδιορίζεται συνήθως με ακρίβεια αιώνα, ενώ η ημερομηνία γέννησης ενός ανθρώπου καταγράφεται με ακρίβεια ημέρας. Για τις σταθερές τύπου *χρονικό σημείο* θα χρησιμοποιείται ο συμβολισμός

έτος-μήνας-ημέρα ώρα:λεπτό:δευτερόλεπτο.κλάσματα_δευτερολέπτου

με το τμήμα του συμβολισμού που δεν καλύπτεται από την επιθυμητή διακριτότητα να παραλείπεται. (Π.χ. για το χρονικό σημείο *Μάρτιος 1821*, που έχει διακριτότητα ημέρας, θα χρησιμοποιείται ο συμβολισμός *1821-03*, ενώ για το χρονικό σημείο *28 Οκτωβρίου 1940, 7:20μμ* θα χρησιμοποιείται ο συμβολισμός *1940-10-28 19:20*).

Στο σύνολο των χρονικών σημείων *της ίδιας διακριτότητας* $ΧΣ_δ$ ορίζεται η σχέση της *ισότητας*, καθώς και η σχέση *προηγείται χρονικά* ως ακολούθως:

1. ορίζουμε ότι δύο χρονικά σημεία $χ_ι$ και $χ_κ$ είναι ίσα και συμβολίζουμε με $χ_ι = χ_κ$, τότε και μόνον τότε αν αναπαριστούν την ίδια χρονική στιγμή.

2. ορίζουμε ότι το χρονικό σημείο χ_ι προηγείται χρονικά του σημείου χ_κ. και συμβολίζουμε με $\chi_\iota < \chi_\kappa$, τότε και μόνον τότε αν η χρονική στιγμή που αναπαριστά το σημείο χ_ι είναι προγενέστερη της χρονικής στιγμής που αναπαριστά το σημείο χ_κ.

Η σχέση < είναι μια σχέση ολικής διάταξης, δηλαδή για κάποιο ζεύγος χρονικών σημείων της ίδιας διακριτότητας χ_ι και χ_κ ισχύει ότι

είτε $\chi_\iota < \chi_\kappa$, είτε $\chi_\iota = \chi_\kappa$, είτε $\chi_\kappa < \chi_\iota$

Σύμφωνα με τη σχέση <, στο σύνολο ΧΣ_δ μπορούμε να ορίσουμε τη σχέση *αμέσως επόμενο*. Ορίζουμε ότι το χ_κ είναι το *αμέσως επόμενο* του χ_ι, τότε και μόνο τότε αν:

1. $\chi_\iota < \chi_\kappa$.
2. δεν υπάρχει $\chi_\lambda \in \text{ΧΣ}_\delta$ τέτοιο ώστε $\chi_\iota < \chi_\lambda$ και $\chi_\lambda < \chi_\kappa$.

Για το αμέσως επόμενο ενός χρονικού σημείου χ_ι θα χρησιμοποιείται ο συμβολισμός $\chi_{\iota+1}$. Επίσης, το αμέσως επόμενο του χρονικού σημείου $\chi_{\iota+1}$ θα συμβολίζεται με $\chi_{\iota+2}$, κ.ο.κ.

Στο σύνολο ΧΣ_δ ορίζονται επίσης οι σχέσεις ≤, >, ≥ και *αμέσως προηγούμενο*, σύμφωνα με τα ακόλουθα:

- $\chi_\iota \le \chi_\kappa \Leftrightarrow \chi_\iota < \chi_\kappa$ ή $\chi_\iota = \chi_\kappa$.
- $\chi_\iota > \chi_\kappa \Leftrightarrow \chi_\kappa < \chi_\iota$.
- $\chi_\iota \ge \chi_\kappa \Leftrightarrow \chi_\iota > \chi_\kappa$ ή $\chi_\iota = \chi_\kappa$.
- χ_ι *αμέσως προηγούμενο* $\chi_\kappa \Leftrightarrow \chi_\kappa$ *αμέσως επόμενο* χ_ι.

Κατ' αντιστοιχία με το συμβολισμό $\chi_{\iota+1}$ που χρησιμοποιείται για το αμέσως επόμενο του χρονικού σημείου χ_ι, για το αμέσως προηγούμενο θα χρησιμοποιείται ο συμβολισμός $\chi_{\iota-1}$. Επίσης, για το προηγούμενο του χρονικού σημείου $\chi_{\iota-1}$ θα χρησιμοποιείται ο συμβολισμός $\chi_{\iota-2}$ κ.ο.κ.

Σημειώνουμε ότι οι σχέσεις <, ≤, >, ≥, *επόμενο* και *προηγούμενο* δεν ορίζονται για χρονικά σημεία διαφορετικής διακριτότητας. Έτσι, για τα χρονικά σημεία "1990-01" (Ιανουάριος του 1990, που είναι χρονικό σημείο με διακριτότητα μήνα) και "1990-01-11" (η 11η Ιανουαρίου του 1990, που είναι χρονικό σημείο με διακριτότητα ημέρας) δεν ορίζονται (και άρα δεν έχουν τιμή *αληθές* ή *ψευδές*) οι σχέσεις "1990-01" < "1990-01-11", "1990-01" > "1990-01-11", "1990-01" ≤ "1990-01-11", "1990-01" ≥ "1990-01-11", "1990-01" *αμέσως επόμενο* "1990-01-11" και "1990-01" *αμέσως προηγούμενο* "1990-01-11".

Ο τύπος δεδομένων *χρονικό διάστημα* παριστά μία χρονική περίοδο με συγκεκριμένη αρχή και τέλος[1]. Ένα στιγμιότυπο του τύπου *χρονικό διάστημα* ορίζεται βάσει ενός διατεταγμένου ζεύγους χρονικών σημείων (χ_λ, χ_κ). Το χρονικό σημείο χ_λ καλείται *αρχή* του διαστήματος, το χ_κ *πέρας*, ενώ ο όρος *άκρα* θα χρησιμοποιείται για να αναφερθούμε στο ζεύγος. Τα άκρα χ_λ και χ_κ έχουν την ίδια διακριτότητα και πληρούν τον περιορισμό $\chi_\lambda < \chi_\kappa$. Το χρονικό διάστημα που ορίζεται από το διατεταγμένο ζεύγος (χ_λ, χ_κ) συμβολίζεται με $[\chi_\lambda, \chi_\kappa)$ και περιλαμβάνει τα χρονικά σημεία χ_λ, $\chi_{\lambda+1}$, $\chi_{\lambda+2}$,..., $\chi_{\kappa-1}$. Ορίζουμε ότι η διακριτότητα ενός χρονικού διαστήματος είναι ίση με τη διακριτότητα των άκρων του. Το χρονικό διάστημα $[\chi_\iota, \chi_{\iota+1})$ περιλαμβάνει μόνο ένα χρονικό σημείο το χ_ι και καλείται *εκφυλισμένο χρονικό διάστημα*.

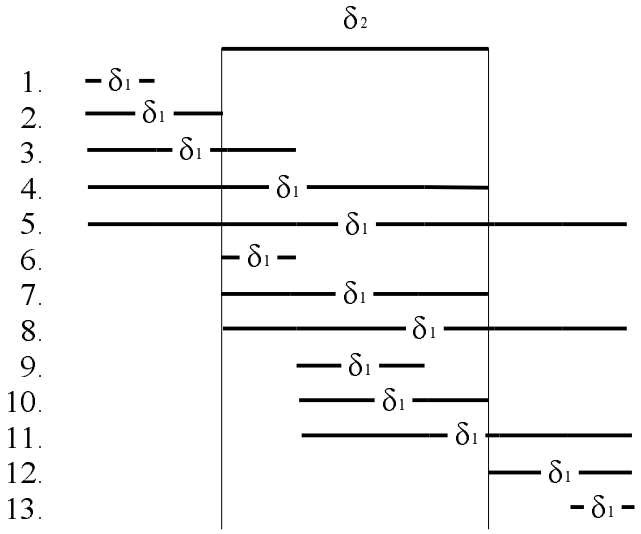
Δεδομένου ότι το *χρονικό διάστημα* είναι στην πραγματικότητα ένα σύνολο από χρονικά σημεία, του οποίου τα στοιχεία πληρούν έναν επιπρόσθετο περιορισμό (είναι διαδοχικά στοιχεία του συνόλου ΧΣ_δ, σύμφωνα με τη σχέση *επόμενο*), οι σχέσεις *ίσο*, *υποσύνολο*, *γνήσιο υποσύνολο*, *υπερσύνολο* και *γνήσιο υπερσύνολο* που ορίζονται μεταξύ συνόλων, έχουν φυσική αντιστοίχιση στα χρονικά διαστήματα. Για τα χρονικά διαστήματα της ίδιας διακριτότητας $\delta_1 \equiv [\chi_\lambda, \chi_\kappa)$ και $\delta_2 \equiv [\chi_\lambda, \chi_\mu)$ ορίζουμε έτσι τις σχέσεις *ίσο* (=), *υπερδιάστημα* ($\supseteq$), *γνήσιο υπερδιάστημα* ($\supset$), *υποδιάστημα* ($\subseteq$) και γνήσιο υποδιάστημα ($\subset$) ως εξής:

- $\delta_1 = \delta_2 \Leftrightarrow \forall\ t: (t \in \delta_1 \Leftrightarrow t \in \delta_2)$
- $\delta_1 \supseteq \delta_2 \Leftrightarrow \forall\ t \in \delta_2,\ t \in \delta_1$
- $\delta_1 \supset \delta_2 \Leftrightarrow \delta_1 \supseteq \delta_2 \wedge \exists\ t \in \delta_1: t \notin \delta_2$
- $\delta_1 \subseteq \delta_2 \Leftrightarrow \delta_2 \supseteq \delta_1$
- $\delta_1 \subset \delta_2 \Leftrightarrow \delta_2 \supset \delta_1$

Για δύο διαστήματα δ_1 και δ_2 της ίδιας διακριτότητας μπορούμε να διακρίνουμε 13 σχετικές θέσεις, ανάλογα με τις σχετικές θέσεις των άκρων τους. Στο σχήμα 2.1 παρουσιάζονται παραστατικά οι σχετικές αυτές θέσεις.

---

[1]Η απαίτηση να έχει το χρονικό διάστημα συγκεκριμένη αρχή και τέλος επιβάλλεται, προκειμένου να διαχωριστεί ο τύπος χρονικό διάστημα από τον τύπο *χρονική περίοδος* (period), ο οποίος αναπαριστά μία χρονική διάρκεια, χωρίς συγκεκριμένη αρχή ή τέλος, π.χ. 60 χρόνια. Ο τύπος *χρονική περίοδος* δεν θα μας απασχολήσει στη συνέχεια.



Σχήμα 2.1 - Σχετικές θέσεις διαστημάτων.

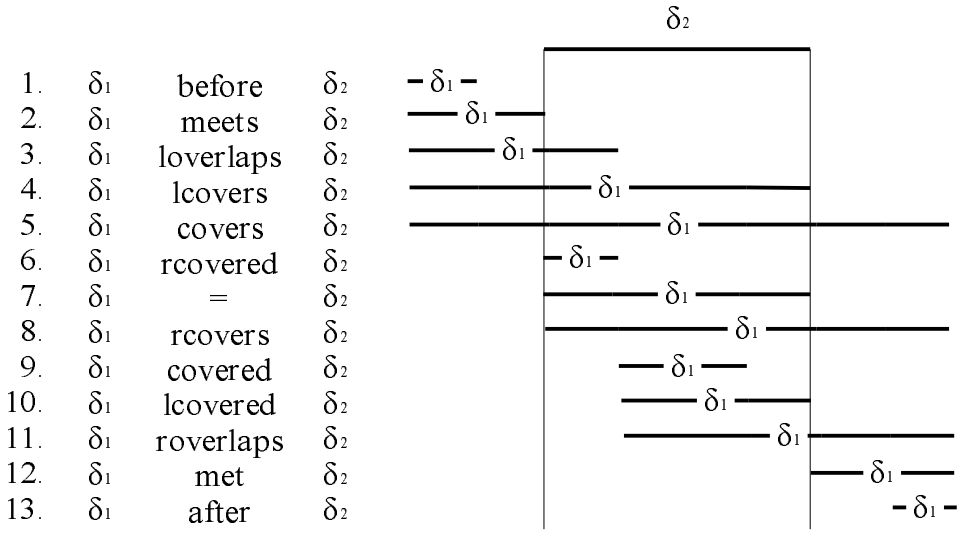
Ο Allen ([Allen83]) έδειξε ότι για να είναι δυνατή η εξέταση της σχετικής θέσης δύο διαστημάτων, αρκούν επτά κατηγορήματα, ενώ οι υπόλοιπες περιπτώσεις μπορούν να εκφραστούν συναρτήσει των κατηγορημάτων αυτών. Ως βασικά κατηγορήματα ο Allen πρότεινε τα δ_1 *before* δ_2 (περίπτωση 1), δ_1 *equals* δ_2 (περίπτωση 7), δ_1 *meets* δ_2 (περίπτωση 2), δ_1 *overlaps* δ_2 (περίπτωση 3), δ_1 *during* δ_2 (περίπτωση 9), δ_1 *starts* δ_2 (περίπτωση 6) και δ_1 *finishes* δ_2 (περίπτωση 10). Υπάρχουν και άλλες προτάσεις σχετικά με το ποιες περιπτώσεις πρέπει να θεωρηθούν βασικές και ποιες παράγωγες, ενώ σε άλλες προτάσεις παραβιάζεται η αρχή της *ελαχιστότητας* (minimality) του συνόλου των κατηγορημάτων, προς όφελος της *εκφραστικότητας* (expressiveness), δηλαδή της ευκολίας διατύπωσης από τον χρήστη του επιθυμητού ελέγχου. Στο [Snodgrass94γ] για να καλυφθούν όλες οι περιπτώσεις χρησιμοποιούνται πέντε κατηγορήματα για χρονικά διαστήματα, σε συνδυασμό με τις σχέσεις <, = και > των χρονικών σημείων, καθώς και δύο συναρτήσεις *begin* και *end*, οι οποίες δέχονται ως ορίσματα χρονικά διαστήματα και επιστρέφουν την αρχή και το πέρας του ορίσματός τους, αντίστοιχα. Τέλος, αν το προτεινόμενο αλγεβρικό μοντέλο πρόκειται να υλοποιηθεί, μία παράμετρος που πρέπει να ληφθεί υπόψη κατά την επιλογή των βασικών κατηγορημάτων είναι η συχνότητα της χρήσης τους: κατηγορήματα που χρησιμοποιούνται συχνά, είναι σκόπιμο να είναι άμεσα διαθέσιμα, παρά να εκφράζονται ως συνδυασμός άλλων, προκειμένου να είναι ταχύτερη η αποτίμησή τους από τον υπολογιστή. Σύμφωνα με αυτό το κριτήριο, η πρόταση του Allen δεν είναι η καλύτερη επιλογή, διότι δεν περιέχει κάποιο κατηγόρημα που να αληθεύει όταν τα ορίσματά του προσδιορίζουν διαστήματα που έχουν τουλάχιστον ένα κοινό χρονικό σημείο, μία σύγκριση που χρησιμοποιείται ιδιαίτερα

συχνά (η περίπτωση να έχουν δύο χρονικά διαστήματα τουλάχιστον ένα κοινό χρονικό σημείο δεν αντιστοιχεί σε κάποια από τις 13 περιπτώσεις του σχήματος 2.1, αλλά είναι η διάζευξη των περιπτώσεων 3 έως 11). Η σύγκριση αυτή, με τα κατηγορήματα του Allen θα γραφόταν

NOT (*δ1* before *δ2* OR *δ2* before *δ1* OR *δ1* meets *δ2* OR *δ2* meets *δ1*)

που για την αποτίμησή της χρειάζεται (στη χειρότερη περίπτωση) η αποτίμηση τεσσάρων βασικών κατηγορημάτων και μίας άρνησης, με σημαντικές επιπτώσεις στην απόδοση του συστήματος.

Είναι πέρα από τους σκοπούς της παρούσας έρευνας να εξαχθούν στατιστικά στοιχεία για τη συχνότητα χρήσης των διαφόρων κατηγορημάτων και να αναλυθεί η συμβολή του κάθε κατηγορήματος στην εκφραστικότητα της γλώσσας ερωταποκρίσεων, προκειμένου να βρεθεί ένα ακόμη ελάχιστο (ή μη) σύνολο τελεστών που θα είναι αρκετό για να εκφράσει όλες τις δυνατές σχέσεις μεταξύ δύο χρονικών διαστημάτων. Στη συνέχεια θα θεωρήσουμε ότι υπάρχει ένα κατηγόρημα για όλες τις περιπτώσεις που απεικονίζονται στο σχήμα 2.2. Οι περιπτώσεις αυτές δεν συμπεριλαμβάνουν μόνο τις 13 σχετικές θέσεις που παρουσιάζονται στο σχήμα 2.1, αλλά και συνδυασμούς αυτών.



14 δ1 `psubinterv` δ2 Το κατηγόρημα αυτό αληθεύει όταν το `δ1` είναι γνήσιο υποδιάστημα του `δ2` (αληθεύει μία από τις συνθήκες 6, 9 και 10).

15 δ1 `subinterv` δ2 Το κατηγόρημα αυτό αληθεύει όταν το `δ1` είναι υποδιάστημα του `δ2` (αληθεύει μία από τις συνθήκες 6, 9, 7 και 10).

(Συνεχίζεται στην επόμενη σελίδα)

(Συνέχεια από την προηγούμενη σελίδα)

16	δ1	psupinterv	δ2	Το κατηγόρημα αυτό αληθεύει όταν το δ1 είναι γνήσιο υπερδιάστημα του δ2 (αληθεύει μία από τις συνθήκες 4, 5 και 8).
17	δ1	supinterv	δ2	Το κατηγόρημα αυτό αληθεύει όταν το δ1 είναι υπερδιάστημα του δ2 (αληθεύει μία από τις συνθήκες 4, 5, 7 και 8).
18	δ1	cp	δ2	Το κατηγόρημα αυτό αληθεύει όταν τα δ1 και δ2 επικαλύπτονται ή, τυπικά, $\exists$ t $\in$ δ1: t $\in$ δ2 (μία από τις συνθήκες 3, 4, 5, 6, 7, 8, 9, 10 και 11 είναι αληθής).
19	δ1	adjacent	δ2	Το κατηγόρημα αυτό είναι αληθές όταν το πέρας του δ1 ισούται με την αρχή του δ2 ή το πέρας του δ2 ισούται με την αρχή του δ1 (αληθεύει μία από τις συνθήκες 2 και 12).
20	δ1	overlaps	δ2	Το κατηγόρημα αυτό αληθεύει όταν τα διαστήματα δ1 και δ2 έχουν κοινά σημεία, αλλά κανένα από τα δύο δεν είναι υποδιάστημα του άλλου (αληθεύει μία από τις συνθήκες 3 και 11).
21	δ1	merges	δ2	Το κατηγόρημα αυτό αληθεύει όταν τα διαστήματα δ1 και δ2 είναι επικαλυπτόμενα, ή η αρχή του ενός ισούται με το πέρας του άλλου (μία από τις συνθήκες 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 και 12 είναι αληθής).
22	δ1	prequals	δ2	Το κατηγόρημα αυτό αληθεύει όταν το η αρχή του διαστήματος δ1 προηγείται ή είναι ίση της αρχής του δ2 και το πέρας του δ1 προηγείται ή είναι ίσο του πέρατος του δ2 (μία από τις συνθήκες 1, 2, 3, 4, 6 και 7 είναι αληθής).

(Συνεχίζεται στην επόμενη σελίδα)

(Συνέχεια από την προηγούμενη σελίδα)

23	δ1	`precedes`	δ2	Το κατηγόρημα αυτό είναι ισοδύναμο με το κατηγόρημα `prequals`, εκτός της περιπτώσεως που τα ορίσματα είναι ίσα, κατά την οποία το κατηγόρημα `prequals` αληθεύει, σε αντίθεση με το κατηγόρημα `precedes` (μία από τις συνθήκες 1, 2, 3, 4 και 6 είναι αληθής).
24	δ1	`folequals`	δ2	Το κατηγόρημα αυτό αληθεύει όταν η αρχή του διαστήματος `δ1` έπεται ή είναι ίση της αρχής του `δ2` και το πέρας του `δ1` έπεται ή είναι ίσο του πέρατος του `δ2` (μία από τις συνθήκες 7, 8, 10, 11, 12 και 13 είναι αληθής).
25	δ1	`follows`	δ2	Το κατηγόρημα αυτό είναι ισοδύναμο με το κατηγόρημα `folequals`, εκτός της περιπτώσεως που τα ορίσματα είναι ίσα, κατά την οποία το κατηγόρημα `folequals` αληθεύει, σε αντίθεση με το κατηγόρημα `follows` (μία από τις συνθήκες 8, 10, 11, 12 και 13 είναι αληθής).

Σχήμα 2.2 - Τα κατηγορήματα σύγκρισης διαστημάτων της ΣΑΧΕ.

2.2. Πράξεις της ΣΑΧΕ.

Στις επόμενες παραγράφους περιγράφονται οι πράξεις της σχεσιακής άλγεβρας χρόνου εγκυρότητας. Οι πράξεις της *προβολής* (projection), της *ένωσης* (union), της *διαφοράς* (difference) και της *σύνδεσης* (join) δεν διαφέρουν από τις ομώνυμες πράξεις της σχεσιακής άλγεβρας, οπότε η περιγραφή τους είναι σύντομη. Πιο αναλυτική περιγραφή δίνεται για τις πράξεις του *υπολογισμού* (computation), της *σύμπτυξης* (folding), της *ανάπτυξης* (unfolding), της *κανονικοποίησης* (normalisation), της *σημειακής ένωσης* (point union) και της *σημειακής διαφοράς* (point difference), οι οποίες είναι νέες, καθώς και για την πράξη της *επιλογής* (selection), η οποία επεκτείνεται.

2.2.1. Επιλογή.

Η πράξη της επιλογής στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = SELECT [κριτήριο] (πίνακας)

όπου *αποτέλεσμα* είναι η σχέση στην οποία θα τοποθετηθούν οι πλειάδες της σχέσης *πίνακας* που πληρούν τη συνθήκη *κριτήριο*. Το σχήμα της σχέσης *αποτέλεσμα* είναι ίδιο με το σχήμα της σχέσης *πίνακας*.

Η ΣΑΧΕ επεκτείνει την πράξη της επιλογής της σχεσιακής άλγεβρας, επιτρέποντας σε κατηγορήματα σύγκρισης διαστημάτων καθώς και στους τελεστές σύγκρισης χρονικών σημείων να εμφανίζονται στη συνθήκη επιλογής. Τα κατηγορήματα και οι τελεστές μπορούν να χρησιμοποιούνται σε οποιοδήποτε σημείο επιτρέπεται η εμφάνιση μιας συνθήκης της σχεσιακής άλγεβρας, σε μία πράξη επιλογής.

Στο σχήμα 2.3 φαίνεται μία πράξη επιλογής που χρησιμοποιεί ένα κατηγόρημα σύγκρισης διαστημάτων. Το αποτέλεσμα *M1* αποτελείται από τις πλειάδες της σχέσης *Μισθός* οι οποίες περιέχουν στοιχεία για το μισθό του εργαζόμενου *Γιάννης* κατά τη χρονική περίοδο $[\chi_3, \chi_{10})$.

Μισθός

Όνομα	Μισθός	Διάστημα
Γιάννης	100000	$[\chi_1, \chi_3)$
Γιάννης	120000	$[\chi_5, \chi_8)$
Γιάννης	130000	$[\chi_{10}, \chi_{15})$
Πέτρος	80000	$[\chi_2, \chi_5)$
Πέτρος	120000	$[\chi_8, \chi_{11})$

M1 = SELECT [Όνομα = 'Γιάννης' AND Διάστημα cp '$[\chi_3, \chi_{11})$'] (Μισθός)

Όνομα	Μισθός	Διάστημα
Γιάννης	120000	$[\chi_5, \chi_8)$
Γιάννης	130000	$[\chi_{10}, \chi_{15})$

Σχήμα 2.3 - Η πράξη επιλογής της ΣΑΧΕ.

2.2.2. Προβολή.

Η πράξη της προβολής στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = PROJECT [στήλη$_1$, στήλη$_2$, …, στήλη$_v$] (πίνακας)

όπου *αποτέλεσμα* είναι μία σχέση, η οποίος αποτελείται μόνο από τις στήλες $στήλη_1$, $στήλη_2$, …, $στήλη_v$ της σχέσης *πίνακας*. Στη θέση μιας στήλης μπορεί να χρησιμοποιηθεί ο συμβολισμός

ΝέοΌνομα = στήλη$_i$

που έχει ως αποτέλεσμα να επιλέγεται η στήλη $στήλη_i$, αλλά να εμφανίζεται στον πίνακα *αποτέλεσμα* με την ονομασία *ΝέοΌνομα*.

Αν στο σύνολο στηλών $στήλη_1$, $στήλη_2$, …, $στήλη_v$ δεν συμπεριλαμβάνονται όλες οι στήλες της σχέσης *πίνακας*, τότε ενδέχεται η αφαίρεση στηλών να δημιουργήσει πλειάδες των

οποίων όλες οι στήλες είναι ίσες. Το τελικό αποτέλεσμα θα περιλαμβάνει μόνο μία από τις πλειάδες αυτές.

2.2.3. Ένωση.

Η πράξη της ένωσης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = Σ UNION Π

όπου Σ και Π είναι δύο *συμβατές ως προς την ένωση* (union compatible) σχέσεις.

Δύο σχέσεις Σ και Π των οποίων τα σχήματα είναι $(\sigma_1, \sigma_2, \ldots, \sigma_\nu)$ και $(\pi_1, \pi_2, \ldots, \pi_\mu)$ καλούνται *συμβατές ως προς την ένωση* όταν ισχύουν τα ακόλουθα:

1. $\nu = \mu$
2. $\forall\ \iota = 1, 2, \ldots, \nu$: το πεδίο ορισμού της στήλης σ_ι είναι ίδιο με το πεδίο ορισμού της στήλης π_ι.

Το σχήμα της σχέσης *αποτέλεσμα* που προκύπτει από την εφαρμογή της πράξης *ένωση* είναι ίδιο με το σχήμα της σχέσης Σ. Η σχέση *αποτέλεσμα* περιέχει τόσο τις πλειάδες της σχέσης Σ, όσο και αυτές της σχέσης Π. Αν μία πλειάδα περιέχεται και στις δύο σχέσεις, τότε εμφανίζεται μόνο μία φορά στο αποτέλεσμα.

2.2.4. Διαφορά.

Η πράξη της διαφοράς στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = Σ EXCEPT Π

όπου Σ και Π είναι δύο συμβατές ως προς την ένωση σχέσεις. Το σχήμα της σχέσης *αποτέλεσμα* που προκύπτει από την εφαρμογή της πράξης *διαφορά* είναι ίδιο με το σχήμα της σχέσης Σ. Η σχέση *αποτέλεσμα* περιέχει τις πλειάδες που εμφανίζονται στη σχέση Σ, αλλά δεν εμφανίζονται στη σχέση Π.

Η πράξη της διαφοράς χρησιμοποιείται για να εκφρασθεί και η πράξη της *τομής*, μια και

$\Sigma \cap \Pi = \Sigma$ EXCEPT $(\Sigma$ EXCEPT $\Pi)$

2.2.5. Σύνδεση.

Η πράξη της σύνδεσης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = Σ CP Π

και δημιουργεί τη σχέση *αποτέλεσμα*, η οποία περιέχει το καρτεσιανό γινόμενο των σχέσεων Σ και Π. Αν τα σχήματα των σχέσεων Σ και Π είναι $(\sigma_1, \sigma_2, \ldots, \sigma_\nu)$ και $(\pi_1, \pi_2, \ldots, \pi_\mu)$, αντίστοιχα, τότε το σχήμα της σχέσης *αποτέλεσμα* είναι $(\Sigma.\sigma_1, \Sigma.\sigma_2, \ldots, \Sigma.\sigma_\nu, \Pi.\pi_1, \Pi.\pi_2, \ldots,$

Π.π_μ). Η προσθήκη του ονόματος της σχέσης προελεύσεως της κάθε στήλης γίνεται για να αποφευχθεί η εμφάνιση κάποιου ονόματος στήλης δύο φορές.

2.2.6. Υπολογισμός.

Η πράξη του *υπολογισμού* δίνει τη δυνατότητα να υπολογισθεί το αποτέλεσμα κάποιας παράστασης. Η παράσταση μπορεί να περιέχει μαθηματικές συναρτήσεις σε συνδυασμό με τελεστές που έχουν ως αποτέλεσμα ακέραιους ή πραγματικούς, συναρτήσεις υπολογισμού συμβολοσειρών ή συναρτήσεις *χρονικών σημείων* και *χρονικών διαστημάτων*, καθώς και συνδυασμούς αυτών. Η πράξη του υπολογισμού στη ΣΑΧΕ συμβολίζεται με

COMPUTE(παράσταση)

και μπορεί να χρησιμοποιηθεί τόσο στη συνθήκη της επιλογής, όσο και ως δεξί μέρος του συμβολισμού

ΝέοΌνομα = στήλη$_\iota$

στην πράξη της προβολής.

Οι συναρτήσεις *χρονικών σημείων* και *χρονικών διαστημάτων* ορίζονται στα πλαίσια της ΣΑΧΕ και δέχονται ως ορίσματα ή/και επιστρέφουν αποτελέσματα των οποίων ο τύπος είναι χρονικό σημείο ή χρονικό διάστημα. Οι συναρτήσεις αυτές περιγράφονται στις επόμενες παραγράφους.

1. **dist(χ_κ, χ_λ)**

Η συνάρτηση *dist* δέχεται δύο ορίσματα χ_κ και χ_λ, τα οποία πρέπει να είναι τύπου *χρονικό σημείο* και της ίδιας διακριτότητας και επιστρέφει την τιμή $|\kappa - \lambda|$, που αντιστοιχεί στον αριθμό των χρονικών σημείων που μεσολαβούν μεταξύ των ορισμάτων της. Το αποτέλεσμα είναι πάντα ένας μη αρνητικός ακέραιος, ανεξάρτητα από τη σχετική θέση του χ_κ ως προς το χ_λ.

2. **dur(δ_1)**

Η συνάρτηση *dur* δέχεται ένα όρισμα δ_1, το οποίο πρέπει να είναι τύπου *χρονικό διάστημα*. Αν $\delta_1 = [\chi_\kappa, \chi_\lambda)$, η συνάρτηση επιστρέφει την τιμή $(\lambda - \kappa)$, που αντιστοιχεί στον αριθμό των χρονικών σημείων που περιλαμβάνονται στο διάστημα δ_1. Το αποτέλεσμα είναι ένας θετικός ακέραιος.

3. **interv(χ_κ, χ_λ)**

Η συνάρτηση *interv* δέχεται δύο ορίσματα χ_κ και χ_λ τύπου *χρονικό σημείο*, τα οποία πρέπει να είναι της ίδιας διακριτότητας και το χ_λ να έπεται του χ_κ. Το αποτέλεσμα της συνάρτησης

είναι το χρονικό διάστημα $[\chi_\kappa, \chi_\lambda)$. Αν τα ορίσματα είναι διαφορετικής διακριτότητας ή το χ_λ δεν έπεται του χ_κ, η συνάρτηση δεν ορίζεται.

4. intervsect(δ_1, δ_2)

Η συνάρτηση *intervsect* δέχεται δύο ορίσματα δ_1 και δ_2, τα οποία πρέπει να είναι τύπου *χρονικό διάστημα* και της ίδιας διακριτότητας. Το αποτέλεσμα της συνάρτησης είναι ένα *χρονικό διάστημα*, της ίδιας διακριτότητας με τα ορίσματά της, το οποίο περιλαμβάνει τα χρονικά σημεία που ανήκουν και στα δύο ορίσματα. Αν τα ορίσματα είναι διαφορετικής διακριτότητας ή δεν υπάρχουν χρονικά σημεία που να ανήκουν από κοινού στα δ_1 και δ_2 (ισοδύναμα, αν τα διαστήματα δεν είναι επικαλυπτόμενα), τότε η συνάρτηση δεν ορίζεται.

5. merge(δ_1, δ_2)

Η συνάρτηση *merge* δέχεται δύο ορίσματα δ_1 και δ_2, τα οποία πρέπει να είναι τύπου *χρονικό διάστημα* και της ίδιας διακριτότητας. Το αποτέλεσμα της συνάρτησης είναι ένα χρονικό διάστημα, της ίδιας διακριτότητας με τα ορίσματά της, το οποίο περιλαμβάνει όλα τα χρονικά σημεία που περιέχονται είτε στο πρώτο είτε στο δεύτερο όρισμα. Αν τα χρονικά σημεία δεν μπορούν να συμπεριληφθούν σε ένα μόνο χρονικό διάστημα (γεγονός που θα συμβαίνει όταν μία από τις συνθήκες δ_1 *before* δ_2 και δ_1 *after* δ_2 αληθεύει), η συνάρτηση δεν ορίζεται.

6. middle(δ_1)

Η συνάρτηση *middle* δέχεται ως όρισμα ένα χρονικό διάστημα $\delta_1 = [\chi_\kappa, \chi_\lambda)$ και επιστρέφει το χρονικό σημείο $\chi_{\kappa+(\lambda-\kappa-1)/2}$, το οποίο αντιστοιχεί στο μέσον του διαστήματος δ_1.

7. span(χ_1, χ_2)

Η συνάρτηση *span* συμπεριφέρεται όπως και η συνάρτηση *dist*, με τη διαφορά ότι το αποτέλεσμα μπορεί να είναι αρνητικό, ανάλογα με τη σχετική θέση των ορισμάτων. Πιο συγκεκριμένα το αποτέλεσμα θα είναι θετικός, αν το σημείο χ_1 προηγείται του σημείου χ_2, 0 αν τα ορίσματα είναι ίσα και αρνητικός αν το σημείο χ_1 έπεται του σημείου χ_2.

8. start(δ_1)

Η συνάρτηση *start* δέχεται ως όρισμα ένα χρονικό διάστημα $\delta_1 = [\chi_\kappa, \chi_\lambda)$ και επιστρέφει το χρονικό σημείο χ_κ, δηλαδή την αρχή του ορίσματος.

9. stop(δ_1)

Η συνάρτηση *stop* δέχεται ως όρισμα ένα χρονικό διάστημα $\delta_1 = [\chi_\kappa, \chi_\lambda)$ και επιστρέφει το χρονικό σημείο χ_λ, δηλαδή το πέρας του ορίσματος.

10. succ(χ_κ, ι)

Η συνάρτηση *succ* δέχεται ως όρισμα ένα χρονικό σημείο χ_κ και έναν ακέραιο ι και επιστρέφει το χρονικό σημείο $\chi_{\kappa+\iota}$. Το αποτέλεσμα θα προηγείται, έπεται ή θα είναι ίσο του χ_κ, ανάλογα με το αν ο ακέραιος ι είναι αρνητικός, θετικός ή 0, αντίστοιχα.

11. tointerv(χ_κ)

Η συνάρτηση *tointerv* δέχεται ως όρισμα ένα χρονικό σημείο χ_κ και επιστρέφει το εκφυλισμένο χρονικό διάστημα $[\chi_\kappa, \chi_{\kappa+1})$, το οποίο είναι της ίδιας διακριτότητας με το όρισμα της συνάρτησης.

12. topoint(δ_1)

Η συνάρτηση *topoint* δέχεται ως όρισμα ένα χρονικό διάστημα $\delta_1 = [\chi_\kappa, \chi_\lambda)$. Αν το όρισμα είναι ένα εκφυλισμένο χρονικό διάστημα (δηλαδή $\chi_\lambda = \chi_{\kappa+1}$), τότε η συνάρτηση επιστρέφει την αρχή του διαστήματος, η οποία είναι της ίδιας διακριτότητας με το όρισμα. Στην αντίθετη περίπτωση, η συνάρτηση δεν ορίζεται.

13. window(χ_κ, λ, μ)

Η συνάρτηση *window* δέχεται τρία ορίσματα, το πρώτο από τα οποία είναι ένα χρονικό σημείο χ_κ, το δεύτερο είναι ένας θετικός ακέραιος λ και το τρίτο ένας μη μηδενικός ακέραιος μ. Αν το τρίτο όρισμα είναι θετικό, η συνάρτηση επιστρέφει ένα χρονικό διάστημα, της ίδιας διακριτότητας με το πρώτο όρισμα, η αρχή του οποίου είναι το χρονικό σημείο $\chi_{\kappa+\lambda*(\mu-1)}$ και το πέρας του είναι το χρονικό σημείο $\chi_{\kappa+\lambda*\mu}$. Αν το τρίτο όρισμα είναι αρνητικό, η συνάρτηση επιστρέφει ένα χρονικό διάστημα, η αρχή του οποίου είναι το χρονικό σημείο $\chi_{\kappa+\lambda*\mu+1}$, και το πέρας του είναι το χρονικό σημείο $\chi_{\kappa+\lambda*(\mu+1)+1}$.

14. windowno(χ_κ, ι, χ_λ)

Η συνάρτηση *windowno* δέχεται τρία ορίσματα, το πρώτο από τα οποία είναι ένα χρονικό σημείο χ_κ, το δεύτερο είναι ένας θετικός ακέραιος ι και το τρίτο είναι ένα χρονικό σημείο χ_λ, της ίδιας διακριτότητας με το πρώτο όρισμα. Το αποτέλεσμα της συνάρτησης θα είναι:

- $\left\lfloor \frac{\lambda - \kappa}{\iota} \right\rfloor$, αν $\lambda \geq \kappa$.
- $-\left\lfloor \frac{\kappa - \lambda + \iota - 1}{\iota} \right\rfloor$, αν $\lambda < \kappa$.

Αν το δεύτερο όρισμα είναι 0 ή αρνητικό, τότε η συνάρτηση δεν ορίζεται.

2.2.7. Σύμπτυξη.

Η πράξη της σύμπτυξης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = FOLD [στήλη_σύμπτυξης] (πίνακας)

όπου *πίνακας* είναι μία υπάρχουσα σχέση, της οποίας το σχήμα είναι $(\sigma_1, \sigma_2, ..., \sigma_\nu)$, και *στήλη_σύμπτυξης* = σ_ι, για κάποιο ι $(1 \leq \iota \leq \nu)$. Ο τύπος της στήλης *στήλη_σύμπτυξης* θα πρέπει να είναι *χρονικό σημείο* ή *χρονικό διάστημα*. Το σχήμα της σχέσης *αποτέλεσμα*, που προκύπτει ως εξαγόμενο της εφαρμογής της πράξης σύμπτυξης θα είναι ίδιο με το σχήμα της σχέσης *πίνακας*, με τη διαφορά ότι ο τύπος της στήλης *Αποτέλεσμα*.σ_ι θα είναι *χρονικό διάστημα*, της ίδιας διακριτότητας με τη στήλη *Πίνακας*.σ_ι, ανεξάρτητα από το αν ο τύπος της στήλης *Πίνακας*.σ_ι είναι *χρονικό σημείο* ή *χρονικό διάστημα*.

Δύο πλειάδες π_1 και π_2 της σχέσης *πίνακας* καλούνται *συμπτύξιμες ως προς τη στήλη* σ_ι, τότε και μόνον τότε αν ισχύουν οι ακόλουθες συνθήκες:

1) $\pi_1.\sigma_\iota$ *cp* $\pi_2.\sigma_\iota \vee \pi_1.\sigma_\iota$ *adjacent* $\pi_2.\sigma_\iota$

2) $\pi_1.\sigma_\kappa = \pi_2.\sigma_\kappa, \forall\ \kappa \neq \iota$

(Όταν η στήλη σ_ι είναι τύπου *χρονικό σημείο*, η πρώτη συνθήκη τροποποιείται ως εξής:

1. $\pi_1.\sigma_\iota = \pi_2.\sigma_\iota \vee \pi_1.\sigma_\iota = (\pi_2.\sigma_\iota)_{+1} \vee \pi_1.\sigma_\iota = (\pi_2.\sigma_\iota)_{-1}$)

Η πράξη της σύμπτυξης αρχικά κατατάσσει τις πλειάδες της σχέσης *πίνακας* σε ομάδες. Δύο πλειάδες π_1 και π_2 της σχέσης *πίνακας* ανήκουν στην ίδια ομάδα τότε και μόνον τότε αν ισχύει μία από τις ακόλουθες συνθήκες:

1) οι πλειάδες π_1 και π_2 είναι συμπτύξιμες ως προς τη στήλη σ_ι

2) $\exists$ πλειάδα $\pi_3 \in$ *πίνακας*:

   i) οι πλειάδες π_1 και π_3 είναι συμπτύξιμες προς τη στήλη σ_ι

   ii) οι πλειάδες π_2 και π_3 ανήκουν στην ίδια ομάδα

Για κάθε ομάδα πλειάδων $(\pi_1, \pi_2, ..., \pi_\lambda)$ που θα σχηματιστεί, εισάγεται στη σχέση *αποτέλεσμα* μία μοναδική πλειάδα α, οι τιμές των στηλών της οποίας προσδιορίζονται ως εξής:

1. $\alpha.\sigma_\iota = merge(\pi_1.\sigma_\iota, merge(\pi_2.\sigma_\iota, merge(..., merge(\pi_{\lambda-1}.\sigma_\iota, \pi_\lambda.\sigma_\iota)...)))$

2. $\alpha.\sigma_\kappa = \pi_1.\sigma_\kappa, \forall\ \kappa \neq \iota$

(Δίχως βλάβη της γενικότητας και για να ορίζεται πάντα το αποτέλεσμα της παράστασης (1), θεωρούμε ότι οι πλειάδες μιας ομάδας είναι ταξινομημένες κατά αύξουσα διάταξη της αρχής του διαστήματος σ_ι).

Τυπικά, για τις σχέσεις *πίνακας* και *αποτέλεσμα* σε μία πράξη σύμπτυξης που εφαρμόζεται σε στήλη τύπου *χρονικό διάστημα* θα ισχύουν τα ακόλουθα:

1) $\forall (\tau_1, \tau_2, \ldots, \tau_\iota, \ldots, \tau_\nu) \in$ *πίνακας* $\exists (\upsilon_1, \upsilon_2, \ldots, \upsilon_\iota, \ldots, \upsilon_\nu) \in$ *αποτέλεσμα*: $\forall \kappa (1 \leq \kappa \leq \nu$ και $\kappa \neq \iota) \tau_\kappa = \upsilon_\kappa \wedge \tau_\iota$ *subinterv* υ_ι

2) για κάθε ζεύγος *διαφορετικών* πλειάδων $(\tau_1, \tau_2, \ldots, \tau_\iota, \ldots, \tau_\nu)$, $(\upsilon_1, \upsilon_2, \ldots, \upsilon_\iota, \ldots, \upsilon_\nu)$ της σχέσης *αποτέλεσμα* ισχύει τουλάχιστον μία από τις δύο ακόλουθες συνθήκες:

   i) $\exists \kappa (1 \leq \kappa \leq \nu$ και $\kappa \neq \iota)$: $\tau_\kappa \neq \upsilon_\kappa$

   ii) τ_ι *before* $\upsilon_\iota \vee \tau_\iota$ *after* υ_ι

Αν η πράξη της σύμπτυξης εφαρμόζεται πάνω σε στήλη τύπου *χρονικό σημείο*, η πρώτη συνθήκη τροποποιείται ως εξής:

1. $\forall (\tau_1, \tau_2, \ldots, \tau_\iota, \ldots, \tau_\nu) \in$ *πίνακας* $\exists (\upsilon_1, \upsilon_2, \ldots, \upsilon_\iota, \ldots, \upsilon_\nu) \in$ *αποτέλεσμα*: $\forall \kappa (1 \leq \kappa \leq \nu$ και $\kappa \neq \iota) \tau_\kappa = \upsilon_\kappa \wedge \tau_\iota \in \upsilon_\iota$

Η πρώτη συνθήκη εξασφαλίζει ότι δεν χάνεται πληροφορία για κανένα χρονικό σημείο κατά την πράξη της σύμπτυξης. Η δεύτερη συνθήκη εγγυάται ότι η σύμπτυξη παράγει τα μέγιστα δυνατά χρονικά διαστήματα στη στήλη σύμπτυξης.

Η πράξη της σύμπτυξης μπορεί να εφαρμοστεί και σε περισσότερες της μίας στήλες τύπου *χρονικό σημείο* ή *χρονικό διάστημα*. Η εφαρμογή αυτή συμβολίζεται με

αποτέλεσμα = FOLD [στήλη$_1$, στήλη$_2$, …, στήλη$_\mu$](πίνακας)

και είναι ισοδύναμη με την εφαρμογή της πράξης της σύμπτυξης πρώτα στη *στήλη$_1$*, κατόπιν στη *στήλη$_2$* κ.ο.κ. Η σειρά εφαρμογής των πράξεων σύμπτυξης είναι σημαντική, δεδομένου ότι έχει επίπτωση στο τελικό αποτέλεσμα.

Στο σχήμα 2.4 παρουσιάζεται ένα παράδειγμα εφαρμογής της πράξης της σύμπτυξης σε στήλη τύπου *χρονικό διάστημα*, ενώ στο σχήμα 2.5 παρουσιάζεται ένα παράδειγμα εφαρμογής της πράξης της σύμπτυξης σε στήλη τύπου *χρονικό σημείο*.

Εργαζόμενος

Όνομα	Τμήμα	Διάστημα
Νίκος	Ταμείο	$[\chi_1, \chi_8)$
Νίκος	Ταμείο	$[\chi_{10}, \chi_{20})$
Έλλη	Ταμείο	$[\chi_1, \chi_5)$
Έλλη	Ταμείο	$[\chi_3, \chi_8)$
Έλλη	Πωλήσεις	$[\chi_7, \chi_{11})$
Έλλη	Πωλήσεις	$[\chi_{11}, \chi_{15})$

Σ = Fold [Διάστημα](Εργαζόμενος)

Όνομα	Τμήμα	Διάστημα
Νίκος	Ταμείο	$[\chi_1, \chi_8)$
Νίκος	Ταμείο	$[\chi_{10}, \chi_{20})$
Έλλη	Ταμείο	$[\chi_1, \chi_8)$
Έλλη	Πωλήσεις	$[\chi_7, \chi_{15})$

Σχήμα 2.4 - Εφαρμογή της πράξης *σύμπτυξη* σε στήλη τύπου *χρονικό διάστημα*.

Εργαζόμενος

Όνομα	Τμήμα	Χρόνος
Νίκος	Ταμείο	χ_1
Νίκος	Ταμείο	χ_2
Νίκος	Ταμείο	χ_5
Έλλη	Ταμείο	χ_2
Έλλη	Ταμείο	χ_3
Έλλη	Πωλήσεις	χ_5
Έλλη	Πωλήσεις	χ_7

Σ = Fold [Χρόνος](Εργαζόμενος)

Name	Dept	Χρόνος
Νίκος	Ταμείο	$[\chi_1, \chi_3)$
Νίκος	Ταμείο	$[\chi_5, \chi_6)$
Έλλη	Ταμείο	$[\chi_2, \chi_4)$
Έλλη	Πωλήσεις	$[\chi_5, \chi_6)$
Έλλη	Πωλήσεις	$[\chi_7, \chi_8)$

Σχήμα 2.5 - Εφαρμογή της πράξης *σύμπτυξη* σε στήλη τύπου *χρονικό σημείο*.

2.2.8. Ανάπτυξη.

Η πράξη της ανάπτυξης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = UNFOLD [στήλη_ανάπτυξης] (πίνακας)

όπου *πίνακας* είναι μία υπάρχουσα σχέση, της οποίας το σχήμα είναι $(\sigma_1, \sigma_2, \ldots, \sigma_\nu)$, και *στήλη_ανάπτυξης* = σ_ι, για κάποιο ι $(1 \le \iota \le \nu)$. Ο τύπος της στήλης *στήλη_ανάπτυξης* θα πρέπει να είναι *χρονικό σημείο* ή *χρονικό διάστημα*. Το σχήμα της σχέσης *αποτέλεσμα*, που προκύπτει ως από την εφαρμογή της πράξης ανάπτυξης θα είναι ίδιο με το σχήμα της σχέσης *πίνακας*, με τη διαφορά ότι ο τύπος της στήλης *Αποτέλεσμα*.σ_ι θα είναι *χρονικό σημείο*, της ίδιας διακριτότητας με τη στήλη *Πίνακας*.σ_ι, ανεξάρτητα από το αν ο τύπος της στήλης *Πίνακας*.σ_ι είναι *χρονικό σημείο* ή *χρονικό διάστημα*.

Όταν η πράξη της ανάπτυξης εφαρμόζεται πάνω σε μία στήλη τύπου *χρονικό διάστημα*, τότε κάθε πλειάδα π, της σχέσης *πίνακας* αντικαθίσταται στη σχέση *αποτέλεσμα* από ένα σύνολο πλειάδων $\{\pi_1, \pi_2, \ldots, \pi_\kappa\}$, οι τιμές των στηλών των οποίων προσδιορίζονται ως ακολούθως:

1. $\pi_\lambda.\sigma_\iota \in \pi.\sigma_\iota$, $\forall\ \lambda = 1, \ldots, \kappa$
2. $\pi_\lambda.\sigma_\mu \in \pi.\sigma_\mu$, $\forall\ \lambda = 1, \ldots, \kappa$ και $\forall\ \mu = 1, \ldots, \nu$ και $\mu \ne \iota$

Τυπικά, για τις σχέσεις *πίνακας* και *αποτέλεσμα* σε μία πράξη ανάπτυξης που εφαρμόζεται πάνω σε στήλη τύπου *χρονικό διάστημα* θα ισχύουν τα ακόλουθα:

1. $\forall\ \pi \in$ *πίνακας*, $\forall\ \chi \in \pi.\sigma_\iota$: $\exists\ \rho \in$ *αποτέλεσμα* με $\pi.\sigma_\kappa = \rho.\sigma_\kappa$ $\forall\ \kappa\ (1 \le \kappa \le \nu$ και $\kappa \ne \iota) \wedge \chi = \rho.\sigma_\iota$
2. για κάθε ζεύγος πλειάδων π και ρ της σχέσης *αποτέλεσμα*, $\exists\ \kappa\ (1 \le \kappa \le \nu)$: $\pi.\sigma_\kappa \ne \rho.\sigma_\kappa$

Η πρώτη συνθήκη εξασφαλίζει ότι κατά την ανάπτυξη δεν χάνεται πληροφορία για κανένα χρονικό σημείο κατά την πράξη της ανάπτυξης, ενώ η δεύτερη συνθήκη εγγυάται ότι η σχέση *αποτέλεσμα* δεν περιέχει ζεύγη ομοίων πλειάδων.

Η πράξη της ανάπτυξης, εφαρμοζόμενη σε στήλη τύπου *χρονικό σημείο* παράγει μία σχέση όμοια με την αρχική (υποθέτουμε ότι η αρχική σχέση δεν περιέχει ζεύγη ομοίων πλειάδων, υπόθεση που ισχύει στα πλαίσια της ΣΑΧΕ).

Η πράξη της ανάπτυξης, μπορεί να εφαρμοστεί και σε περισσότερες από μία στήλες τύπου *χρονικό σημείο* ή *χρονικό διάστημα*. Η εφαρμογή αυτή συμβολίζεται με

αποτέλεσμα = UNFOLD [στήλη$_1$, στήλη$_2$, …, στήλη$_\mu$] (πίνακας)

και είναι ισοδύναμη με την εφαρμογή της πράξης της ανάπτυξης πρώτα στη *στήλη$_1$*, κατόπιν στη *στήλη$_2$* κ.ο.κ. Η σειρά εφαρμογής των πράξεων ανάπτυξης δεν επηρεάζει το αποτέλεσμα.

Στο σχήμα 2.6 παρουσιάζεται ένα παράδειγμα εφαρμογής της πράξης ανάπτυξης σε μία στήλη τύπου *χρονικό διάστημα*.

Εργαζόμενος

Όνομα	Τμήμα	Χρόνος
Νίκος	Ταμείο	$[\chi_1, \chi_3)$
Νίκος	Ταμείο	$[\chi_2, \chi_5)$
Νίκος	Πωλήσεις	$[\chi_4, \chi_5)$
Έλλη	Πωλήσεις	$[\chi_{11}, \chi_{13})$

Σ = UNFOLD [Διάστημα](Εργαζόμενος)

Όνομα	Τμήμα	Χρόνος
Νίκος	Ταμείο	χ_1
Νίκος	Ταμείο	χ_2
Νίκος	Ταμείο	χ_3
Νίκος	Ταμείο	χ_4
Νίκος	Πωλήσεις	χ_4
Νίκος	Πωλήσεις	χ_5
Έλλη	Πωλήσεις	χ_{11}
Έλλη	Πωλήσεις	χ_{12}

Σχήμα 2.6 - Εφαρμογή της πράξης *ανάπτυξη* σε στήλη τύπου *χρονικό διάστημα*.

2.2.9. Κανονικοποίηση.

Η πράξη της κανονικοποίησης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = NORMALISE [στήλη$_1$, στήλη$_2$, …, στήλη$_\mu$](πίνακας)

όπου *πίνακας* είναι μία υπάρχουσα σχέση και στήλη$_1$, στήλη$_2$, …, στήλη$_\mu$ είναι στήλες που ανήκουν στο σχήμα της σχέσης *πίνακας*, κάθε μία από τις οποίες είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*.

Το αποτέλεσμα της πράξης της κανονικοποίησης είναι ίσο με το αποτέλεσμα που προκύπτει από την εφαρμογή της πράξης FOLD [στήλη$_1$,…, στήλη$_\mu$] στο αποτέλεσμα της πράξης UNFOLD [στήλη$_1$,…, στήλη$_\mu$](πίνακας) ή, τυπικά

NORMALISE [στήλη$_1$,…, στήλη$_\mu$] (πίνακας) =

FOLD [στήλη$_1$,…, στήλη$_\mu$] ° UNFOLD [στήλη$_1$,…, στήλη$_\mu$](πίνακας)

Η πράξη της κανονικοποίησης εισάγεται για ευκολία συντακτικού συμβολισμού.

2.2.10. Σημειακή Ένωση.

Η πράξη της σημειακής ένωσης στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = Σ PUNION [στήλη$_1$, …, στήλη$_κ$] Π

όπου Σ και Π είναι δύο συμβατές ως προς την ένωση σχέσεις και στήλη$_1$, …, στήλη$_κ$ είναι στήλες που ανήκουν στο σχήμα της σχέσης Σ, κάθε μία από τις οποίες είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*.

Αν τα σχήματα των σχέσεων Σ είναι ($σ_1$, $σ_2$, …, $σ_ν$) και ($π_1$, $π_2$, …, $π_ν$), αντίστοιχα, και στήλη$_1$ = $σ_{ι_1}$, στήλη$_2$ = $σ_{ι_2}$, …, στήλη$_κ$ = $σ_{ι_κ}$, τότε

Σ PUNION [στήλη$_1$, …, στήλη$_κ$] Π =

NORMALISE [στήλη$_1$, …, στήλη$_κ$] (Σ UNION Π)

Η πράξη PUNION πραγματοποιεί την ένωση των δύο σχέσεων, φροντίζοντας ώστε το τελικό αποτέλεσμα να είναι κανονικοποιημένο στις προσδιορισθείσες στήλες. Στο σχήμα 2.7 παρουσιάζεται ένα παράδειγμα εφαρμογής της πράξης σημειακής ένωσης σε μία στήλη τύπου *χρονικό διάστημα*.

ΠρομηθευτήςΑγαθών

Όνομα	Διάστημα
Υγιεινές Τροφές	[$χ_1$, $χ_5$)
Γενικές Επιχειρήσεις	[$χ_3$, $χ_5$)
Υγιεινές Τροφές	[$χ_4$, $χ_9$)

ΠρομηθευτήςΥπηρεσιών

Όνομα	Διάστημα
Γενικές Επιχειρήσεις	[$χ_1$, $χ_3$)
Διαφημιστική Α.Ε.	[$χ_1$, $χ_4$)
Γενικές Επιχειρήσεις	[$χ_7$, $χ_9$)

Προμηθευτής = ΠρομηθευτήςΑγαθών PUNION[Διάστημα] ΠρομηθευτήςΥπηρεσιών

Όνομα	Διάστημα
Υγιεινές Τροφές	[$χ_1$, $χ_9$)
Γενικές Επιχειρήσεις	[$χ_1$, $χ_5$)
Διαφημιστική Α.Ε.	[$χ_1$, $χ_4$)
Γενικές Επιχειρήσεις	[$χ_7$, $χ_9$)

Σχήμα 2.7 - Εφαρμογή της πράξης *σημειακή ένωση* σε μία στήλη τύπου *χρονικό διάστημα*.

2.2.11. Σημειακή Διαφορά.

Η πράξη της σημειακής διαφοράς στη ΣΑΧΕ συμβολίζεται με

αποτέλεσμα = Σ PEXCEPT [στήλη$_1$, …, στήλη$_κ$] Π

όπου Σ και Π είναι δύο συμβατές ως προς την ένωση σχέσεις και στήλη$_1$, …, στήλη$_κ$ είναι στήλες που ανήκουν στο σχήμα της σχέσης Σ, κάθε από τις οποίες είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*.

Αν τα σχήματα των σχέσεων Σ είναι (σ_1, σ_2, …, σ_v) και (π_1, π_2, …, π_v), αντίστοιχα, και στήλη$_1$ = σ_{i_1}, στήλη$_2$ = σ_{i_2}, …, στήλη$_\kappa$ = σ_{i_κ}, τότε

Σ PEXCEPT [στήλη$_1$, …, στήλη$_K$] Π =

FOLD [στήλη$_1$, …, στήλη$_K$] (UNFOLD [σ_{i_1}, …, σ_{i_κ}] (Σ) EXCEPT

UNFOLD [π_{i_1}, …, π_{i_κ}] (Π))

Η πράξη PEXCEPT υπολογίζει τη διαφορά των δύο σχέσων, αφού αναλύσει την περιεχόμενη σ' αυτές πληροφορία σε επίπεδο χρονικών σημείων και φροντίζει ώστε το τελικό αποτέλεσμα να είναι κανονικοποιημένο στις προσδιορισθείσες στήλες.

Στο σχήμα 2.8 φαίνεται ένα παράδειγμα εφαρμογής της πράξης PEXCEPT σε μία στήλη τύπου *χρονικό διάστημα*.

ΕργαζόμενοςΤαμείου

Όνομα	Διάστημα
Γιάννης	[χ_1, χ_7)
Μαρία	[χ_3, χ_5)
Γιάννης	[χ_8, χ_{10})
Ιωάννα	[χ_6, χ_9)

ΕργαζόμενοςΠωλήσεων

Όνομα	Διάστημα
Γιάννης	[χ_3, χ_5)
Μαρία	[χ_2, χ_8)
Νίκος	[χ_5, χ_8)
Ιωάννα	[χ_8, χ_9)

ΕργαζόμενοςΜόνοΣτοΤαμείο = ΕργαζόμενοςΤαμείου PEXCEPT [Διάστημα] ΕργαζόμενοςΠωλήσεων

Όνομα	Διάστημα
Γιάννης	[χ_1, χ_3)
Γιάννης	[χ_5, χ_7)
Γιάννης	[χ_8, χ_{10})
Ιωάννα	[χ_6, χ_8)

Σχήμα 2.8 - Παράδειγμα εφαρμογής της πράξης *σημειακή διαφορά* σε μία στήλη τύπου *χρονικό διάστημα*.

2.3. Σύνοψη.

Στο κεφάλαιο αυτό παρουσιάσαμε τη Σχεσιακή Άλγεβρα Χρόνου Εγκυρότητας. Στα πλαίσια της ΣΑΧΕ ορίζονται δύο νέοι τύποι δεδομένων, το *χρονικό σημείο* και το *χρονικό διάστημα*, κατηγορήματα και συναρτήσεις πάνω στους τύπους αυτούς, καθώς και νέες πράξεις που εφαρμόζονται πάνω σε σχέσεις που περιέχουν στήλες, των οποίων το πεδίο ορισμού είναι ένας από τους δύο νέους τύπους. Στο επόμενο κεφάλαιο θα παρουσιαστεί μία χρονολογική επέκταση της SQL, η SQL χρόνου εγκυρότητας (SQL-XE (VT-SQL) [ESPRIT93β]) που βασίζεται στη ΣΑΧΕ.

3. Συντακτικό και σημασιολογία της SQL-XE.

Η SQL-XE ([ESPRIT93β]) είναι μία δηλωτική γλώσσα ερωτήσεων (declarative query language), της οποίας το συντακτικό είναι βασισμένο στο πρότυπο SQL89 του οργανισμού ANSI ([Lans88]), ενώ η σημασιολογία της στηρίζεται πάνω στις πράξεις της ΣΑΧΕ, την οποία παρουσιάσαμε στο προηγούμενο κεφάλαιο. Η SQL-XE επεκτείνει το συντακτικό και τη σημασιολογία της SQL89, επιτρέποντας τη δημιουργία και διαχείριση πινάκων που περιέχουν δεδομένα με καθορισμένο χρόνο εγκυρότητας (valid time data). Η επέκταση γίνεται κατά συνεπή τρόπο, τόσο από συντακτικής όσο και από σημασιολογικής άποψης, με την έννοια ότι:

1. όλες οι εντολές που είναι έγκυρες στην SQL89 είναι έγκυρες και στην SQL-XE.
2. το συντακτικό των επεκτάσεων είναι παρόμοιο με το συντακτικό των ήδη υπαρχόντων δομών (π.χ. για τα κατηγορήματα χρησιμοποιείται η ενδοθεματική (infix) μορφή, για τις συναρτήσεις χρησιμοποιείται το συντακτικό *όνομα_συνάρτησης(όρισμα, ...)* κ.λπ.).
3. η σημασιολογία των εντολών της SQL-XE είναι παρόμοια με αυτή των εντολών της SQL89, με πιθανή εφαρμογή κάποιων τελεστών της ΣΑΧΕ, είτε επιπροσθέτως, είτε σε αντικατάσταση τελεστών της σχεσιακής άλγεβρας.

Στις επόμενες παραγράφους παρουσιάζεται το συντακτικό και η σημασιολογία της SQL-XE. Για την παρουσίαση του συντακτικού της γλώσσας υιοθετούνται οι ακόλουθοι συμβολισμοί:

- τα τμήματα που περικλείονται σε αγκύλες (`[ ]`) είναι προαιρετικά.
- τα τμήματα που περικλείονται σε άγκιστρα (`{ }`) μπορούν να παραλειφθούν ή να επαναληφθούν μία ή περισσότερες φορές.
- ο συμβολισμός $\alpha \mid \beta$ διαβάζεται "είτε το α, είτε το β" και δηλώνει ότι ακριβώς ένας από τους όρους α και β μπορεί να παρατίθεται στο συγκεκριμένο σημείο.
- οι παρενθέσεις χρησιμοποιούνται για ομαδοποίηση. Όταν οι παρενθέσεις πρέπει να παρατίθενται αυτούσιες, θα εσωκλείονται σε εισαγωγικά (`'('` και `')'`).
- οι λέξεις που γράφονται με κεφαλαία είναι δεσμευμένες.
- οι λέξεις που γράφονται με πλάγιους χαρακτήρες υποδηλώνουν μεταβλητές που είτε αντικαθίστανται από κάποιον συντακτικό όρο της SQL-XE, είτε δίνονται από τον χρήστη.
- το πέρας του ορισμού κάποιου συντακτικού όρου σημειώνεται με το χαρακτήρα `;`.

3.1. Η Γλώσσα Ορισμού Δεδομένων της SQL-XE.

Η γλώσσα ορισμού δεδομένων της SQL-XE αποτελείται από τις εντολές *CREATE TABLE, DROP TABLE, CREATE VIEW* και *DROP VIEW*, οι οποίες επιτρέπουν τον ορισμό τόσο του *ιδεατού σχήματος* (conceptual schema) της βάσης δεδομένων, που αποτελείται από τα σχήματα των πινάκων, όσο και του *εξωτερικού σχήματος* (external schema), που αποτελείται από τους ορισμούς των πινάκων και των *όψεων* (views) ([Date82]). Στις επόμενες παραγράφους περιγράφονται οι εντολές της γλώσσας ορισμού δεδομένων της SQL-XE. Μερικές από τις εντολές δεν τροποποιούνται, σε σχέση με το πρότυπο SQL89, αναφέρονται όμως για λόγους πληρότητας.

3.1.1. Δημιουργία Πινάκων: Η Εντολή CREATE TABLE.

Η εντολή *CREATE TABLE* χρησιμοποιείται για να δημιουργηθούν νέοι πίνακες στη βάση δεδομένων. Η SQL-XE επεκτείνει το συντακτικό της εντολής *CREATE TABLE* της SQL89, προσθέτοντας προτάσεις για τον προσδιορισμό των στηλών που μετέχουν στο *πρωτεύον κλειδί* του πίνακα, καθώς και των στηλών που περιέχουν το χρόνο εγκυρότητας των πλειάδων. Η σύνταξη της εντολής *CREATE TABLE* στην SQL-XE είναι η ακόλουθη:

CREATE TABLE *όνομα_πίνακα* '(' *στήλες_πίνακα*
[NORMALISED '(' *στήλες_κανονικοποίησης* ')']
[PRIMARY KEY '(' *στήλες_κλειδιά* ')'] ')' ;

όπου

στήλες_πίνακα ::= *στήλη τύπος* [NOT NULL]
{, *στήλη τύπος* [NOT NULL]} ;

τύπος ::= *τύπος_SQL89* | INTERVAL(*διακριτότητα*) | POINT(*διακριτότητα*) ;

στήλες_κανονικοποίησης ::= *στήλη* {, *στήλη*} ;

στήλες_κλειδιά ::= *στήλη* [POINT | INTERVAL]
{, *στήλη* [POINT | INTERVAL] } ;

Η εκτέλεση της εντολής *CREATE TABLE* έχει ως αποτέλεσμα να δημιουργηθεί ο πίνακας *όνομα πίνακα*, με τις στήλες που ορίζονται στον όρο *στήλες πίνακα*. Ο τύπος της κάθε στήλης μπορεί να είναι οποιοσδήποτε από τους τύπους που ορίζονται στην SQL89 ή ένας από τους τύπους *χρονικό διάστημα* και *χρονικό σημείο* με καθορισμένη διακριτότητα. Ο τύπος *χρονικό διάστημα* προσδιορίζεται ως *INTERVAL(διακριτότητα)*, ενώ ο τύπος *χρονικό σημείο* προσδιορίζεται ως *POINT(διακριτότητα)*. Ο όρος *διακριτότητα* είναι μία συμβολοσειρά που καθορίζει τη διακριτότητα του χρονικού διαστήματος (ή του χρονικού σημείου). Οι δυνατές

τιμές της συμβολοσειράς, καθώς και οι αντίστοιχες διακριτότητες παρουσιάζονται στο σχήμα 3.1.

Συμβολοσειρά	Διακριτότητα
'year'	έτος
'month'	μήνας
'day'	ημέρα
'hour'	ώρα
'minute'	λεπτό
'sec'	δευτερόλεπτο
'sec(N)'	10^{-N} δευτερόλεπτα

Σχήμα 3.1 - Οι διακριτότητες χρονικών διαστημάτων και σημείων στην SQL-XE.

Αν η πρόταση *NORMALISED* είναι παρούσα, τότε κάθε στήλη που αναφέρεται στον όρο *στήλες κανονικοποίησης* θα πρέπει να δηλώνεται στον όρο *στήλες πίνακα* ως τύπου *χρονικό διάστημα*. Η πρόταση *NORMALISED* ορίζει ότι οι πράξεις της εισαγωγής, διαγραφής και ενημέρωσης στον πίνακα που δημιουργείται, θα ακολουθούνται από μία πράξη *κανονικοποίησης* στις στήλες που αναφέρονται στον όρο *στήλες κανονικοποίησης*. (Στο επίπεδο της υλοποίησης μπορούν να εφαρμόζονται διαφορετικοί αλγόριθμοι για την υλοποίηση των πράξεων, η τελική κατάσταση της βάσης δεδομένων να είναι ίδια με την κατάσταση που θα προέκυπτε αν εφαρμοζόταν η πράξη της κανονικοποίησης.) Αν η εντολή *CREATE TABLE* που χρησιμοποιήθηκε για τη δημιουργία ενός πίνακα περιείχε την πρόταση *NORMALISED*, ο πίνακας αυτός καλείται *κανονικοποιημένος* και οι στήλες που αναφέρονται στην πρόταση *NORMALISED* καλούνται *στήλες κανονικοποίησης* του πίνακα. Τέλος, οι στήλες που αναφέρονται στον όρο *κανονικοποιημένες στήλες* αποκτούν την ιδιότητα *NOT NULL*, ανεξάρτητα αν προσδιορίζεται ή όχι στη δήλωση της στήλης στον όρο *στήλες πίνακα*.
Η πρόταση *PRIMARY KEY*, αν είναι παρούσα, δηλώνει τις στήλες που αποτελούν το *πρωτεύον κλειδί* της σχέσης. Οι στήλες που αναφέρονται στον όρο *στήλες κλειδιά* πρέπει να έχουν δηλωθεί στον όρο *στήλες πίνακα* και αποκτούν αυτόματα την ιδιότητα *NOT NULL*, έτσι ώστε να πληρείται ο κανόνας ακεραιότητας οντοτήτων (entity integrity rule - [Codd85]) του σχεσιακού μοντέλου. Κάθε στήλη που αναφέρεται στον όρο *στήλες κλειδιά* μπορεί να ακολουθείται από μία από τις δεσμευμένες λέξεις *INTERVAL* και *POINT*, προσδιορίζοντας έτσι τον τρόπο με τον οποίο η συγκεκριμένη στήλη συμμετέχει στο κλειδί. (Αν δεν παρατίθεται καμία από τις δεσμευμένες λέξεις *INTERVAL* και *POINT*, τότε η εξ ορισμού

συμπεριφορά είναι αυτή που καθορίζεται από τη δεσμευμένη λέξη *INTERVAL*). Αν μία στήλη ακολουθείται από τη δεσμευμένη λέξη *POINT* στον όρο *στήλες κλειδιά*, τότε καλείται *κλειδί τύπου σημείο* (point-type key), ενώ στην αντίθετη περίπτωση καλείται *κλειδί τύπου διάστημα* (interval-type key).

Αν η πρόταση *NORMALISED* είναι παρούσα, τότε όλες οι στήλες που αναφέρονται στον όρο *στήλες κανονικοποίησης* πρέπει να είναι παρούσες και στον όρο *στήλες κλειδιά*, ακολουθούμενες από τη δεσμευμένη λέξη *POINT*. Η δεσμευμένη λέξη *POINT* μπορεί να χρησιμοποιηθεί μόνο για τις στήλες που αναφέρονται στον όρο *στήλες κανονικοποίησης*.

Αν το πρωτεύον κλειδί ενός πίνακα Σ αποτελείται από τις στήλες σ_1, ..., σ_κ, που ακολουθούνται από τη δεσμευμένη λέξη *INTERVAL* και τις στήλες $\sigma_{\kappa+1}$, ..., σ_ν, που ακολουθούνται από τη δεσμευμένη λέξη *POINT*, τότε ισχύουν τα ακόλουθα:

1) ο πίνακας Σ δεν μπορεί να περιέχει ένα ζεύγος πλειάδων π_1 και π_2 για το οποίο συναληθεύουν οι ακόλουθες συνθήκες:
   i) $\pi_1.\sigma_\iota = \pi_2.\sigma_\iota$, $\forall\ \iota = 1, \ldots, \kappa$
   ii) $\pi_1.\sigma_\iota\ cp\ \pi_2.\sigma_\iota$, $\forall\ \iota = \kappa + 1, \ldots, \nu$
2) η εισαγωγή μιας πλειάδας π_1 στον πίνακα Σ προϋποθέτει ότι ο πίνακας δεν περιέχει ήδη κάποια πλειάδα π_2, για την οποία να συναληθεύουν οι συνθήκες (i) και (ii) της περίπτωσης (1). (Η μαζική εισαγωγή ενός συνόλου πλειάδων σε έναν πίνακα αντιμετωπίζεται ως εισαγωγή της πρώτης πλειάδας του συνόλου στον πίνακα, ακολουθούμενη από την εισαγωγή των υπολοίπων πλειάδων.)

Η SQL-XE υποστηρίζει δύο διαφορετικούς τύπους κλειδιών προκειμένου να παρέχεται η δυνατότητα αναπαράστασης τόσο των περιπτώσεων όπου τα χρονικά διαστήματα μπορούν να αναλυθούν σε χρονικά σημεία (κλειδιά τύπου σημείο), καθώς και των περιπτώσεων όπου τα χρονικά διαστήματα πρέπει να αντιμετωπίζονται ως αδιαίρετες χρονικές ολότητες (κλειδιά τύπου διάστημα). Ένα παράδειγμα όπου κάποιο χρονικό διάστημα μπορεί να αναλυθεί σε χρονικά σημεία είναι το ιστορικό της εξέλιξης ενός καθηγητή σε κάποιο ακαδημαϊκό ίδρυμα: αν το σχήμα του πίνακα *Βαθμίδα*, στον οποίο καταγράφεται η εξέλιξη αυτή είναι *(Ονομα, Βαθμίδα, Διάστημα)*, τότε η πλειάδα *(Πλάτων, Γ, [1991-02-05, 1993-10-02))* σημαίνει ότι ο καθηγητής Πλάτων ανήκε στην τρίτη βαθμίδα καθ' όλη την περίοδο από την 5η Φεβρουαρίου του 1991 έως και την 1η Οκτωβρίου 1993. Το γεγονός ότι ο καθηγητής Πλάτων ανήκει στην τρίτη βαθμίδα, είναι επίσης αληθές για όλα τα υποδιαστήματα της περιόδου [1991-02-05, 1993-10-02) καθώς επίσης και για όλα τα χρονικά σημεία που ανήκουν σ' αυτό το διάστημα.

Είναι επίσης σαφές ότι η προσπάθεια εισαγωγής μιας πλειάδας *(Πλάτων, Β, [1992-08-21, 1994-07-12))* πρέπει να αποτύχει, γιατί στην αντίθετη περίπτωση στη βάση δεδομένων θα βρίσκονται καταχωρημένες αντιφατικές πληροφορίες για τη βαθμίδα του καθηγητή Πλάτων κατά το διάστημα [1992-08-21, 1993-10-02). Δεδομένου ότι η ιδιότητα ενός καθηγητή να ανήκει σε κάποια βαθμίδα έχει νόημα σε επίπεδο χρονικού σημείου, ο πίνακας *Βαθμίδα* θα πρέπει να έχει δημιουργηθεί με την εντολή

```
CREATE TABLE Βαθμίδα (Όνομα CHAR(30) NOT NULL,
                      Βαθμίδα CHAR,
                      Διάστημα INTERVAL('day') NOT NULL
                NORMALISED (Διάστημα)
                PRIMARY KEY (Όνομα INTERVAL, Διάστημα POINT))
```

(Ο τύπος *INTERVAL('day')* ορίζει χρονικά διαστήματα με διακριτότητα ημέρας.)

Αντίθετα με την περίπτωση του ιστορικού της εξέλιξης ενός καθηγητή, στο ιστορικό του μέσου όρου βαθμολογίας ενός φοιτητή κατά τα έτη σπουδών του, το χρονικό διάστημα δεν μπορεί να αναλυθεί σε υποδιαστήματα ή χρονικά σημεία. Αν το σχήμα του πίνακα *ΜέσοςΌρος*, στον οποίο καταγράφεται ο μέσος όρος της βαθμολογίας του φοιτητή είναι *(ΑριθΜητρώου, ΜέσοςΌρος, Διάστημα)*, τότε μία πλειάδα *(732, 8.0, [1992-10-01, 1993-10-01))* σημαίνει ότι ο μέσος όρος του φοιτητή με αριθμό μητρώου 732 για το ακαδημαϊκό έτος 1992 ήταν ακριβώς 8. Καμία όμως πληροφορία δεν μπορεί να συναχθεί για τα υποδιαστήματα του ακαδημαϊκού έτους: είναι πιθανό ο συγκεκριμένος φοιτητής να εξετάστηκε επιτυχώς στη διάρκεια του ακαδημαϊκού έτους σε 10 μαθήματα με βαθμό 8, οπότε ο μέσος όρος του να ήταν 8 σε κάθε χρονικό σημείο του διαστήματος, είναι όμως πιθανό να εξετάστηκε επιτυχώς σε πέντε μαθήματα με βαθμό 7 στο χειμερινό εξάμηνο και σε άλλα πέντε στο εαρινό εξάμηνο με βαθμό 9, οπότε ο τελικός μέσος όρος του για το ακαδημαϊκό έτος να είναι 8, αλλά ο μέσος όρος του χειμερινού εξαμήνου είναι 7 και του εαρινού 9. Επιπροσθέτως, σύμφωνα με τα παραπάνω, οι πλειάδες *(732, 8.0, [1992-10-01, 1993-10-01))*, *(732, 7.0, [1992-10-01, 1993-02-15))* και *(732, 9.0, [1993-02-15, 1993-10-01))* θα πρέπει να μπορούν να συνυπάρχουν στη βάση δεδομένων, μια και όλες περιέχουν έγκυρη πληροφορία. Έτσι, δεδομένου ότι ο μέσος όρος ενός φοιτητή έχει νόημα σε επίπεδο χρονικής περιόδου, ο πίνακας *ΜέσοςΌρος* θα πρέπει να δημιουργηθεί με την εντολή

CREATE TABLE ΜέσοςΌρος (ΑριθΜητρώου CHAR(30) NOT NULL,
ΜέσοςΌρος REAL,
Διάστημα INTERVAL('day') NOT NULL
PRIMARY KEY (ΑριθΜητρώου INTERVAL, Διάστημα INTERVAL))

Η παρουσία της πρότασης *PRIMARY KEY*, στην οποία προσδιορίζεται ότι οι στήλες *ΑριθΜητρώου* και *Διάστημα* μετέχουν στο πρωτεύον κλειδί ως κλειδιά τύπου διάστημα, εγγυάται ότι ο πίνακας *ΜέσοςΌρος* δεν θα περιέχει κάποιο ζεύγος πλειάδων με ίσες τιμές στις στήλες *ΑριθΜητρώου* και *Διάστημα*.

3.1.2. Διαγραφή Πινάκων: Η Εντολή DROP TABLE.

Τόσο το συντακτικό όσο και η σημασιολογία της εντολής *DROP TABLE* δεν διαφοροποιούνται στην SQL-XE, σε σχέση με την SQL89. Η σύνταξη της εντολής DROP TABLE στην SQL-XE είναι

DROP TABLE *όνομα_πίνακα* ;

και η εκτέλεσή της καταστρέφει τον προσδιοριζόμενο πίνακα.

3.1.3. Δημιουργία Όψεων: Η Εντολή CREATE VIEW.

Η σύνταξη της εντολής *CREATE VIEW* στην SQL-XE είναι

CREATE VIEW *όνομα_όψης* AS *ερώτηση* ;

όπου *όνομα όψης* είναι το όνομα της όψης που επιθυμούμε να δημιουργήσουμε και *ερώτηση* είναι μία εντολή *SELECT* της SQL-XE (το συντακτικό και η σημασιολογία της εντολής *SELECT* στην SQL-XE περιγράφονται στην παράγραφο 3.2.1). Το αποτέλεσμα της εκτέλεσης αυτής της εντολής είναι η αποθήκευση του *ορισμού* της όψης στη βάση δεδομένων. Όταν μια όψη χρησιμοποιείται σε μία εντολή, τότε ο ορισμός της όψης ανακαλείται από τη βάση δεδομένων και συνδυάζεται με την εντολή, καταλήγοντας σε μία *τροποποιημένη εντολή*, η οποία τελικά εκτελείται.

Η εισαγωγή, διαγραφή και ενημέρωση δεδομένων μέσω όψεων δεν είναι πάντα δυνατή, διότι μπορεί να οδηγήσει σε παραβίαση των κανόνων ακεραιότητας του σχεσιακού μοντέλου ([Date82], [Korth86]). Ειδικότερα σε ένα χρονολογικό σύστημα διαχείρισης βάσεων δεδομένων, η τροποποίηση όψεων χρήζει ιδιαίτερης ανάλυσης, προκειμένου να αντιμετωπισθούν οι περιπτώσεις των κλειδιών τύπου σημείο, καθώς και οι μετατροπές μεταξύ των δύο δυνατών αναπαραστάσεων του χρόνου, των *χρονικών σημείων* και των *χρονικών διαστημάτων*. Η αντιμετώπιση των ζητημάτων αυτών είναι έξω από τους σκοπούς της παρούσας έρευνας και έτσι οι όψεις δεν θα μας απασχολήσουν στη συνέχεια.

3.1.4. Διαγραφή Όψεων: Η Εντολή DROP VIEW.

Το συντακτικό και η σημασιολογία της εντολής *DROP VIEW* δεν διαφοροποιείται στην SQL-XE, σε σχέση με την SQL89. Η σύνταξη της εντολής DROP VIEW στην SQL-XE είναι

DROP VIEW *όνομα_όψης* ;

και η εκτέλεση της εντολής αυτής διαγράφει τον ορισμό της όψης που προσδιορίζεται από τη βάση δεδομένων.

3.2. Η Γλώσσα Διαχείρισης Δεδομένων της SQL-XE.

Η γλώσσα διαχείρισης δεδομένων της SQL-XE αποτελείται από τις εντολές *SELECT, INSERT, DELETE* και *UPDATE*, οι οποίες έχουν επεκταθεί τόσο συντακτικά όσο και σημασιολογικά, σε σχέση με τις αντίστοιχες εντολές της SQL89, προκειμένου να παρέχουν τη δυνατότητα χειρισμού χρονικών δεδομένων. Στις επόμενες παραγράφους παρουσιάζονται το συντακτικό και η σημασιολογία της γλώσσα διαχείρισης δεδομένων της SQL-XE.

3.2.1. Ανάκτηση δεδομένων: Η εντολή SELECT.

Η σύνταξη της εντολής *SELECT* στην SQL-XE είναι

ερώτηση_SQL_XE ::= *επεκταμένη_ερώτηση*

{ (UNION | UNION ALL | EXCEPT) [*στήλες_αποτελέσματος*]
επεκταμένη_ερώτηση}
[ORDER BY *στήλες_ταξινόμησης*] ;

όπου

επεκταμένη_ερώτηση ::= *απλή_ερώτηση_SQL89*

[REFORMAT AS
(FOLD | UNFOLD | UNFOLD ALL) *στήλες_αποτελέσματος*
{(FOLD | UNFOLD | UNFOLD ALL) *στήλες_αποτελέσματος*}]
[NORMALISE ON *στήλες_αποτελέσματος*] ;

απλή_ερώτηση_SQL89 ::= SELECT [ALL | DISTINCT] *λίστα_εκφράσεων*

FROM *λίστα_σχέσεων_και_ψευδωνύμων*
[WHERE *επεκταμένη_συνθήκη*]
[GROUP BY *λίστα_στηλών*]
[HAVING *επεκταμένη_συνθήκη*] ;

επεκταμένη_συνθήκη ::= *συνθήκη_SQL_XE*
| *συνθήκη_SQL_XE* AND *επεκταμένη_συνθήκη*
| *συνθήκη_SQL_XE* OR *επεκταμένη_συνθήκη*
| NOT '(' *επεκταμένη_συνθήκη* ')'
| '(' *επεκταμένη_συνθήκη* ')'
;

συνθήκη_SQL_XE ::= *έκφραση τελεστής_SQL_XE έκφραση*
| *έκφραση* [NOT] *τελεστής_SQL_XE* [ALL | ANY]
'(' *επεκταμένη_ερώτηση* ')'
| [NOT] EXISTS '(' *επεκταμένη_ερώτηση* ')'
;

τελεστής_SQL_XE ::= *τελεστής_υποερώτησης_SQL89*
| *κατηγόρημα_σύγκρισης_χρονικών_σημείων*
| *κατηγόρημα_χρονικών_διαστημάτων*
;

στήλες_αποτελέσματος ::= *στήλη_αποτελέσματος*
{, *στήλη_αποτελέσματος*} ;

στήλη_αποτελέσματος ::= *στήλη* | *πίνακας.στήλη* | *θετικός_ακέραιος* ;

στήλες_ταξινόμησης ::= *στήλη_αποτελέσματος* [ASC | DESC]
{, *στήλη_αποτελέσματος* [ASC | DESC]} ;

Η *λίστα εκφράσεων* είναι μία ακολουθία από εκφράσεις, διαχωρισμένες με κόμματα. Κάθε έκφραση μπορεί να είναι μία παράσταση που περιέχει σταθερές (συμπεριλαμβανομένων σταθερών τύπου *χρονικό σημείο* και *χρονικό διάστημα*), ονόματα στηλών, καθώς και συναρτήσεις, συνδυασμένα με τελεστές. Τα ονόματα των στηλών μπορεί να είναι *χαρακτηρισμένα* (qualified), δηλαδή να περιέχουν και το όνομα της σχέσης στην οποία ανήκουν ή *χωρίς χαρακτηρισμό* (unqualified), αρκεί η έλλειψη χαρακτηρισμού να μην οδηγεί σε ασάφεια, ή, με άλλα λόγια, το συγκεκριμένο όνομα στήλης να απαντάται στο σχήμα μίας μόνο από τις σχέσεις που προσδιορίζονται στη *λίστα σχέσεων και ψευδωνύμων*.

Σε μία παράσταση μπορούν να χρησιμοποιηθούν όλες οι συναρτήσεις που ορίζονται από την SQL, οι συναρτήσεις που ορίζονται από τη ΣΑΧΕ (περιγράφονται στην παράγραφο 2.2.6), καθώς και οι κάτωθι συναρτήσεις:

1. **countap(δ_1)**

Η συνάρτηση *countap* είναι μία συναθροιστική συνάρτηση, δηλαδή εφαρμόζεται σε *ομάδες* πλειάδων και όχι σε κάθε πλειάδα ξεχωριστά. Κάθε ομάδα πλειάδων αποτελείται από όλες τις πλειάδες του αποτελέσματος, των οποίων όλες οι στήλες που αναφέρονται στην πρόταση *GROUP BY* έχουν ίσες τιμές (αν η πρόταση *GROUP BY* δεν είναι παρούσα, τότε όλες οι πλειάδες του αποτελέσματος ανήκουν στην ίδια ομάδα). Για κάθε ομάδα, υπολογίζεται η τιμή της έκφρασης δ_1 (η οποία πρέπει να δίνει αποτέλεσμα τύπου *διαστήματος*) για όλες τις πλειάδες της και η συνάρτηση επιστρέφει το πλήθος των χρονικών σημείων που περιέχονται σε όλα τα διαστήματα. Αν ένα χρονικό σημείο περιλαμβάνεται σε περισσότερα από ένα διαστήματα, το αποτέλεσμα της συνάρτησης *countap* αυξάνεται κατά 1 για κάθε εμφάνιση του χρονικού σημείου.

2. **countdp(δ_1)**

Η συνάρτηση *countdp* είναι παρόμοια με τη συνάρτηση *countap*, με τη διαφορά ότι κάθε χρονικό σημείο συνεισφέρει κατά 1 στο αποτέλεσμα της συνάρτησης, ανεξάρτητα από τον αριθμό των διαστημάτων στα οποία συμπεριλαμβάνεται, στα πλαίσια μιας ομάδας.

3. **maxpoint(σ_1)**

Η συνάρτηση *maxpoint* επιστρέφει το μέγιστο χρονικό σημείο που υποστηρίζεται από το σύστημα. Το όρισμα σ_1 είναι μία συμβολοσειρά που καθορίζει την επιθυμητή διακριτότητα του αποτελέσματος, όπως περιγράφεται στον πίνακα 3.1.

Το μέγιστο υποστηριζόμενο χρονικό σημείο μπορεί να εξαρτάται από τη ζητούμενη διακριτότητα: π.χ. για χρονικά σημεία διακριτότητας έτους μπορεί να υποστηρίζεται ένα εύρος τιμών από το 15.000 π.Χ. έως το 9.999 μ.Χ., ενώ για χρονικά σημεία διακριτότητας μικροδευτερολέπτου, το εύρος τιμών μπορεί να περιορίζεται στην περιοχή 1970-01-01 00:00:00.0 έως 1999-12-31 23:59:59.999999.

4. **minpoint(σ_1)**

Η συνάρτηση *minpoint* επιστρέφει το ελάχιστο χρονικό σημείο που υποστηρίζεται από το σύστημα. Για το όρισμα σ_1 ισχύουν οι παρατηρήσεις που διατυπώνονται για τη συνάρτηση *maxpoint*.

5. **now(σ_1)**

Η συνάρτηση *now* επιστρέφει το τρέχον χρονικό σημείο. Για το όρισμα σ_1 ισχύουν οι παρατηρήσεις που διατυπώνονται για τη συνάρτηση *maxpoint*.

Όπως φαίνεται από τη συντακτική περιγραφή της, η *επεκταμένη συνθήκη* της SQL-XE που χρησιμοποιείται στις προτάσεις *WHERE* και *HAVING*, έχει τις ακόλουθες επεκτάσεις, σε σχέση με τη *συνθήκη* που ορίζεται στο πρότυπο SQL89:

1. στις εκφράσεις που μετέχουν στην *επεκταμένη συνθήκη* μπορούν να χρησιμοποιούνται σταθερές τύπου *χρονικό σημείο* και *χρονικό διάστημα* καθώς και οι συναρτήσεις της SQL-XE.
2. τα κατηγορήματα σύγκρισης χρονικών σημείων, καθώς και τα κατηγορήματα χρονικών διαστημάτων της ΣΑΧΕ έχουν ισότιμη μεταχείριση με τους τελεστές σύγκρισης της SQL89 και μπορούν να χρησιμοποιηθούν στο σχηματισμό συνθηκών καθώς και στη διαμόρφωση υποερωτήσεων.
3. στις υποερωτήσεις μπορούν να χρησιμοποιούνται οι προτάσεις *REFORMAT AS* και *NORMALISE ON* της SQL-XE.

Για την αποτίμηση μιας *επεκταμένης ερώτησης*, εφαρμόζεται ο πιο κάτω αλγόριθμος:

1. αρχικά αποτιμάται το τμήμα *απλή ερώτηση SQL*, σύμφωνα με τους ορισμούς του προτύπου SQL89 και το αποτέλεσμα αποθηκεύεται σε έναν προσωρινό πίνακα.
2. αν η πρόταση *REFORMAT AS* είναι παρούσα, τότε πρέπει όλες οι στήλες που αναφέρονται στον όρο *στήλες αποτελέσματος* της κάθε υποπρότασης *FOLD*, *UNFOLD* και *UNFOLD ALL* να είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*. Οι υποπροτάσεις *FOLD*, *UNFOLD* και *UNFOLD ALL* εκτελούνται διαδοχικά, κατά σειρά εμφάνισης στην πρόταση *REFORMAT AS*. Κάθε προσδιορισμός στήλης στον όρο *στήλες αποτελέσματος* μπορεί να είναι είτε το όνομά της στήλης (χαρακτηρισμένο ή μη), είτε ο αύξων αριθμός μιας στήλης στη *λίστα εκφράσεων*. Η εκτέλεση μιας υποπρότασης συνίσταται στην εφαρμογή της αντίστοιχης πράξης της ΣΑΧΕ, πάνω στις *στήλες αποτελέσματος* (που αναφέρονται στη συγκεκριμένη υποπρόταση) του τελευταίου προσωρινού πίνακα που έχει προκύψει και η δημιουργία ενός νέου προσωρινού πίνακα. Ειδικότερα η πράξη *UNFOLD ALL*, είναι ισοδύναμη με την αλγεβρική πράξη *UNFOLD*, με τη διαφορά ότι επιτρέπεται να υπάρχουν στο αποτέλεσμα ζεύγη πλειάδων των οποίων όλες οι στήλες είναι ίσες.
3. αν η πρόταση *NORMALISE ON* είναι παρούσα, τότε πρέπει όλες οι στήλες που προσδιορίζονται στον όρο *στήλες αποτελέσματος* της πρότασης αυτής να είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*. Όπως και στις υποπροτάσεις της πρότασης *REFORMAT AS*, κάθε στήλη μπορεί να προσδιορίζεται στον όρο *στήλες αποτελέσματος* είτε με το όνομά της, είτε με τον αύξοντα αριθμό της στη *λίστα εκφράσεων*. Η παρουσία

της πρότασης *NORMALISE ON* προκαλεί την εκτέλεση μιας πράξης κανονικοποίησης πάνω στις στήλες που αναφέρονται στον όρο *στήλες αποτελέσματος* του τελευταίου προσωρινού πίνακα που έχει προκύψει και τη δημιουργία ενός νέου προσωρινού πίνακα.

Ο τελευταίος προσωρινός πίνακας που έχει προκύψει ως αποτέλεσμα της πιο πάνω διαδικασίας περιέχει το αποτέλεσμα της *επεκταμένης ερώτησης*.

Όπως προκύπτει από το συντακτικό της, μία ερώτηση SQL-XE αποτελείται από μία *επεκταμένη ερώτηση*, η οποία μπορεί να ακολουθείται (προαιρετικά) από μία από ή περισσότερες επαναλήψεις μιας συντακτικής δομής που περιλαμβάνει:

1. μία από τις δεσμευμένες λέξεις *UNION, UNION ALL* και *EXCEPT*.
2. μία *επεκταμένη ερώτηση*.
3. προαιρετικά, ανάμεσα στη δεσμευμένη λέξη και την *επεκταμένη ερώτηση* μπορεί να παρεμβάλλεται ο όρος *στήλες αποτελέσματος*, δηλαδή μία ακολουθία από αναφορές σε στήλες αποτελέσματος, διαχωρισμένες με κόμματα.

Η χρήση τέτοιων συντακτικών δομών καλείται *συνδυασμός επεκταμένων ερωτήσεων* ή, πιο σύντομα, *συνδυασμός ερωτήσεων* και οι *επεκταμένες ερωτήσεις* που μετέχουν σε μία *ερώτηση SQL-XE* που περιλαμβάνει συνδυασμό ερωτήσεων καλούνται *συνδυασμένες ερωτήσεις*.

Τέλος, μετά από όλες τις επαναλήψεις της συντακτικής δομής που περιγράψαμε, μπορεί να παρατίθεται μία πρόταση *ORDER BY*, η οποία καθορίζει την ταξινόμηση του αποτελέσματος.

Η αποτίμηση μιας *ερώτησης SQL-XE* που περιλαμβάνει συνδυασμένες ερωτήσεις γίνεται σύμφωνα με τον ακόλουθο αλγόριθμο:

1) υπολογίζονται και συγκρίνονται τα σχήματα των αποτελεσμάτων των δύο *επεκταμένων ερωτήσεων*, για να διαπιστωθεί αν είναι *συμβατά ως προς την ένωση*. Αν ο έλεγχος είναι επιτυχής, ο αλγόριθμος συνεχίζεται από το βήμα (2), ενώ στην αντίθετη περίπτωση προκύπτει σημασιολογικό σφάλμα και η *ερώτηση SQL-XE* θεωρείται άκυρη.
2) αποτιμάται η πρώτη *επεκταμένη ερώτηση*, όπως περιγράφηκε προηγουμένως, και το αποτέλεσμα αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο συμβολίζουμε με *Π1*.
3) αποτιμάται η επόμενη *επεκταμένη ερώτηση* και το αποτέλεσμα αποθηκεύεται σε έναν δεύτερο προσωρινό πίνακα, τον οποίο συμβολίζουμε με *Π2*.
4) στους πίνακες *Π1* και *Π2* εφαρμόζεται ένας από τους τελεστές της ΣΑΧΕ, και το αποτέλεσμα της εφαρμογής του τελεστή αντικαθιστά τον πίνακα *Π1*, ενώ ο πίνακας *Π2* καταστρέφεται. Ο τελεστής της ΣΑΧΕ που θα χρησιμοποιηθεί καθορίζεται σύμφωνα με τα ακόλουθα κριτήρια:

i) αν ανάμεσα στις δύο *επεκταμένες ερωτήσεις* παρεμβάλλεται μία από τις δεσμευμένες λέξεις *UNION* και *UNION ALL*, χωρίς να ακολουθείται από τον όρο *στήλες αποτελέσματος*, τότε χρησιμοποιείται ο τελεστής *UNION* της ΣΑΧΕ. Η διαφορά των δύο δεσμευμένων λέξεων είναι ότι η δεσμευμένη λέξη *UNION ALL* επιτρέπει εμφάνιση διπλοτύπων (duplicate) πλειάδων στο αποτέλεσμα, σε αντίθεση με τη δεσμευμένη λέξη *UNION*.

ii) αν ανάμεσα στις δύο *επεκταμένες ερωτήσεις* παρεμβάλλεται η δεσμευμένη λέξη *EXCEPT* χωρίς να ακολουθείται από τον όρο *στήλες αποτελέσματος*, τότε χρησιμοποιείται ο τελεστής *EXCEPT* της ΣΑΧΕ.

iii) αν ανάμεσα στις δύο *επεκταμένες ερωτήσεις* παρεμβάλλεται μία από τις δεσμευμένες λέξεις *UNION* και *UNION ALL*, ακολουθούμενη από τον όρο *στήλες αποτελέσματος*, τότε χρησιμοποιείται ο τελεστής *PUNION [στήλες αποτελέσματος]* της ΣΑΧΕ. Στην περίπτωση αυτή, όλες οι στήλες που αναφέρονται στον όρο *στήλες αποτελέσματος* που ακολουθεί τη δεσμευμένη λέξη θα πρέπει να ανήκουν στη *λίστα_εκφράσεων* της ερώτησης και να είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*, διαφορετικά προκύπτει σημασιολογικό σφάλμα και η *ερώτηση SQL-XE* θεωρείται άκυρη. Οι δεσμευμένες λέξεις *UNION* και *UNION ALL* σ' αυτή την περίπτωση είναι σημασιολογικά ισοδύναμες.

iv) αν ανάμεσα στις δύο *επεκταμένες ερωτήσεις* παρεμβάλλεται η δεσμευμένη λέξη *EXCEPT*, ακολουθούμενη από τον όρο *στήλες αποτελέσματος*, τότε χρησιμοποιείται ο τελεστής *PEXCEPT [στήλες αποτελέσματος]* της ΣΑΧΕ. Στην περίπτωση αυτή, όλες οι στήλες που αναφέρονται στον όρο *στήλες αποτελέσματος* που ακολουθεί τη δεσμευμένη λέξη θα πρέπει να ανήκουν στη *λίστα_εκφράσεων* της ερώτησης να είναι τύπου *χρονικό σημείο* ή *χρονικό διάστημα*, διαφορετικά προκύπτει σημασιολογικό σφάλμα και η *ερώτηση SQL-XE* θεωρείται άκυρη.

5) τα βήματα (2) έως (4) επαναλαμβάνονται και για τις υπόλοιπες συνδυασμένες ερωτήσεις, κατά σειρά εμφάνισής τους στην *ερώτηση SQL-XE*.

Όταν έχει ολοκληρωθεί ο υπολογισμός του αποτελέσματος των συνδυασμένων ερωτήσεων (ή της μοναδικής *επεκταμένης ερώτησης* αν η *ερώτηση SQL-XE* δεν περιλαμβάνει συνδυασμένες ερωτήσεις), τότε το αποτέλεσμα που βρίσκεται στον τελευταίο προσωρινό πίνακα που προέκυψε ταξινομείται σύμφωνα με τα προσδιοριζόμενα στην πρόταση *ORDER BY* (αν η πρόταση αυτή είναι παρούσα) και προωθείται προς το χρήστη ως το τελικό εξαγόμενο της *ερώτησης SQL-XE*.

Η ταξινόμηση για τους τύπους που ορίζονται στο πρότυπο SQL89 ακολουθεί το πρότυπο αυτό. Για τον τύπο *χρονικό σημείο*, η αύξουσα σειρά ταξινόμησης ορίζεται από τη σχέση *προηγείται χρονικά*, δηλαδή το χρονικό σημείο χ_1 θα προηγείται του χρονικού σημείου χ_κ στην αύξουσα σειρά ταξινόμησης, τότε και μόνον τότε αν χ_1 *προηγείται χρονικά* χ_κ. Η φθίνουσα σειρά ταξινόμησης ορίζεται ως η αντίστροφη της αύξουσας, ενώ αν δύο χρονικά σημεία είναι ίσα, δεν παρέχεται καμία εγγύηση για τη σχετική τους θέση στο τελικό αποτέλεσμα.

Για τον τύπο *χρονικό διάστημα* η αύξουσα σειρά ταξινόμησης ορίζεται από το κατηγόρημα *precedes*, δηλαδή το χρονικό διάστημα δ_1 προηγείται του χρονικού διαστήματος δ_2 στην αύξουσα σειρά ταξινόμησης, τότε και μόνον τότε αν δ_1 *precedes* δ_2. Η φθίνουσα σειρά ταξινόμησης ορίζεται ως η αντίστροφη της αύξουσας, ενώ αν δύο χρονικά διαστήματα είναι ίσα, δεν παρέχεται καμία εγγύηση για τη σχετική τους θέση στο τελικό αποτέλεσμα.

3.2.2. Εισαγωγή δεδομένων: Η εντολή INSERT.

Η σύνταξη της εντολής *INSERT* στην SQL-XE είναι

INSERT INTO *πίνακας* ['(' *λίστα_στηλών* ')'] *προσδιορισμός_τιμών* ;

όπου ο *προσδιορισμός_τιμών* ορίζεται ως

προσδιορισμός_τιμών ::= VALUES '(' *λίστα_εκφράσεων* ')'
| *επεκταμένη_ερώτηση*
;

Όπως φαίνεται από την περιγραφή του συντακτικού, η SQL-XE επεκτείνει το συντακτικό της αντίστοιχης εντολής της SQL89 κατά δύο τρόπους:

1. όταν οι προς εισαγωγή τιμές προσδιορίζονται μέσω της πρότασης *VALUES*, η *λίστα εκφράσεων* μπορεί να περιέχει οποιαδήποτε έκφραση είναι έγκυρη στην SQL-XE, συμπεριλαμβάνοντας έτσι σταθερές τύπου *χρονικό σημείο* και *χρονικό διάστημα*, καθώς και συναρτήσεις που δέχονται ορίσματα ή επιστρέφουν αποτελέσματα αυτών των τύπων.
2. όταν οι προς εισαγωγή τιμές προσδιορίζονται μέσω μιας ερώτησης, η ερώτηση αυτή μπορεί να είναι μία *επεκταμένη ερώτηση*.

Ο αλγόριθμος που ακολουθείται για την εκτέλεση μιας εντολής *INSERT* της SQL-XE εξαρτάται τόσο από τη σύνταξη της εντολής, όσο και από το αν ο πίνακας στον οποίο εισάγονται τα δεδομένα (τον οποίο θα συμβολίζουμε με *Π*) είναι κανονικοποιημένος ή έχει πρωτεύον κλειδί. Για την εκτέλεση μιας εντολής *INSERT* εφαρμόζεται ο ακόλουθος αλγόριθμος:

1) δημιουργείται ένας προσωρινός πίνακας *Σ*, του οποίου το σχήμα είναι ίδιο με το σχήμα του πίνακα *Π*. Ο πίνακας Σ θα περιέχει:
   i) το αποτέλεσμα της *επεκταμένης ερώτησης*, αν οι προς εισαγωγή τιμές προσδιορίζονται μέσω μιας επεκταμένης ερώτησης. Η επεκταμένη ερώτηση αποτιμάται σύμφωνα με όσα περιγράφησαν στην παράγραφο 3.2.1.
   ii) τις τιμές των εκφράσεων που παρατίθενται στη *λίστα εκφράσεων*, αν οι προς εισαγωγή τιμές προσδιορίζονται μέσω της πρότασης *VALUES*.
2) αν ο πίνακας *Π* δεν είναι κανονικοποιημένος και δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν, τότε οι πλειάδες του πίνακα *Σ* προστίθενται σ' αυτές του πίνακα *Π*.
3) αν ο πίνακας *Π* είναι κανονικοποιημένος στις στήλες σ_1, ..., σ_ν και δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν, τότε ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *Π PUNION* $[\sigma_1, ..., \sigma_\nu]$ *Σ*.
4) αν ο πίνακας *Π* δεν είναι κανονικοποιημένος και έχει ορισθεί σ' αυτόν πρωτεύον κλειδί, αποτελούμενο από τις στήλες κ_1, ..., κ_ν (σύμφωνα με τους σημασιολογικούς κανόνες της εντολής *CREATE TABLE* της SQL-XE, οι στήλες κ_1, ..., κ_ν θα συμμετέχουν στο πρωτεύον κλειδί ως *κλειδιά τύπου διαστήματος*), τότε:
   i) οι πλειάδες του πίνακα *Σ* εισάγονται στον πίνακα *Π*.
   ii) ελέγχεται ότι ο πίνακας *Π* δεν περιέχει ένα ζεύγος διαφορετικών πλειάδων π_1 και π_2, για τις οποίες $\pi_1.\kappa_\iota = \pi_2.\kappa_\iota$, $\forall\ \iota = 1, ..., \nu$. Αν ο έλεγχος αποτύχει, η εντολή εισαγωγής θεωρείται άκυρη, διότι παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού και ο πίνακας *Π* επανέρχεται στην κατάσταση που βρισκόταν πριν την έναρξη της εκτέλεσης της εντολής.
5) αν ο πίνακας *Π* είναι κανονικοποιημένος στις στήλες σ_1, ..., σ_ν και έχει ορισθεί σ' αυτόν πρωτεύον κλειδί που αποτελείται από τις στήλες κ_1,..., κ_μ, οι οποίες συμμετέχουν στο πρωτεύον κλειδί ως *κλειδιά τύπου διαστήματος* και τις στήλες σ_1, ..., σ_ν, οι οποίες συμμετέχουν στο πρωτεύον κλειδί ως *κλειδιά τύπου σημείο* (το σύνολο $\{\kappa_1,..., \kappa_\mu\}$ μπορεί να είναι κενό) τότε:
   i) οι πλειάδες του πίνακα *Σ* προστίθενται στον πίνακα *Π*.
   ii) ελέγχεται ότι ο πίνακας *Π* δεν περιέχει ένα ζεύγος διαφορετικών πλειάδων π_1 και π_2, για τις οποίες να συναληθεύουν οι ακόλουθες συνθήκες:
      a) $\pi_1.\kappa_\iota = \pi_2.\kappa_\iota$, $\forall\ \iota = 1, ..., \mu$.
      b) $\pi_1.\sigma_\iota\ cp\ \pi_2.\sigma_\iota$, $\forall\ \iota = 1, ..., \nu$.

Αν ο έλεγχος αποτύχει, η εντολή εισαγωγής θεωρείται άκυρη, διότι παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού και ο πίνακας *Π* επανέρχεται στην κατάσταση που βρισκόταν πριν την έναρξη της εκτέλεσης της εντολής.

iii) ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *NORMALISE* $[\sigma_1, ..., \sigma_v]$ *(Π)*.

Η εισαγωγή πλειάδων σε έναν κανονικοποιημένο πίνακα ενδέχεται να μην αυξήσει το συνολικό αριθμό πλειάδων στον πίνακα αυτό: ο τελικός αριθμός πλειάδων μπορεί να αυξηθεί, να παραμείνει ίσος ή να μειωθεί, σε σχέση με τον αρχικό, ανάλογα με τις συμπτύξεις που θα λάβουν χώρα. Έτσι η εισαγωγή της πλειάδας *(Πλάτων, Β, [1993-01-01, 1994-01-01))* στον πίνακα *Βαθμίδα* του σχήματος 3.2 θα οδηγήσει τον πίνακα στην κατάσταση που απεικονίζεται στο σχήμα 3.3, στην οποία ο πίνακας έχει λιγότερες πλειάδες. Η εισαγωγή πλειάδων σε έναν κανονικοποιημένο πίνακα αυξάνει έτσι το *συνολικό πλήθος χρονικών σημείων* για τα οποία υπάρχει πληροφορία και όχι απαραίτητα τον αριθμό των πλειάδων.

Βαθμίδα

Όνομα	Βαθμίδα	Διάστημα
Πλάτων	Β	[1992-01-01, 1993-01-01)
Πλάτων	Β	[1994-01-01, 1995-01-01)
Θαλής	Α	[1989-01-01, 1994-06-01)

Σχήμα 3.2 - Ο πίνακας *Βαθμίδα* πριν την εισαγωγή.

Βαθμίδα

Όνομα	Βαθμίδα	Διάστημα
Πλάτων	Β	[1992-01-01, 1995-01-01)
Θαλής	Α	[1989-01-01, 1994-06-01)

Σχήμα 3.3 - Ο πίνακας *Βαθμίδα* μετά την εισαγωγή.

3.2.3. Διαγραφή δεδομένων: Η εντολή DELETE.

Η σύνταξη της εντολής *DELETE* στην SQL-XE είναι η ακόλουθη:

DELETE FROM *πίνακας*

[PORTION *στήλη* = *έκφραση* {, *στήλη* = *έκφραση*}]

[WHERE *επεκταμένη_συνθήκη*] ;

Όπως προκύπτει από την περιγραφή του συντακτικού, η SQL-XE επεκτείνει το συντακτικό της αντίστοιχης εντολής της SQL89 κατά δύο τρόπους:

1. εισάγεται μία νέα πρόταση η *PORTION*, η σημασιολογία της οποίας θα αναλυθεί στη συνέχεια.
2. η συνθήκη που εμφανίζεται στην πρόταση *WHERE* μπορεί να είναι μία *επεκταμένη συνθήκη*, με τις επαυξήσεις που περιγράφονται στην παράγραφο 3.2.1.

Αν η πρόταση *PORTION* δεν είναι παρούσα, τότε η εντολή *DELETE* λειτουργεί ακριβώς όπως η αντίστοιχη εντολή της SQL89, διαγράφοντας τις πλειάδες για τις οποίες η *επεκταμένη συνθήκη* της πρότασης *WHERE* αληθεύει (αν η πρόταση *WHERE* δεν είναι παρούσα, τότε διαγράφονται όλες οι πλειάδες του πίνακα).

Αν πρόταση *PORTION* είναι παρούσα, τότε ο πίνακας από τον οποίο θα διαγραφούν τα δεδομένα (τον οποίο θα συμβολίζουμε με *Π*) θα πρέπει να είναι κανονικοποιημένος και οι στήλες που αναφέρονται στο αριστερό τμήμα των όρων *στήλη = έκφραση* της πρότασης *PORTION* πρέπει να ανήκουν στο σύνολο των στηλών κανονικοποίησης του πίνακα. Επίσης, ο τύπος του αποτελέσματος της έκφρασης που παρατίθεται στο δεξί τμήμα κάθε όρου *στήλη = έκφραση* της πρότασης *PORTION* πρέπει είναι ίδιος με τον τύπο της στήλης που αναφέρεται στο αριστερό τμήμα του ίδιου όρου. Σημασιολογικά, μία εντολή *DELETE* που περιέχει την πρόταση *PORTION* διαγράφει από τη βάση δεδομένων την πληροφορία που ικανοποιεί τη συνθήκη της πρότασης *WHERE* και αφορά τα χρονικά διαστήματα που προσδιορίζονται στην πρόταση *PORTION*.

Η διαδικασία που ακολουθείται για την εκτέλεση μιας εντολής *DELETE* της SQL-XE που περιέχει την πρόταση *PORTION* είναι η ακόλουθη:

1) δημιουργείται ένας προσωρινός πίνακας *P*, το σχήμα του οποίου είναι ίδιο με το σχήμα του πίνακα *Π*.
2) για κάθε πλειάδα του πίνακα *Π* που πληρεί την επεκταμένη συνθήκη της πρότασης *WHERE* και για την οποία κάθε στήλη που αναφέρεται στο αριστερό τμήμα ενός όρου *στήλη = έκφραση* της πρότασης *PORTION* έχει τιμή επικαλυπτόμενη με το αποτέλεσμα της *έκφρασης* που παρατίθεται στο δεξιό τμήμα του ίδιου όρου, εισάγεται στον πίνακα *P* μία πλειάδα, οι τιμές των στηλών της οποίας προσδιορίζονται σύμφωνα με τον ακόλουθο αλγόριθμο:
   i) αν η στήλη δεν αναφέρεται στο αριστερό τμήμα κάποιου όρου *στήλη = έκφραση* στην πρόταση *PORTION*, τότε χρησιμοποιείται η τιμή της αντίστοιχης στήλης στην εξεταζόμενη πλειάδα του πίνακα *Π*.

ii) αν η στήλη αναφέρεται στο αριστερό τμήμα κάποιου όρου *στήλη = έκφραση* στην πρόταση *PORTION*, τότε χρησιμοποιείται το αποτέλεσμα της συνάρτησης *intervsect(στήλη, έκφραση)*, όπου *στήλη* είναι η τιμή της στήλης στην εξεταζόμενη πλειάδα και *έκφραση* είναι το δεξιό μέλος του όρου.

3) τέλος, ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *Π PEXCEPT [σ_1, ..., σ_v] P*, όπου σ_1, ..., σ_v είναι οι στήλες κανονικοποίησης του πίνακα.

Η διαγραφή πλειάδων από έναν κανονικοποιημένο πίνακα με χρήση της πρότασης *PORTION* δεν είναι απαραίτητο να μειώσει το συνολικό αριθμό πλειάδων του πίνακα: ο τελικός αριθμός πλειάδων μπορεί να είναι μικρότερος, ίσος ή και μεγαλύτερος, συγκρινόμενος με τον αρχικό, ανάλογα με το αν διαγράφονται πλήρη διαστήματα (οπότε διαγράφονται και οι αντίστοιχες πλειάδες), υποδιαστήματα που περιέχουν τα άκρα των διαστημάτων ή υποδιαστήματα που δεν περιέχουν τα άκρα των διαστημάτων. Για παράδειγμα, η εκτέλεση της εντολής

```
DELETE FROM Βαθμίδα
PORTION Διάστημα = '[1993-01-01, 1994-01-01)'
WHERE Όνομα = 'Πλάτων'
```

όταν ο πίνακας *Βαθμίδα* βρίσκεται στην κατάσταση του σχήματος 3.3 θα οδηγήσει τον πίνακα στην κατάσταση του σχήματος 3.2, η οποία περιέχει περισσότερες πλειάδες. Η διαγραφή δεδομένων από έναν κανονικοποιημένο πίνακα μειώνει έτσι το *συνολικό πλήθος χρονικών σημείων* για τα οποία υπάρχει πληροφορία και όχι απαραίτητα τον αριθμό των πλειάδων.

Τέλος, η λειτουργία της διαγραφής δεδομένων δεν επηρεάζεται από το αν έχει ορισθεί πρωτεύον κλειδί στον πίνακα, διότι η διαγραφή δεδομένων δεν μπορεί να οδηγήσει σε παραβίαση της μοναδικότητας του πρωτεύοντος κλειδιού.

3.2.4. Ενημέρωση δεδομένων: Η εντολή UPDATE.

Η εντολή *UPDATE* στην SQL-XE επεκτείνεται κατά τρόπο παρόμοιο με την εντολή *DELETE*, προστίθεται δηλαδή μία πρόταση *PORTION* και επιτρέπεται η χρήση μιας επεκταμένης συνθήκης στην πρόταση *WHERE*. Η σύνταξη έτσι της εντολής *UPDATE* διαμορφώνεται ως ακολούθως:

UPDATE *πίνακας*

[PORTION *στήλη* = *έκφραση* {, *στήλη* = *έκφραση*}]

SET *στήλη* = *έκφραση* {, *στήλη* = *έκφραση*}

[WHERE *επεκταμένη_συνθήκη*] ;

Αν πρόταση *PORTION* είναι παρούσα, τότε ο πίνακας που θα ενημερωθεί θα πρέπει να είναι κανονικοποιημένος και οι στήλες που αναφέρονται στο αριστερό τμήμα των όρων *στήλη = έκφραση* της πρότασης *PORTION* πρέπει να ανήκουν στο σύνολο των στηλών κανονικοποίησης του πίνακα. Επίσης, ο τύπος του αποτελέσματος της έκφρασης που παρατίθεται στο δεξί τμήμα κάθε όρου *στήλη = έκφραση* των προτάσεων *PORTION* και *SET* πρέπει είναι ίδιος με τον τύπο της στήλης που αναφέρεται στο αριστερό τμήμα του ίδιου όρου. Στη συνέχεια της παραγράφου, θα χρησιμοποιούνται οι ακόλουθοι συμβολισμοί:

- ο πίνακας που υπόκειται σε ενημέρωση συμβολίζεται με *Π*.
- το σχήμα του πίνακα *Π* είναι $(\pi_1, \pi_2, \ldots, \pi_\kappa)$.
- αν ο πίνακας *Π* είναι κανονικοποιημένος, οι στήλες κανονικοποίησης συμβολίζονται με $\sigma_1, \ldots, \sigma_\nu$.
- τα κλειδιά *τύπου διαστήματος*, αν υπάρχουν, συμβολίζονται με $\kappa_1, \ldots, \kappa_\mu$.

Σύμφωνα με τα παραπάνω, αν ο πίνακας είναι κανονικοποιημένος *και* έχει ορισθεί κλειδί πάνω σ' αυτόν, το κλειδί αποτελείται από τις στήλες $\kappa_1, \ldots, \kappa_\mu, \sigma_1, \ldots, \sigma_\nu$.

Σημασιολογικά, η εντολή *UPDATE* επιτρέπει την τροποποίηση των δεδομένων που βρίσκονται σε έναν πίνακα. Αν η πρόταση *PORTION* δεν είναι παρούσα, τότε η ενημέρωση γίνεται σε επίπεδο πλειάδας, ενώ στην αντίθετη περίπτωση υπάρχει η περίπτωση να ενημερωθούν *τμήματα* πλειάδων που αφορούν συγκεκριμένες περιοχές του χρόνου εγκυρότητας των πλειάδων, ανάλογα με τα προσδιοριζόμενα στην πρόταση *PORTION*. Τέλος, αν έχει ορισθεί πρωτεύον κλειδί στον πίνακα, διενεργούνται έλεγχοι ώστε η λειτουργία της ενημέρωσης να μην παραβιάζει τη μοναδικότητα του κλειδιού, ενώ αν ο πίνακας είναι κανονικοποιημένος, η λειτουργία της ενημέρωσης ακολουθείται από μία πράξη κανονικοποίησης, πάνω στις στήλες κανονικοποίησης του πίνακα. Αναλυτικά, για τη διαδικασία που ακολουθείται για την εκτέλεση μιας εντολής *UPDATE* διακρίνονται οι ακόλουθες περιπτώσεις:

1) ο πίνακας *Π* δεν είναι κανονικοποιημένος και δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν: στην περίπτωση αυτή η πρόταση PORTION δεν μπορεί να χρησιμοποιηθεί (η

χρήση της οδηγεί σε σημασιολογικό σφάλμα, καθιστώντας άκυρη την εντολή *UPDATE*) και η εντολή εκτελείται όπως ορίζεται στο πρότυπο SQL89.

2) ο πίνακας *Π* δεν είναι κανονικοποιημένος και έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν: στην περίπτωση αυτή η εντολή εκτελείται όπως στην περίπτωση (1), αλλά μετά το πέρας της ενημέρωσης διενεργείται ένας έλεγχος για το αν στον πίνακα *Π* περιέχεται ένα ζεύγος διαφορετικών πλειάδων χ_1 και χ_2, τέτοιες ώστε $\chi_1.\kappa_\iota = \chi_2.\kappa_\iota \; \forall \; \iota = 1, \ldots, \mu$. Αν διαπιστωθεί η ύπαρξη ενός τέτοιου ζεύγους πλειάδων, τότε έχουμε παραβίαση της μοναδικότητας του πρωτεύοντος κλειδιού: η εντολή UPDATE θεωρείται άκυρη και ο πίνακας *Π* επανέρχεται στην κατάσταση που βρισκόταν πριν την έναρξη εκτέλεσης της εντολής.

3) ο πίνακας *Π* είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και η πρόταση *PORTION* δεν είναι παρούσα: στην περίπτωση αυτή οι πλειάδες του πίνακα *Π* που ικανοποιούν τη συνθήκη της πρότασης *WHERE* ενημερώνονται όπως ορίζεται στην πρόταση *SET* και η λειτουργία της ενημέρωσης ακολουθείται από μία πράξη κανονικοποίησης πάνω στις στήλες κανονικοποίησης του πίνακα *Π*.

4) ο πίνακας *Π* είναι κανονικοποιημένος, η πρόταση *PORTION* δεν είναι παρούσα και έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν: οι πλειάδες του πίνακα *Π* που ικανοποιούν τη συνθήκη της πρότασης *WHERE* ενημερώνονται όπως ορίζεται στην πρόταση *SET* και στη συνέχεια διενεργείται έλεγχος για το αν στον πίνακα *Π* περιέχεται ένα ζεύγος διαφορετικών πλειάδων χ_1 και χ_2, τέτοιες ώστε:

   i) $\chi_1.\kappa_\iota = \chi_2.\kappa_\iota \; \forall \; \iota = 1, \ldots, \mu$.

   ii) $\chi_1.\sigma_\iota \; cp \; \chi_2.\sigma_\iota \; \forall \; \iota = 1, \ldots, \nu$.

Αν διαπιστωθεί η ύπαρξη ενός τέτοιου ζεύγους πλειάδων, τότε έχουμε παραβίαση της μοναδικότητας του πρωτεύοντος κλειδιού: η εντολή *UPDATE* θεωρείται άκυρη και ο πίνακας *Π* επανέρχεται στην κατάσταση που βρισκόταν πριν την έναρξη εκτέλεσης της εντολής. Στην αντίθετη περίπτωση, ο αλγόριθμος τερματίζεται με μία πράξη κανονικοποίησης πάνω στις στήλες κανονικοποίησης του πίνακα *Π*.

5) ο πίνακας *Π* είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και η πρόταση *PORTION* είναι παρούσα έχοντας τη μορφή *PORTION* $\tau_1 = \varepsilon_1, \ldots, \tau_\kappa = \varepsilon_\kappa$: στην περίπτωση αυτή ακολουθείται ο εξής αλγόριθμος:

   i) δημιουργείται ένας προσωρινός πίνακας *P1*, το σχήμα του οποίου είναι ίδιο με το σχήμα του *Π*. Για κάθε πλειάδα του πίνακα *Π* που ικανοποιεί τη συνθήκη της

πρότασης *WHERE* και για την οποία τ_ι *cp* ε_ι, $\forall$ ι = 1, ..., κ, εισάγεται στον πίνακα *P1* μία πλειάδα, οι τιμές των στηλών της οποίας καθορίζονται σύμφωνα με τα εξής κριτήρια:

a) αν η στήλη αναφέρεται στο αριστερό μέλος ενός όρου $\tau_i = \varepsilon_i$ της πρότασης *PORTION*, τότε η τιμή της στήλης θα είναι το αποτέλεσμα της συνάρτησης *intervsect*(*στήλη*, ε_i), όπου *στήλη* είναι η τιμή της στήλης στην εξεταζόμενη πλειάδα και ε_i είναι το δεξιό μέλος του όρου.

b) στην αντίθετη περίπτωση, η τιμή της στήλης θα είναι ίση με την τιμή της αντίστοιχης στήλης της εξεταζόμενης πλειάδας.

ii) δημιουργείται ένας προσωρινός πίνακας *P2*, το σχήμα του οποίου είναι ίδιο με το σχήμα του πίνακα *Π*. Για κάθε πλειάδα του πίνακα *Π* που ικανοποιεί τη συνθήκη της πρότασης *WHERE* και για την οποία τ_ι *cp* ε_ι, $\forall$ ι = 1, ..., κ, εισάγεται στον πίνακα *P2* μία πλειάδα, οι τιμές των στηλών της οποίας καθορίζονται σύμφωνα με τα εξής κριτήρια:

a) αν η στήλη αναφέρεται στο αριστερό τμήμα ενός όρου *στήλη* = *έκφραση* της πρότασης *SET*, τότε χρησιμοποείται η τιμή της *έκφρασης* στο αριστερό τμήμα του ίδιου όρου.

b) αν η στήλη δεν αναφέρεται στο αριστερό τμήμα κανενός όρου *στήλη* = *έκφραση* της πρότασης *SET* και αναφέρεται στο αριστερό τμήμα ενός όρου $\tau_i = \varepsilon_i$ της πρότασης *PORTION*, τότε η τιμή της στήλης θα είναι το αποτέλεσμα της συνάρτησης *intervsect*(*στήλη*, ε_i), όπου *στήλη* είναι η τιμή της στήλης στην εξεταζόμενη πλειάδα και ε_i είναι το δεξιό μέλος του όρου.

c) αν η στήλη δεν αναφέρεται στο αριστερό τμήμα κανενός όρου *στήλη* = *έκφραση* των προτάσεων *SET* και *PORTION*, τότε χρησιμοποιείται η τιμή της στήλης στην εξεταζόμενη πλειάδα.

iii) ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *Π PEXCEPT* [σ_1, ..., σ_v] *P1*.

iv) ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *Π PUNION* [σ_1, ..., σ_v] *P2*.

6) ο πίνακας *Π* είναι κανονικοποιημένος, έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και η πρόταση *PORTION* είναι παρούσα έχοντας τη μορφή *PORTION* $\tau_1 = \varepsilon_1$, ..., $\tau_\kappa = \varepsilon_\kappa$: στην

περίπτωση αυτή εκτελούνται τα βήματα (i) έως (iii) της περίπτωσης (5) και ο αλγόριθμος συνεχίζει ως ακολούθως:

iv) ο πίνακας *Π* αντικαθίσταται από το αποτέλεσμα της πράξης *Π* UNION *P2*

v) διενεργείται έλεγχος για το αν στον πίνακα *Π* περιέχεται ένα ζεύγος διαφορετικών πλειάδων χ_1 και χ_2, τέτοιες ώστε:

- $\chi_1.\kappa_\iota = \chi_2.\kappa_\iota \; \forall \; \iota = 1, \ldots, \mu$.
- $\chi_1.\sigma_\iota \; cp \; \chi_2.\sigma_\iota \; \forall \; \iota = 1, \ldots, \nu$.

Αν διαπιστωθεί η ύπαρξη ενός τέτοιου ζεύγους πλειάδων, τότε έχουμε παραβίαση της μοναδικότητας του πρωτεύοντος κλειδιού: η εντολή UPDATE θεωρείται άκυρη και ο πίνακας *Π* επανέρχεται στην κατάσταση που βρισκόταν πριν την έναρξη εκτέλεσης της εντολής.

vi) ο πίνακας Π αντικαθίσταται από το αποτέλεσμα της πράξης NORMALISE $[\sigma_1, \ldots, \sigma_\nu]$ *(Π)*.

Η ενημέρωση πλειάδων σε έναν κανονικοποιημένο πίνακα ενδέχεται να τροποποιήσει τον αριθμό πλειάδων που περιέχονται σ' αυτόν. Για παράδειγμα, η εκτέλεση της εντολής

```
UPDATE Βαθμίδα
PORTION Διάστημα = [1994-01-01, 1995-01-01)
SET Βαθμίδα = 'A'
WHERE Όνομα = 'Πλάτων'
```

όταν ο πίνακας *Βαθμίδα* βρίσκεται στην κατάσταση του σχήματος 3.3, θα οδηγήσει τον πίνακα στην κατάσταση του σχήματος 3.4. Στην κατάσταση αυτή ο πίνακας περιέχει περισσότερες πλειάδες απ' ότι στην αρχική.

Βαθμίδα

Όνομα	Βαθμίδα	Διάστημα
Πλάτων	B	[1992-01-01, 1994-01-01)
Πλάτων	A	[1994-01-01, 1995-01-01)
Θαλής	A	[1989-01-01, 1994-06-01)

Σχήμα 3.4 - Ο πίνακας *Βαθμίδα* μετά την ενημέρωση.

3.3. Οι εντολές ελέγχου στην SQL-XE.

Οι εντολές ελέγχου της SQL-XE επιτρέπουν τη δημιουργία και την καταστροφή δεικτών, την εργασία με δοσοληψίες και τον προσδιορισμό δικαιωμάτων χρηστών για προσπέλαση σε πίνακες. Στις επόμενες παραγράφους περιγράφονται οι εντολές ελέγχου της SQL-XE.

3.3.1. Δημιουργία Δεικτών: Η Εντολή CREATE INDEX.

Όπως και στην SQL89, έτσι και στην SQL-XE η εντολή *CREATE INDEX* δημιουργεί μία δευτερεύουσα δομή δείκτη πάνω σε έναν πίνακα, η οποία μπορεί να χρησιμοποιηθεί για ταχύτερη προσπέλαση στα δεδομένα. Η σύνταξη της εντολής είναι

CREATE [UNIQUE] INDEX *δείκτης* ON *όνομα_πίνακα* '(' *στήλες_δείκτη* ')' ;

όπου ο όρος *στήλες δείκτη* ορίζεται ως

στήλες_δείκτη ::= *στήλη* [ASC | DESC] {, *στήλη* [ASC | DESC] } ;

Ο νέος δείκτης θα δημιουργηθεί πάνω στον πίνακα *όνομα πίνακα* και θα έχει το όνομα *δείκτης*. Στη δομή δείκτη θα συμμετέχουν οι στήλες που αναφέρονται στον όρο *στήλες δείκτη*, κάθε μία από τις οποίες θα πρέπει να ανήκει στο σχήμα της σχέσης *όνομα πίνακα*. Η σειρά ταξινόμησης της κάθε στήλης, στα πλαίσια του δείκτη, θα είναι φθίνουσα αν το όνομά της ακολουθείται από τη δεσμευμένη λέξη *DESC* ή αύξουσα στην αντίθετη περίπτωση. Αν ο προσδιορισμός *UNIQUE* είναι παρών στην εντολή, τότε δεν επιτρέπεται να υπάρχει στον πίνακα πάνω στον οποίο δημιουργείται ο δείκτης, ζεύγος πλειάδων, των οποίων όλες οι στήλες που αναφέρονται στον όρο *στήλες δείκτη* να έχουν ίσες τιμές.

3.3.2. Διαγραφή Δεικτών: Η Εντολή DROP INDEX.

Η σύνταξη της εντολής DROP INDEX είναι

DROP INDEX *όνομα_δείκτη* ;

και το αποτέλεσμά της είναι να καταστραφεί η προσδιοριζόμενη δομή δείκτη, η οποία θα πρέπει να έχει δημιουργηθεί προηγουμένως με μία εντολή *CREATE INDEX*.

3.3.3. Δοσοληψίες: Οι εντολές BEGIN TRANSACTION, COMMIT και ROLLBACK.

Η SQL-XE υποστηρίζει δοσοληψίες, με την ίδια ακριβώς σημασιολογία που αυτές ορίζονται στην SQL89. Η αρχή μιας δοσοληψίας δηλώνεται με την εντολή

BEGIN TRANSACTION ;

και η τρέχουσα δοσοληψία τερματίζεται με μία από τις εντολές

COMMIT [WORK] ;

η οποία μονιμοποιεί τα αποτελέσματα των εντολών που εκτελέστηκαν κατά τη διάρκεια της τρέχουσας δοσοληψίας και

ROLLBACK [WORK] ;

η οποία αναιρεί αποτελέσματα των εντολών που εκτελέστηκαν κατά τη διάρκεια της τρέχουσας δοσοληψίας, επαναφέροντας τη βάση δεδομένων στην κατάσταση που βρισκόταν όταν εκτελέστηκε η τελευταία εντολή *BEGIN TRANSACTION*. Η SQL-XE δεν υποστηρίζει *ένθετες* (nested) δοσοληψίες και έτσι μόνο μία δοσοληψία ανά σύνοδο (session) μπορεί να είναι ενεργός ανά πάσα χρονική στιγμή.

3.3.4. Προσδιορισμός Δικαιωμάτων: Οι εντολές GRANT και REVOKE.

Οι εντολές *GRANT* και *REVOKE* χρησιμοποιούνται στην SQL-XE για να προσδιορίσουν τα δικαιώματα των χρηστών πάνω στους πίνακες που περιέχονται στη βάση δεδομένων. Η σύνταξη και η σημασιολογία των εντολών δεν διαφοροποιείται, σε σχέση με τις αντίστοιχες εντολές της SQL89, έτσι η σύνταξη της εντολής GRANT στην SQL-XE είναι

GRANT *δικαίωμα* {, *δικαίωμα*} ON *όνομα_πίνακα* TO *χρήστης* {, *χρήστης*} ;

όπου ο όρος *δικαίωμα* ορίζεται ως

δικαίωμα ::= SELECT | DELETE | UPDATE '(' *λίστα_στηλών* ')' ;

και *χρήστης* είναι μία ταυτότητα χρήστη. Η εκτέλεση της εντολής έχει ως αποτέλεσμα να εκχωρούνται στους προσδιορισθέντες χρήστες τα δικαιώματα που αναφέρονται αμέσως μετά τη δεσμευμένη λέξη *GRANT*. Τα δικαιώματα ανάκτησης και διαγραφής ορίζονται σε επίπεδο πλειάδας, ενώ το δικαίωμα ενημέρωσης μπορεί να παραχωρηθεί σε επίπεδο στήλης.

Αντίστοιχα, η σύνταξη της εντολής REVOKE είναι

REVOKE *δικαίωμα* {, *δικαίωμα*} ON *όνομα_πίνακα*

FROM *χρήστης* {, *χρήστης*} ;

και αφαιρεί από τους προσδιορισθέντες χρήστες τα δικαιώματα που αναφέρονται αμέσως μετά τη δεσμευμένη λέξη *REVOKE*.

3.4. Σύνοψη.

Στο κεφάλαιο αυτό παρουσιάσαμε την SQL-XE. Η SQL-XE επεκτείνει το συντακτικό και τη σημασιολογία των εντολών της SQL89, παρέχοντας δυνατότητα ορισμού πινάκων που περιέχουν χρονικά δεδομένα, καθώς και διαχείρισης των χρονικών δεδομένων. Στο επόμενο κεφάλαιο θα παρουσιαστεί ο σχεδιασμός και η υλοποίηση ενός χρονολογικού συστήματος διαχείρισης βάσεων δεδομένων το οποίο χρησιμοποιεί την SQL-XE ως γλώσσα ορισμού και διαχείρισης δεδομένων.

4. Μία βελτιστοποιημένη υλοποίηση ενός χρονολογικού ΣΔΒΔ.

Τα χρονολογικά ΣΔΒΔ έχουν συγκεντρώσει σημαντικό ερευνητικό ενδιαφέρον κατά την τελευταία δεκαετία. Σε αρκετές εργασίες προτείνονται μοντέλα για την αναπαράσταση και αποθήκευση χρονολογικών δεδομένων ([Elmarsi83], [Ariav86], [Clifford87], [McKenzie87], [Gadia88], [Sarda90], [Navathe93]), τα περισσότερα από τα οποία αποτελούν επέκταση του σχεσιακού μοντέλου, ενώ πιο πρόσφατα έχουν διερευνηθεί και επεκτάσεις του αντικειμενοστρεφούς μοντέλου ([Caruso88], [Caruso90], [Carey88], [Rose91], [Kδfer92], [Rose93]). Άλλες εργασίες προτείνουν γλώσσες ορισμού και διαχείρισης δεδομένων, οι οποίες ενσωματώνουν χρονολογική σημασιολογία ([Jones79], [Snodgrass87], [Gadia88], [Tansel91], [Navathe93], [Snodgrass94δ], [Sciore94]). Όμως, παρά το αυξημένο ενδιαφέρον που παρουσιάζουν τα χρονολογικά ΣΔΒΔ, ο αριθμός των υλοποιήσεων είναι μικρός: στο [Jones79] παρουσιάζεται η γλώσσα LEGOL, η οποία επρόκειτο να υλοποιηθεί, στο [Snodgrass87] περιγράφεται η υλοποίηση της γλώσσας TQUEL, στο [Stonebreaker90] περιγράφεται η υλοποίηση του Postgress, μιας αντικειμενοστρεφούς επέκτασης του Ingres που συμπεριλαμβάνει χρονολογικά στοιχεία και στο [Tansel91] αναφέρεται ως αντικείμενο μελλοντικής εργασίας η τροποποίηση του Ingres, με στόχο την υλοποίηση ενός πρωτοτύπου της γλώσσας HQUEL. Πιο πρόσφατα, στο [ESPRIT94] παρουσιάζεται η υλοποίηση ενός ΣΔΒΔ που υποστηρίζει δεδομένα με μονοδιάστατο χρόνο εγκυρότητας και χρησιμοποιεί την SQL-XE ως γλώσσα ορισμού και διαχείρισης δεδομένων.

Η *απόδοση* ενός ΣΔΒΔ είναι μία ιδιαίτερα σημαντική συνιστώσα, η οποία πρέπει να ληφθεί υπόψη κατά την υλοποίηση ενός ΣΔΒΔ. Συστήματα που προσφέρουν πλούσια χρονολογικά σημασιολογικά χαρακτηριστικά δεν θα τύχουν της αποδοχής των χρηστών αν η απόδοσή τους δεν είναι ικανοποιητική: οι χρήστες θα καταφύγουν σε ειδικά διαμορφωμένες για την κάθε περίπτωση εφαρμογές, οι οποίες θα καλύπτουν τις ανάγκες τους με αποδεκτή απόδοση.

Στο κεφάλαιο αυτό θα παρουσιάσουμε τον σχεδιασμό και την υλοποίηση ενός βελτιστοποιημένου χρονολογικού ΣΔΒΔ, το οποίο χρησιμοποιεί την SQL-XE ως γλώσσα ορισμού και διαχείρισης δεδομένων. Η βελτιστοποίηση της απόδοσης τεκμηριώνεται τόσο βάσει ενός μαθηματικού μοντέλου κόστους όσο και από διαγράμματα απόδοσης.

4.1. Σχεδιασμός και αρχιτεκτονική του χρονολογικού ΣΔΒΔ.

Για την υλοποίηση ενός χρονολογικού ΣΔΒΔ μπορούν να ακολουθηθούν δύο κατευθύνσεις:

1) να επεκταθεί κάποιο από τα υπάρχοντα ΣΔΒΔ.

2) το χρονολογικό ΣΔΒΔ να αναπτυχθεί από μηδενική βάση, δηλαδή να υλοποιηθούν εξ αρχής όλα τα τμήματα του ΣΔΒΔ.

Η δεύτερη προσέγγιση παρέχει μεγαλύτερη ευελιξία σε σχέση με την πρώτη, επιτρέποντας μεταξύ άλλων τη χρήση οποιουδήποτε μοντέλου φυσικής αποθήκευσης κριθεί καταλληλότερο για τα χρονολογικά δεδομένα και στρατηγικών προσπέλασης και εκτέλεσης πράξεων που αξιοποιούν τα ιδιαίτερα χαρακτηριστικά των χρονολογικών δεδομένων. Το κύριο μειονέκτημα της εναλλακτικής αυτής λύσης είναι ο όγκος της απαιτούμενης εργασίας: η ανάπτυξη ενός πλήρους χρονολογικού ΣΔΒΔ εξ αρχής είναι έργο εξαιρετικά μεγάλης κλίμακας και απαιτεί πολλούς ανθρωπομήνες εργασίας για να ολοκληρωθεί. Τα διάφορα τμήματα του ΣΔΒΔ θα πρέπει να κωδικοποιηθούν, να ελεγχθούν για ύπαρξη σφαλμάτων και να βελτιστοποιηθούν. Επιπροσθέτως, η απόδοση των νέων τεχνικών θα πρέπει να συγκριθεί με αυτή των ήδη υπαρχουσών τεχνικών, προκειμένου να διαπιστωθεί αν πραγματικά επιφέρουν την προσδοκώμενη βελτίωση.

Σύμφωνα με τα παραπάνω, η μόνη εφικτή προσέγγιση στην υλοποίηση του χρονολογικού ΣΔΒΔ είναι να επεκταθεί ένα από τα υπάρχοντα ΣΔΒΔ. Η επέκταση αυτή μπορεί να γίνει με τους ακόλουθους τρόπους:

1) με απευθείας επέμβαση στον πυρήνα του ΣΔΒΔ. Η επέμβαση αυτή περιλαμβάνει τα ακόλουθα:
   i) την προσθήκη στον πυρήνα των απαραίτητων τύπων δεδομένων (*χρονικό διάστημα*), καθώς και των συναρτήσεων και των κατηγορημάτων που έχουν ορισθεί πάνω σ' αυτούς τους τύπους.
   ii) την τροποποίηση του συντακτικού αναλυτή του ΣΔΒΔ, ώστε να αναγνωρίζει το εμπλουτισμένο συντακτικό της γλώσσας ορισμού και διαχείρισης δεδομένων και να δημιουργεί τις κατάλληλες δομές δεδομένων που θα επιτρέπουν στο υποσύστημα εκτέλεσης να προβεί στις κατάλληλες ενέργειες.
   iii) την τροποποίηση του υποσυστήματος εκτέλεσης, ώστε να εξετάζει τις επιπρόσθετες πληροφορίες που θα παράγει ο συντακτικός αναλυτής, καθώς και τις ιδιότητες των πινάκων (κανονικοποίηση, ύπαρξη πρωτεύοντος κλειδιού) και να διαμορφώνει σύμφωνα με αυτά το σχέδιο εκτέλεσης της ερώτησης (query execution plan). Η βιβλιοθήκη αλγορίθμων του υποσυστήματος εκτέλεσης θα πρέπει επίσης να εμπλουτιστεί με υλοποιήσεις των πράξεων *FOLD*, *UNFOLD*, *NORMALISE*, *PUNION* και *PEXCEPT* της ΣΑΧΕ.

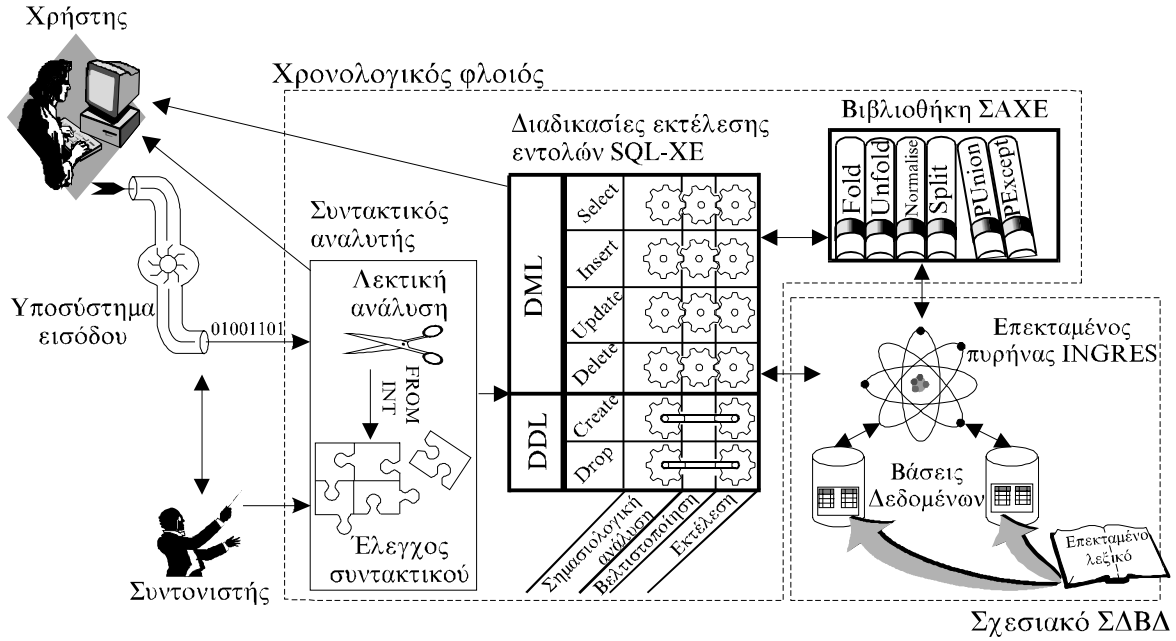
2) με τη δημιουργία ενός *χρονολογικού φλοιού* (temporal engine), δηλαδή μιας μετωπικής εφαρμογής (front-end application), η οποία θα δέχεται τις εντολές στην εμπλουτισμένη γλώσσα ορισμού και διαχείρισης δεδομένων και θα τις απεικονίζει σε εντολές προς το ΣΔΒΔ, οι οποίες, βέβαια, δεν θα χρησιμοποιούν τις συντακτικές και σημασιολογικές επεκτάσεις. Ο χρονολογικός φλοιός θα είναι επίσης υπεύθυνος για τη διαχείριση των νέων τύπων δεδομένων, μετατρέποντας τα δεδομένα των τύπων αυτών σε κάποια αποδεκτή από το ΣΔΒΔ μορφή προκειμένου να αποθηκευθούν και εφαρμόζοντας τους αντίστροφους μετασχηματισμούς, όταν αυτά πρέπει να παρουσιαστούν στον χρήστη. Σύμφωνα με την προσέγγιση αυτή, το υπάρχον ΣΔΒΔ χρησιμοποιείται για την αποθήκευση και ανάκτηση δεδομένων, αλλά μπορεί επίσης να χρησιμοποιηθεί για την εκτέλεση των εντολών της εμπλουτισμένης γλώσσας που δεν χρησιμοποιούν τα επιπρόσθετα σημασιολογικά χαρακτηριστικά και τους νέους τύπους δεδομένων, ενώ ο χρονολογικός φλοιός αναλαμβάνει το χειρισμό των εντολών που δεν μπορούν να εκτελεστούν άμεσα από το ΣΔΒΔ.

Η απευθείας επέμβαση στον πυρήνα του ΣΔΒΔ πλεονεκτεί, σε σχέση με τη δημιουργία μετωπικής εφαρμογής, στον τομέα της απόδοσης: όλες οι πράξεις εκτελούνται στο εσωτερικό του ΣΔΒΔ και τα δεδομένα μετακινούνται εκτός του πυρήνα μόνο για να παρουσιαστούν στον χρήστη. Επίσης, δεν είναι απαραίτητο να εφαρμοστούν μετασχηματισμοί πάνω στα δεδομένα των νέων τύπων, διότι ο πυρήνας μπορεί να χειρίζεται αυτούς τους τύπους άμεσα. Τέλος, μία συντακτική ανάλυση της εντολής είναι αρκετή, ενώ στη δεύτερη προσέγγιση ο χρονολογικός φλοιός θα αναλύσει συντακτικά την κάθε εντολή και θα την απεικονίσει σε ένα σύνολο εντολών προς το ΣΔΒΔ, κάθε μία από τις οποίες θα αναλυθεί συντακτικά από το ΣΔΒΔ.

Μία τέτοια επέμβαση στον πυρήνα ενός ΣΔΒΔ προϋποθέτει ότι υπάρχει πρόσβαση στον πρωτογενή κώδικα (source code) του ΣΔΒΔ, κάτι που σπάνια είναι εφικτό. Επίσης, μία τέτοια επέκταση είναι άρρηκτα συνδεδεμένη με το ΣΔΒΔ για οποίο αναπτύσσεται, μια και ο κώδικας θα είναι προσαρμοσμένος στην εσωτερική αρχιτεκτονική του. Αντίθετα, ένας χρονολογικός φλοιός μπορεί να μεταφερθεί πολύ πιο εύκολα σε άλλα ΣΔΒΔ (ειδικότερα αν αυτά χρησιμοποιούν την ίδια γλώσσα ορισμού και διαχείρισης δεδομένων με το ΣΔΒΔ για το οποίο αρχικά αναπτύχθηκε ο φλοιός), απαιτώντας μόνο μικρές αλλαγές, στα σημεία όπου πρέπει να χρησιμοποιηθούν ιδιαίτερα χαρακτηριστικά του ΣΔΒΔ.

Δεδομένου ότι στα πλαίσια της έρευνας δεν υπήρχε πρόσβαση στον πρωτογενή κώδικα κανενός ΣΔΒΔ, αναγκαστικά ακολουθήθηκε η δεύτερη προσέγγιση, με τη διαφορά ότι

αξιοποιήθηκε το εργαλείο *Object Manager*[2], με το οποίο ενσωματώθηκαν στον πυρήνα του Ingres οι απαραίτητοι τύποι δεδομένων και οι πράξεις (συναρτήσεις και κατηγορήματα) πάνω σ' αυτούς[3]. Η συνολική αρχιτεκτονική του χρονολογικού ΣΔΒΔ, όπως τελικά υλοποιήθηκε, απεικονίζεται στο σχήμα 4.1.



Σχήμα 4.1 - Αρχιτεκτονική του χρονολογικού ΣΔΒΔ.

4.2. Περιορισμοί που τέθηκαν στην υλοποίηση.

Στο χρονολογικό ΣΔΒΔ που υλοποιήθηκε έχει ενσωματωθεί η θεμελιώδης λειτουργικότητα της SQL-XE, για λόγους απλότητας, όμως, επιβλήθηκαν οι κάτωθι περιορισμοί:

1) *περιορισμός των υποστηριζόμενων διαστάσεων χρόνου εγκυρότητας*. Η τρέχουσα υλοποίηση υποστηρίζει μόνο μία διάσταση χρόνου εγκυρότητας για τους πίνακες ή, ισοδύναμα, μόνο μία στήλη τύπου *χρονικό διάστημα* μπορεί να δηλώνεται στην πρόταση *NORMALISED* της εντολής *CREATE TABLE*. Ο περιορισμός αυτός έδωσε τη δυνατότητα σχεδιασμού και κωδικοποίησης αποδοτικών αλγορίθμων, χωρίς να επηρεάζει σε μεγάλο βαθμό την έκταση του πεδίου εφαρμογών στις οποίες μπορεί να χρησιμοποιηθεί το χρονολογικό ΣΔΒΔ, δεδομένου ότι για τη μοντελοποίηση των περισσοτέρων προβλημάτων επαρκεί μία διάσταση χρόνου εγκυρότητας.

---

[2]Ï Ο λογότυπος "Object Manager" είναι σήμα κατατεθέν της Ingres Corporation.

[3] Η ενσωμάτωση των νέων τύπων δεδομένων και των πράξεων στον πυρήνα του Ingres έγινε από την Information Dynamics, στα πλαίσια του προγράμματος ORES.

Σημειώνουμε στο σημείο αυτό, ότι ο συντακτικός αναλυτής επιτρέπει τη χρήση όλων των συντακτικών δομών που ορίζουν πολλαπλές διαστάσεις χρόνου εγκυρότητας (πρόταση *NORMALISED* της εντολής *CREATE TABLE* και πρόταση *PORTION* στις εντολές *DELETE* και *UPDATE*). Ο περιορισμός τίθεται από τις διαδικασίες εκτέλεσης, οι οποίες είναι το μόνο τμήμα του χρονολογικού φλοιού που χρήζει τροποποίησης, προκειμένου να υποστηριχθούν περισσότερες διαστάσεις χρόνου εγκυρότητας. Επίσης κανείς περιορισμός δεν τίθεται στο πλήθος των στηλών τύπου *χρονικό διάστημα* που μπορούν να συμμετέχουν στο σχήμα κάποιου πίνακα.

2) *Περιορισμός στη διακριτότητα των χρονικών σημείων και χρονικών διαστημάτων*. Η ενσωμάτωση των νέων τύπων δεδομένων καθώς και των πράξεων (συναρτήσεων και κατηγορημάτων) πάνω σ' αυτούς έγινε στα πλαίσια του προγράμματος ORES, χρησιμοποιώντας το εργαλείο *Object Manager*. Στα πλαίσια του προγράμματος ORES δημιουργήθηκε μόνο ένας τύπος χρονικού διαστήματος με διακριτότητα ημέρας (ο τύπος *DATEINTERVAL)*, ενώ για τα χρονικά σημεία χρησιμοποιήθηκε ο τύπος *DATE* που υποστηρίζεται άμεσα από τον πυρήνα του Ingres. Δεδομένου ότι το εργαλείο *Object Manager* δεν ήταν διαθέσιμο, δεν ήταν δυνατή η δημιουργία και άλλων τύπων διαστημάτων και χρονικών σημείων.

3) *Περιορισμός της χρήσης των προτάσεων REFORMAT AS και NORMALISE στις υποερωτήσεις*. Σύμφωνα με τη σημασιολογία της SQL-XE, η παρουσία των προτάσεων *REFORMAT AS* και *NORMALISE* σε μία υποερώτηση προκαλεί την εφαρμογή των προσδιοριζόμενων πράξεων της ΣΑΧΕ (*FOLD*, *UNFOLD* και *NORMALISE*) στο αποτέλεσμα της υποερώτησης και στη συνέχεια την εκτέλεση της σύνδεσης (ή της αντισύνδεσης-antijoin) με το αποτέλεσμα της εξωτερικής ερώτησης. Δεδομένου ότι οι πράξεις της ΣΑΧΕ *FOLD*, *UNFOLD* και *NORMALISE* δεν ενσωματώθηκαν στον πυρήνα του Ingres, αλλά η εκτέλεσή τους γίνεται από τον χρονολογικό φλοιό, η παρουσία των προτάσεων *REFORMAT AS* και *NORMALISE* σε μία υποερώτηση θα απαιτούσε από τον χρονολογικό φλοιό να προβεί στις ακόλουθες ενέργειες (ο αλγόριθμος παρουσιάζεται χωρίς λεπτομέρειες):

   i) χρήση του σχεσιακού ΣΔΒΔ για την αποτίμηση του τμήματος *SELECT/FROM/WHERE* της εξωτερικής ερώτησης (από τη συνθήκη του τμήματος *WHERE* θα απουσιάζει η υποερώτηση). Σε κάθε πλειάδα του αποτελέσματος ανατίθεται από τον χρονολογικό φλοιό ένας μοναδικός προσδιοριστής και η επεκταμένη πλειάδα αποθηκεύεται σε έναν προσωρινό πίνακα.

ii) μεταβίβαση του ελέγχου στο χρονολογικό ΣΔΒΔ προκειμένου να αποτιμηθεί το αποτέλεσμα της σύνδεσης μεταξύ του προσωρινού πίνακα που προέκυψε στο βήμα (i) και του τμήματος της υποερώτησης που αποτελείται από τις προτάσεις *SELECT*, *FROM*, *WHERE*, *GROUP BY* και *HAVING*. Το αποτέλεσμα της σύνδεσης αποθηκεύεται σε έναν προσωρινό πίνακα.

iii) εκτέλεση από το χρονολογικό φλοιό των πράξεων της ΣΑΧΕ που ορίζονται από τις προτάσεις *REFORMAT AS* και *NORMALISE* της υποερώτησης πάνω στον προσωρινό πίνακα που προέκυψε στο βήμα (ii). Από τις πλειάδες του αποτελέσματος ο χρονολογικός φλοιός επιλέγει αυτές που πληρούν το κατηγόρημα της υποερώτησης, αφαιρεί το μοναδικό προσδιοριστή και τις αποθηκεύει σε έναν προσωρινό πίνακα.

iv) μεταβίβαση του ελέγχου στον πυρήνα του Ingres για να συνεχιστεί η αποτίμηση της εξωτερικής ερώτησης (προτάσεις *GROUP BY* και *HAVING*, καθώς και των άλλων εξωτερικών ερωτήσεων, αν υπάρχει πολλαπλή ένθεση).

(Ο αλγόριθμος αυτός καλύπτει τη γενική περίπτωση, όπου η εσωτερική ερώτηση μπορεί να περιέχει αναφορές σε πίνακες εξωτερικών ερωτήσεων. Η αντιμετώπιση των περιπτώσεων όπου τέτοιες αναφορές δεν υπάρχουν, καθώς και των υποερωτήσεων εξέτασης ύπαρξης (*EXISTS* και *NOT EXISTS*), είναι απλούστερη.)

Ο ανωτέρω αλγόριθμος, πέραν της πολυπλοκότητας του, έχει το μειονέκτημα ότι η συνεχής εναλλαγή ελέγχου μεταξύ του χρονολογικού φλοιού και του σχεσιακού ΣΔΒΔ, αφαιρεί τη δυνατότητα εφαρμογής οποιασδήποτε γενικής στρατηγικής βελτιστοποίησης του σχεδίου εκτέλεσης της ερώτησης. Επιπρόσθετα, η δημιουργία προσωρινών πινάκων που απαιτείται, αποτελεί *σημείο αναμονής* (holding point) στο σχέδιο εκτέλεσης της ερώτησης, με αποτέλεσμα να μην μπορεί να χρησιμοποιηθεί σωληνωτή επεξεργασία (pipeline). Για τους λόγους αυτούς αποφασίσθηκε η απαγόρευση της χρήσης των προτάσεων *REFORMAT AS* και *NORMALISE* στις υποερωτήσεις, οι οποίες έτσι μπορούν να αποτιμηθούν απ' ευθείας από το Ingres, το οποίο θα χρησιμοποιήσει τους δικούς του μηχανισμούς βελτιστοποίησης.

4) *Περιορισμός των κατηγορημάτων σύγκρισης διαστημάτων που μπορούν να χρησιμοποιηθούν για σχηματισμό υποερωτήσεων*. Το Ingres επιτρέπει την εμφάνιση υποερώτησης μόνο στο δεξιό μέλος ενός τελεστή σύγκρισης, οπότε η πλήρης υποστήριξη περισσοτέρων κατηγορημάτων σύγκρισης διαστημάτων στο σχηματισμό υποερωτήσεων θα είχε ως επακόλουθο τη μετάθεση της αρμοδιότητας εκτέλεσης

ορισμένων υποερωτήσεων στον χρονολογικό φλοιό. Η μετάθεση αυτή δεν είναι επιθυμητή, για τους λόγους που αναφέρθηκαν πιο πάνω.

Αντί για πλήρη υποστήριξη όλων των κατηγορημάτων σύγκρισης διαστημάτων, ο χρονολογικός φλοιός υποστηρίζει *πλήρως* έξι κατηγορήματα σύγκρισης διαστημάτων και *μερικώς* τα υπόλοιπα, μέσω συντακτικών μετασχηματισμών. Για τα κατηγορήματα που υποστηρίζονται πλήρως, ισχύουν οι κανόνες διατύπωσης ερωτήσεων της SQL89, ενώ για τα μερικώς υποστηριζόμενα κατηγορήματα επιβάλλεται ο επιπρόσθετος περιορισμός ότι οι αναφορές σε στήλες οι οποίες περιέχονται στην έκφραση που βρίσκεται στο αριστερό μέλος του κατηγορήματος σύγκρισης διαστημάτων θα πρέπει να μπορούν να χρησιμοποιηθούν και εντός της υποερωτήσεως, χωρίς να προκαλείται σύγχυση. Η διατύπωση των υποερωτήσεων κατά τρόπο ώστε να μη δημιουργείται σύγχυση είναι πάντα δυνατή από πλευράς του χρήστη, με χρήση χαρακτηρισμένων αναφορών στηλών ή/και χρήση ψευδωνύμων σχέσεων. Οι συντακτικοί μετασχηματισμοί που χρησιμοποιούνται περιγράφονται στην παράγραφο 4.3.5.

5) *Περιορισμός στο πλήθος των επεκταμένων ερωτήσεων που μπορούν να συνδυαστούν.* Στην τρέχουσα υλοποίηση, μόνο δύο επεκταμένες ερωτήσεις μπορούν να συνδυαστούν με κάποιον από τους τελεστές *UNION*, *UNION ALL*, *PUNION*, *EXCEPT* και *PEXCEPT* της ΣΑΧΕ. Η υποστήριξη του συνδυασμού περισσοτέρων επεκταμένων ερωτήσεων δεν κρίθηκε απαραίτητη, καθώς σπάνια είναι αναγκαία η διατύπωση τέτοιων ερωτήσεων.

6) *Μη υποστήριξη της συναθροιστικής συνάρτησης countdp.* Για να αποτιμήσουν τις συναθροιστικές συναρτήσεις, τα ΣΔΒΔ εισάγουν ειδικούς κόμβους στο σχέδιο εκτέλεσης της ερώτησης, οι οποίοι υπολογίζουν το αποτέλεσμα της συνάρτησης χρησιμοποιώντας τεχνικές *ένθετων βρόχων* (nested loops), *ταξινόμησης* (sorting) ή *κερματισμού* (hashing) ([Graefe93]). Δεδομένου ότι το εργαλείο *Object Manager* δεν επιτρέπει τον ορισμό τέτοιων κόμβων και ότι ο υπολογισμός του αποτελέσματος της συνάρτησης από το χρονολογικό φλοιό θα απαιτούσε δημιουργία προσωρινών σχέσεων και διαρκή εναλλαγή ελέγχου ανάμεσα στον χρονολογικό φλοιό και τον πυρήνα του Ingres (με τις συνέπειες που έχουν αναφερθεί ανωτέρω), αποφασίστηκε να μην υποστηριχθεί η συνάρτηση *countdp*.

4.3. Ανάλυση των τμημάτων του χρονολογικού ΣΔΒΔ.

Το χρονολογικό ΣΔΒΔ που έχει υλοποιηθεί αποτελείται από *τμήματα* (modules), κάθε ένα από τα οποία είναι επιφορτισμένο με συγκεκριμένες αρμοδιότητες. Στις επόμενες παραγράφους αναλύεται η λειτουργικότητα των τμημάτων του ΣΔΒΔ.

4.3.1. Το επεκταμένο λεξικό δεδομένων.

Όπως έχει ήδη αναφερθεί, η SQL-XE επιτρέπει τη δημιουργία και χρήση πινάκων που έχουν επιπρόσθετα σημασιολογικά χαρακτηριστικά, σε σχέση με τους πίνακες ενός σχεσιακού ΣΔΒΔ. Για να μπορεί το χρονολογικό ΣΔΒΔ να υποστηρίξει τα σημασιολογικά αυτά χαρακτηριστικά, χρησιμοποιεί ένα *επεκταμένο λεξικό δεδομένων*, το οποίο αποτελείται από δύο πίνακες, τους *vtsql_norms* και *vtsql_keys* και το οποίο υπάρχει σε κάθε βάση δεδομένων που διαχειρίζεται το χρονολογικό ΣΔΒΔ. Ο πίνακας *vtsql_norms* χρησιμοποιείται για την αποθήκευση πληροφορίας για τις στήλες κανονικοποίησης των πινάκων και έχει το ακόλουθο σχήμα:

table_name: Το όνομα του κανονικοποιημένου πίνακα.

user_name: Το όνομα του χρήστη στον οποίο ανήκει ο κανονικοποιημένος πίνακας. Οι στήλες *table_name* και *user_name* προσδιορίζουν μοναδικά έναν πίνακα στα πλαίσια μιας βάσης δεδομένων.

column_name: Το όνομα της στήλης κανονικοποίησης.

sequence: Η σειρά με την οποία η στήλη εμφανίζεται στην πρόταση *NORMALISED* της εντολής *CREATE TABLE*. Δεδομένου ότι στο υλοποιηθέν χρονολογικό ΣΔΒΔ υποστηρίζεται μία μόνο διάσταση χρόνου εγκυρότητας, η στήλη αυτή έχει πάντα τιμή 1.

Ο πίνακας *vtsql_keys* χρησιμοποιείται για την αποθήκευση πληροφορίας για το πρωτεύον κλειδί των πινάκων και έχει το ακόλουθο σχήμα:

table_name: Το όνομα του πίνακα στον οποίο έχει ορισθεί το πρωτεύον κλειδί.

user_name: Το όνομα του χρήστη στον οποίο ανήκει ο πίνακας. Οι στήλες *table_name* και *user_name* προσδιορίζουν μοναδικά έναν πίνακα στα πλαίσια μιας βάσης δεδομένων.

column_name: Το όνομα της στήλης που μετέχει στο πρωτεύον κλειδί.

key_type: Η στήλη αυτή έχει τιμή POINT ή INTERVAL, ανάλογα με τον τρόπο με τον οποίο η στήλη που αναφέρεται στο πεδίο *column_name* μετέχει στο πρωτεύον κλειδί.

sequence: Η σειρά με την οποία η στήλη εμφανίζεται στην πρόταση *PRIMARY KEY* της εντολής *CREATE TABLE*.

index_name: Το όνομα μιας δευτερεύουσας δομής δείκτη που ορίζεται πάνω στον πίνακα, προκειμένου να εξασφαλίζεται η μοναδικότητα των κλειδιών τύπου *διάστημα*.

Όταν δημιουργείται ένας πίνακας, στον οποίο ορίζονται στήλες κανονικοποίησης ή/και πρωτεύον κλειδί, το χρονολογικό ΣΔΒΔ εισάγει καταχωρίσεις στους πίνακες *vtsql_norms* και *vtsql_keys*, οι οποίες περιγράφουν τις ιδιότητες αυτές του πίνακα. Για να εκτελεστεί μία εντολή της γλώσσας διαχείρισης δεδομένων, της οποίας η σημασιολογία εξαρτάται από το αν ο πίνακας είναι κανονικοποιημένος ή έχει πρωτεύον κλειδί (*INSERT*, *DELETE*, *UPDATE*), το χρονολογικό ΣΔΒΔ εξετάζει τα περιεχόμενα των πινάκων του επεκταμένου λεξικού, προκειμένου να επιλέξει τον αλγόριθμο που πρέπει να εφαρμοστεί για την εκτέλεση της εντολής. Τέλος, όταν ένας πίνακας διαγράφεται, το χρονολογικό ΣΔΒΔ απομακρύνει τις σχετικές με τον διαγραφόμενο πίνακα καταχωρίσεις από το επεκταμένο λεξικό.

4.3.2. Το υποσύστημα εισόδου.

Το υποσύστημα εισόδου είναι υπεύθυνο για την ανάγνωση των εντολών SQL-XE από τον χρήστη του χρονολογικού ΣΔΒΔ. Το υποσύστημα εισόδου αναγνωρίζει στα δεδομένα που διαβάζει *ακολουθίες διαφυγής* (escape sequences), μέσω των οποίων ο χρήστης ζητά την εκτέλεση των εντολών που έχει εισάγει, ή τον τερματισμό της εκτέλεσης του χρονολογικού φλοιού. Το υποσύστημα εισόδου μεταβιβάζει στον συντονιστή μία ένδειξη, σχετικά με την ακολουθία διαφυγής που αναγνωρίστηκε, προκειμένου αυτός να προβεί στις κατάλληλες ενέργειες.

4.3.3. Ο συντονιστής.

Ο συντονιστής είναι επιφορτισμένος με τις ακόλουθες αρμοδιότητες:

- κατά την έναρξη της εκτέλεσης του χρονολογικού φλοιού, εγκαθιστά τις απαραίτητες συνδέσεις με το σχεσιακό ΣΔΒΔ και δεσμεύει μνήμη για τις δομές δεδομένων σταθερού μεγέθους. Η μνήμη για τις δομές δεδομένων μεταβλητού μεγέθους δεσμεύεται και απελευθερώνεται από τα τμήματα του χρονολογικού φλοιού που τις χρησιμοποιούν.
- καλεί το υποσύστημα εισόδου και, όταν ανακτά τον έλεγχο, αποφασίζει για τις ενέργειες στις οποίες πρέπει να προβεί, βασιζόμενος στην πληροφορία που λαμβάνει από το υποσύστημα εισόδου. Αν ο χρήστης έχει ζητήσει την εκτέλεση των εντολών που έχει εισάγει, δίνοντας την κατάλληλη ακολουθία διαφυγής, ο συντονιστής καλεί το

συντακτικό αναλυτή, ο οποίος θα επεξεργαστεί και θα προωθήσει προς εκτέλεση τις εντολές. Στη συνέχεια, καλείται ξανά το υποσύστημα εισόδου για να διαβαστεί η επόμενη δέσμη εντολών.

- όταν ο χρήστης ζητήσει τον τερματισμό της εκτέλεσης του χρονολογικού φλοιού, εισάγοντας την κατάλληλη ακολουθία διαφυγής, ο συντονιστής τερματίζει τις συνδέσεις με το σχεσιακό ΣΔΒΔ, απελευθερώνει τη δεσμευμένη μνήμη και προκαλεί τον τερματισμό του προγράμματος.

4.3.4. Ο επεκταμένος πυρήνας του Ingres.

Όπως έχει αναφερθεί, ο πυρήνας του ΣΔΒΔ Ingres έχει επεκταθεί, με την ενσωμάτωση σ' αυτόν των απαραίτητων τύπων δεδομένων (*χρονικό σημείο* και *χρονικό διάστημα*), των συναρτήσεων που δέχονται ορίσματα ή επιστρέφουν αποτελέσματα αυτών των τύπων, καθώς και των κατηγορημάτων σύγκρισης διαστημάτων. Οι επεκτάσεις αυτές έχουν υλοποιηθεί χρησιμοποιώντας το εργαλείο *Object Manager* του Ingres, το οποίο επιτρέπει τον ορισμό νέων τύπων δεδομένων, καθώς και συναρτήσεων και την ενσωμάτωσή τους στον πυρήνα. Ο κώδικας που είναι υπεύθυνος για τη διαχείριση των νέων τύπων δεδομένων και υλοποιεί τις συναρτήσεις, γράφεται σε C και μεταγλωττίζεται, παράγοντας καταληκτικά αρχεία (object files), τα οποία συνδέονται στον πυρήνα του Ingres.

Η υλοποίηση των νέων τύπων δεδομένων και των συναρτήσεων έγινε στα πλαίσια του προγράμματος ORES. Για το πρόγραμμα αυτό, υλοποιήθηκε μόνο ένας τύπος χρονικού διαστήματος, με διακριτότητα ημέρας, με το όνομα DATEINTERVAL, ενώ για την αναπαράσταση χρονικών σημείων χρησιμοποιήθηκε ο τύπος DATE, ο οποίος υποστηρίζεται άμεσα από το Ingres. Η εσωτερική αναπαράσταση των δεδομένων τύπου DATEINTERVAL (δηλ. η μορφή με την οποία τα δεδομένα τύπου DATEINTERVAL αποθηκεύονται στη βάση δεδομένων) είναι ένα διατεταγμένο ζεύγος ακεραίων *(αρχή, πέρας)*, όπου *αρχή* < *πέρας*. Οι ακέραιοι αυτοί αντιστοιχούν στις χρονικές αποστάσεις (εκφρασμένες σε δευτερόλεπτα) των αντιστοίχων ημερομηνιών από την 1η Ιανουαρίου του 1970. Η προσέγγιση αυτή παρέχει τα ακόλουθα πλεονεκτήματα:

- ο απαιτούμενος χώρος αποθήκευσης είναι μικρός (μόνο 8 bytes απαιτούνται για την αποθήκευση κάθε δεδομένου).
- οι συγκρίσεις διαστημάτων ανάγονται σε συγκρίσεις ακεραίων, κάθε μία από τις οποίες εκτελείται συνήθως με μία μόνο εντολή γλώσσας μηχανής.

- είναι συμβατή με τον τρόπο αναπαράστασης των ημερομηνιών που χρησιμοποιούν τα συστήματα UNIX, γεγονός που καθιστά διαθέσιμο το σύνολο των βιβλιοθηκών που σχετίζονται με τη διαχείριση ημερομηνιών, χωρίς να είναι απαραίτητες μετατροπές.

Το μόνο μειονέκτημα της προσέγγισης αυτής είναι ότι δεν χρησιμοποιούνται όλες οι δυνατές τιμές ακεραίων για την αναπαράσταση ημερομηνιών, π.χ. ο ακέραιος *43200* που αντιστοιχεί στη χρονική στιγμή *1970-01-01 12:00* δεν χρησιμοποιείται, διότι η χρονική στιγμή που αναπαριστά περιλαμβάνεται στην ημέρα *1970-01-01*, για την αναπαράσταση της οποίας χρησιμοποιείται πάντα ο ακέραιος *0*. Το γεγονός ότι δεν χρησιμοποιούνται όλες οι δυνατές τιμές μειώνει το εύρος των ημερομηνιών που υποστηρίζονται.

Μία εναλλακτική προσέγγιση θα ήταν να αναπαρίσταται εσωτερικά ο τύπος DATEINTERVAL ως ένα ζεύγος ημερομηνιών (χ_1, χ_2), δηλαδή τα χ_1 και χ_2 να είναι τύπου DATE. Η εναλλακτική όμως αυτή λύση δεν μπορούσε να εφαρμοστεί, διότι το Ingres χαρακτηρίζει τον τύπο DATE *σύνθετο* (complex) και το εργαλείο *Object Manager* δεν επιτρέπει τη χρήση συνθέτων τύπων στον ορισμό νέων τύπων.

Η εξωτερική αναπαράσταση των δεδομένων τύπου DATEINTERVAL (δηλαδή η μορφή με την οποία τα δεδομένα τύπου DATEINTERVAL παρουσιάζονται στον χρήστη του Ingres) είναι μία συμβολοσειρά της μορφής $[\eta_1, \eta_2)$ όπου η_1 και η_2 είναι οι ημερομηνίες που αντιστοιχούν στους ακέραιους *start* και *stop*, αντίστοιχα. Η μετατροπή ανάμεσα στην εξωτερική και την εσωτερική αναπαράσταση γίνεται από τη συνάρτηση *usdi_convert()*, η οποία αποτελεί τμήμα του κώδικα υποστήριξης του τύπου DATEINTERVAL και καλείται αυτόματα από τον πυρήνα του Ingres, όταν αυτό είναι απαραίτητο. Όταν η συνάρτηση *usdi_convert()* καλείται για να μετατρέψει την εξωτερική αναπαράσταση σε εσωτερική, επιτελεί εκτεταμένους ελέγχους για να πιστοποιήσει ότι η συμβολοσειρά ξεκινά με το χαρακτήρα *[*, τελειώνει με το χαρακτήρα *)* και περιέχει δύο ημερομηνίες η_1 και η_2 οι οποίες διαχωρίζονται με κόμμα και πληρούν τη συνθήκη $\eta_1 < \eta_2$.

Εκτός από την απαγόρευση χρήσης ενός σύνθετου τύπου δεδομένων για τον ορισμό ενός νέου τύπου, το εργαλείο *Object Manager* επιβάλλει τους ακόλουθους περιορισμούς, σχετικά με τον ορισμό επεκτάσεων, οι οποίοι επηρέασαν την πορεία της υλοποίησης:

- δεν επιτρέπεται ο ορισμός συναρτήσεων οι οποίες δέχονται ορίσματα ή επιστρέφουν αποτελέσματα των οποίων ο τύπος χαρακτηρίζεται *σύνθετος*. Όμως, στα πλαίσια της SQL-XE ορίζονται συναρτήσεις που δέχονται ορίσματα ή/και επιστρέφουν αποτελέσματα τύπου *χρονικό σημείο* και, όπως έχει αναφερθεί, για την αναπαράσταση

των χρονικών σημείων διακριτότητας ημέρας χρησιμοποιείται ο τύπος *DATE*. Για να ξεπεραστεί αυτό το πρόβλημα, στο επίπεδο του επεκταμένου πυρήνα, τα ορίσματα και τα αποτελέσματα των συναρτήσεων αυτών έχουν δηλωθεί ως *τύπου συμβολοσειράς* (ο οποίος θεωρείται βασικός τύπος και μπορεί να χρησιμοποιηθεί) και άλλα τμήματα του χρονολογικού φλοιού φροντίζουν να επιτελούν τις απαραίτητες μετατροπές τύπων.

- δεν επιτρέπεται ο ορισμός κατηγορημάτων. Για να υλοποιηθούν τα κατηγορήματα σύγκρισης διαστημάτων που ορίζονται στην SQL-XE, ορίσθηκαν στο επίπεδο αυτό ομώνυμες συναρτήσεις, οι οποίες δέχονται ως ορίσματα δύο χρονικά διαστήματα και επιστρέφουν 1 ή 0, ανάλογα με το αν το αντίστοιχο κατηγόρημα αληθεύει ή όχι. Ο *συντακτικός αναλυτής* αναλαμβάνει την αρμοδιότητα να μετατρέψει την ενδοθεματική (infix) γραφή των κατηγορημάτων, που χρησιμοποιείται από το χρήστη, σε κλήση της κατάλληλης συνάρτησης και έλεγχο του αποτελέσματός της, μορφή που είναι παραδεκτή από τον επεκταμένο πυρήνα του Ingres.
- δεν επιτρέπεται ο ορισμός συναρτήσεων οι οποίες δέχονται περισσότερα από δύο ορίσματα. Οι συναρτήσεις *window* και *windowno*, οι οποίες δέχονται τρία ορίσματα, ορίζονται στο επίπεδο αυτό ως συναρτήσεις που δέχονται ένα όρισμα τύπου *συμβολοσειρά*, το οποίο περιέχει την αναπαράσταση των ορισμάτων σε μορφή κειμένου. Ο κώδικας που υλοποιεί αυτές τις συναρτήσεις, εξάγει από τη συμβολοσειρά τις τιμές των ορισμάτων και στη συνέχεια υπολογίζει το αποτέλεσμα, το οποίο και επιστρέφεται. Η διαδικασία διευθέτησης των ορισμάτων σε μία συμβολοσειρά και της εξαγωγής από αυτή των τιμών των ορισμάτων είναι αόρατη στο χρήστη.

Τέλος, στο επίπεδο αυτό ορίζεται η σημασιολογία των τελεστών σύγκρισης =, <>, <, <=, > και >=, όταν τα ορίσματά τους είναι τύπου DATEINTERVAL. Οι τελεστές = και <> χρησιμοποιούνται για έλεγχο ισότητας και ανισότητας, αντίστοιχα, ενώ οι τελεστές <, <=, > και >= ορίζονται ως συνώνυμα των κατηγορημάτων *precedes*, *prequals*, *follows* και *folequals*. Ο ορισμός συνωνύμων γίνεται τόσο για διευκόλυνση των χρηστών όσο και για να χρησιμοποιηθεί στο σχηματισμό υποερωτήσεων, όπως περιγράφεται στη συνέχεια.

4.3.5. Ο συντακτικός αναλυτής.

Ο συντακτικός αναλυτής είναι υπεύθυνος για τον συντακτικό έλεγχο και ανάλυση των εντολών SQL-XE που εισάγονται από τον χρήστη, την εφαρμογή συντακτικών μετασχηματισμών, τη διαμόρφωση δομών δεδομένων που περιγράφουν τις ερωτήσεις και

την κλήση των κατάλληλων διαδικασιών εκτέλεσης. Στις ακόλουθες παραγράφους περιγράφεται η λειτουργικότητα του συντακτικού αναλυτή.

I. Συντακτικός έλεγχος.

Ο συντακτικός έλεγχος χωρίζεται σε δύο επίπεδα. Το χαμηλότερο επίπεδο είναι η *λεκτική ανάλυση*, η οποία μετατρέπει την ακολουθία χαρακτήρων εισόδου σε μία ακολουθία *λεκτικών συμβόλων* (lexical tokens). Ένα λεκτικό σύμβολο μπορεί να είναι μία δεσμευμένη λέξη (π.χ. *UPDATE*), ένας προσδιοριστής (π.χ. *department*), μία συμβολοσειρά (π.χ. *'Πλάτων'*), ένα σημείο στίξης (π.χ. κόμμα) ή οποιοδήποτε στοιχείο της SQL-XE. Η ακολουθία λεκτικών συμβόλων προωθείται στο ανώτερο επίπεδο, στον *ελεγκτή συντακτικού*, ο οποίος ελέγχει αν η σειρά με την οποία εμφανίζονται τα λεκτικά σύμβολα είναι σύμφωνη με τον ορισμό της γλώσσας SQL-XE. Αν διαπιστωθεί η ύπαρξη σφαλμάτων, σε οποιοδήποτε από τα δύο επίπεδα, δηλαδή η ακολουθία χαρακτήρων που δόθηκε από τον χρήστη περιέχει χαρακτήρες των οποίων η χρήση δεν επιτρέπεται στην SQL-XE ή η ακολουθία των λεκτικών συμβόλων δεν σχηματίζει μία εντολή SQL-XE, τότε ο συντακτικός αναλυτής εμφανίζει ένα κατάλληλο μήνυμα λάθους.

Ο λεκτικός αναλυτής έχει υλοποιηθεί χρησιμοποιώντας το εργαλείο *lex* του UNIX, αλλά ένα σημαντικό τμήμα της αναγνώρισης λεκτικών συμβόλων έχει κωδικοποιηθεί σε C, προκειμένου να αποφευχθεί η δημιουργία μεγάλων πινάκων μετάβασης καταστάσεων (state transition tables). Ο συντακτικός αναλυτής έχει κωδικοποιηθεί χρησιμοποιώντας το εργαλείο *yacc* του UNIX.

II. Συντακτικοί μετασχηματισμοί.

Ο συντακτικός αναλυτής εφαρμόζει μετασχηματισμούς στην είσοδο που διαβάζεται από τον χρήστη, προκειμένου να της δώσει μορφή συμβατή με τους κανόνες συντακτικού του επεκταμένου πυρήνα του Ingres. Πιο συγκεκριμένα εφαρμόζονται οι εξής μετασχηματισμοί:

1) *μετασχηματισμός των κατηγορημάτων σύγκρισης διαστημάτων σε κλήσεις συναρτήσεων*: συνθήκες της μορφής

έκφραση1 κατηγόρημα έκφραση2

(όπου κατηγόρημα είναι ένα από τα κατηγορήματα σύγκρισης διαστημάτων) μετασχηματίζονται στη μορφή

(*κατηγόρημα*(*έκφραση1*, *έκφραση2*) = 1)

η οποία είναι αποδεκτή από τον επεκταμένο πυρήνα του Ingres. Ο συντακτικός αναλυτής αναγνωρίζει επίσης τις συνθήκες της μορφής

έκφραση1 NOT *κατηγόρημα έκφραση2*

τις οποίες μετατρέπει στη μορφή

(*κατηγόρημα*(*έκφραση1*, *έκφραση2*) = 0)

Οι συνθήκες της μορφής

NOT *έκφραση1 κατηγόρημα έκφραση2*

δεν απαιτούν ιδιαίτερη αντιμετώπιση, μια και θα μετασχηματιστούν βάσει του πρώτου κανόνα στη μορφή

NOT (*κατηγόρημα*(*έκφραση1*, *έκφραση2*) = 1)

2) *μετασχηματισμός συνθηκών που περιλαμβάνουν κατηγορήματα σύγκρισης διαστημάτων και υποερωτήσεις*: ο χρονολογικός φλοιός υποστηρίζει πλήρως το σχηματισμό υποερωτήσεων με τα κατηγορήματα *precedes*, *prequals*, *equals*, *folequals* και *follows* και μερικώς το σχηματισμό υποερωτήσεων με τα υπόλοιπα κατηγορήματα σύγκρισης διαστημάτων. Οι συνθήκες της μορφής

έκφραση κατηγόρημα υποερώτηση

(όπου κατηγόρημα είναι ένα από τα πλήρως υποστηριζόμενα κατηγορήματα σύγκρισης διαστημάτων) μετασχηματίζονται στη μορφή

έκφραση τελεστής_σύγκρισης υποερώτηση

Ο *τελεστής_σύγκρισης* στη μετασχηματισμένη συνθήκη θα είναι ένας από τους <, <=, =, >= και >, ανάλογα με το αν το *κατηγόρημα* στην αρχική μορφή της συνθήκης, είναι το *precedes*, *prequals*, *equals*, *folequals* ή *follows*, αντίστοιχα. Επίσης, υποερωτήσεις της μορφής

έκφραση NOT *κατηγόρημα υποερώτηση*

(όπου κατηγόρημα είναι ένα από τα πλήρως υποστηριζόμενα κατηγορήματα σύγκρισης διαστημάτων) μετασχηματίζονται στη μορφή

NOT (*έκφραση τελεστής_σύγκρισης υποερώτηση*)

Ο *τελεστής σύγκρισης* που θα χρησιμοποιηθεί επιλέγεται βάσει του αλγόριθμου αντικατάστασης κατηγορημάτων που περιγράφηκε πιο πάνω. Και στις δύο περιπτώσεις, η υποερώτηση μπορεί να περιέχει τους ποσοδείκτες *ALL* και *ANY*.

Οι υποερωτήσεις που σχηματίζεται με *μερικώς υποστηριζόμενα* κατηγορήματα σύγκρισης διαστημάτων και έχουν τη μορφή

έκφραση1 [NOT] *κατηγόρημα* [ANY] (SELECT *έκφραση2*

FROM *λίστα_σχέσεων_και_ψευδωνύμων*

WHERE *συνθήκη*)

(δηλαδή δεν περιέχουν τον ποσοδείκτη *ALL*), μετασχηματίζονται στη μορφή

[NOT] EXISTS (SELECT *έκφραση2*

FROM *λίστα_σχέσεων_και_ψευδωνύμων*

WHERE (*συνθήκη*)

AND *κατηγόρημα*(*έκφραση1*, *έκφραση2*) = 1)

Ο τελεστής *NOT* χρησιμοποιείται στη μετασχηματισμένη μορφή ανάλογα με το αν είναι παρών πριν από το *κατηγόρημα* της αρχικής μορφής ή όχι, αντίστοιχα.

Επίσης, οι υποερωτήσεις που έχουν τη μορφή

έκφραση1 [NOT] *κατηγόρημα* ALL (SELECT *έκφραση2*

FROM *λίστα_σχέσεων_και_ψευδωνύμων*

WHERE *συνθήκη*)

μετασχηματίζονται στη μορφή

[NOT] EXISTS (SELECT *έκφραση2*

FROM *λίστα_σχέσεων_και_ψευδωνύμων*

WHERE (*συνθήκη*)

AND *κατηγόρημα*(*έκφραση1*, *έκφραση2*) = 0)

Ο τελεστής *NOT* είναι παρών στη μετασχηματισμένη μορφή αν *δεν* είναι παρών στην αρχική μορφή της ερώτησης και αντίστροφα.

3) *εισαγωγή μετατροπών τύπων στις συναρτήσεις που δέχονται ορίσματα ή επιστρέφουν αποτελέσματα τύπου DATE*: δεδομένου ότι ο τύπος δεδομένων *DATE* χαρακτηρίζεται από το Ingres ως *σύνθετος*, δεδομένα αυτού του τύπου δεν μπορούν να χρησιμοποιηθούν ως ορίσματα ή αποτελέσματα συναρτήσεων που ορίζονται μέσω του εργαλείου *Object Manager*. Τα ορίσματα και τα αποτελέσματα συναρτήσεων της SQL-XE που πρέπει να είναι τύπου *DATE*, έχουν δηλωθεί ως τύπου *συμβολοσειρά* στο επίπεδο του επεκταμένου πυρήνα του Ingres και ο συντακτικός αναλυτής φροντίζει για την εισαγωγή *μετατροπών τύπων*, ώστε:

   i) τα ορίσματα τύπου *DATE* να μετατρέπονται στην ισοδύναμη αναπαράστασή τους στον τύπο *συμβολοσειρά*, πριν μεταβιβαστούν στις συναρτήσεις. Η μετατροπή τύπου γίνεται χρησιμοποιώντας τη συνάρτηση *C* του Ingres.

ii) τα αποτελέσματα που, σύμφωνα με τις προδιαγραφές της SQL-XE, πρέπει να είναι τύπου *DATE* να μετατρέπονται στον τύπο αυτό. Η μετατροπή γίνεται χρησιμοποιώντας τη συνάρτηση *DATE* του Ingres.

Σύμφωνα με τα παραπάνω, η κλήση της συνάρτησης

span(η1, η2)

της οποίας τα ορίσματα είναι τύπου *DATE* μετασχηματίζεται σε

span(c(η1), c(η2))

ενώ η κλήση της συνάρτησης

middle(δ1)

η οποία επιστρέφει αποτέλεσμα τύπου *DATE* μετασχηματίζεται σε

date(middle(δ1))

4) *μετασχηματισμός των ορισμάτων για τις συναρτήσεις που δέχονται περισσότερα από τρία ορίσματα*: οι συναρτήσεις *window* και *windowno* δέχονται τρία ορίσματα η κάθε μία, αλλά ο πυρήνας του Ingres απαγορεύει τον ορισμό συναρτήσεων με περισσότερα από δύο ορίσματα. Για να ξεπεραστεί αυτό το πρόβλημα, οι δύο συναρτήσεις έχουν δηλωθεί στο επίπεδο του πυρήνα ότι δέχονται ένα μόνο όρισμα τύπου *συμβολοσειρά*, το οποίο περιέχει τις τιμές και των τριών πραγματικών ορισμάτων, διαχωρισμένες με κόμμα. Ο συντακτικός αναλυτής φροντίζει για τη δημιουργία της κατάλληλης συμβολοσειράς, η οποία θα μεταβιβαστεί ως όρισμα στις δύο αυτές συναρτήσεις, μετατρέποντας τις κλήσεις της μορφής

window(όρισμα1, όρισμα2, όρισμα3)

στη μορφή

window(c(όρισμα1) + ',' + c(όρισμα2) + ',' + c(όρισμα3))

(ο αντίστοιχος μετασχηματισμός εφαρμόζεται και για τη συνάρτηση *windowno*). Ο μετασχηματισμός που χρησιμοποιείται αναγκάζει τον πυρήνα του Ingres να αποτιμήσει τις παραστάσεις που χρησιμοποιούνται ως ορίσματα, να μετατρέψει το αποτέλεσμά της κάθε μίας στην ισοδύναμη αναπαράσταση σε μορφή κειμένου και τέλος να συνενώσει τις τρεις συμβολοσειρές που θα προκύψουν, διαχωρίζοντάς τες με κόμματα. Το τελικό αποτέλεσμα είναι αυτό που θα αποτελέσει το όρισμα της συνάρτησης *window* (ή *windowno*) που βρίσκεται στον πυρήνα του Ingres. Η εναλλακτική λύση, να μετασχηματίζεται η κλήση των δύο συναρτήσεων στη μορφή

window('όρισμα1, όρισμα2, όρισμα3')

δεν θα είχε το επιθυμητό αποτέλεσμα, διότι δεν θα υπολογιζόταν οι τιμές των αναφορών σε στήλες και των κλήσεων συναρτήσεων που θα εμφανιζόταν στα ορίσματα της συνάρτησης *window* (ή *windowno*), πριν τα ορίσματα μεταβιβαστούν στη συνάρτηση.

5) *μετασχηματισμός της συναθροιστικής συνάρτησης countap*: η κλήση της συνάρτησης

countap(όρισμα)

μετασχηματίζεται στη μορφή

sum(dur(όρισμα))

Δεδομένου ότι η μετασχηματισμένη μορφή περιέχει τη συναθροιστική *sum*, οι σημασιολογικοί κανόνες που διέπουν τη χρήση των συναθροιστικών συναρτήσεων εφαρμόζονται αυτόματα από τον πυρήνα του Ingres, όπως άλλωστε θα έπρεπε, μια και η ίδια η συνάρτηση *countap* είναι συναθροιστική.

III. Διαμόρφωση δομών δεδομένων.

Ο συντακτικός αναλυτής, εκτός από τον έλεγχο συντακτικής ορθότητας των εντολών που εισάγονται από τον χρήστη, φροντίζει για τη διαμόρφωση κατάλληλων δομών δεδομένων, οι οποίες θα επιτρέψουν στο υποσύστημα εκτέλεσης να δημιουργήσει το σχέδιο εκτέλεσης για κάθε ερώτηση και να προβεί στις απαιτούμενες ενέργειες. Οι δομές δεδομένων που σχηματίζονται εξαρτώνται από την εντολή που αναλύεται και παρουσιάζονται στις επόμενες παραγράφους.

i) Η εντολή CREATE TABLE.

Η δομή δεδομένων που χρησιμοποιείται για την περιγραφή μιας εντολής *CREATE TABLE* αποτελείται από τα ακόλουθα πεδία:

1. *όνομα_πίνακα*, που περιέχει το όνομα του υπό δημιουργία πίνακα.
2. *στήλες_πίνακα*. Το πεδίο αυτό έχει τη μορφή λίστας, κάθε κόμβος της οποίας είναι μία τριπλέτα *(όνομα_στήλης, τύπος_στήλης, ιδιότητα)*. Η λίστα περιέχει μία τριπλέτα για κάθε στήλη που δηλώνεται ότι ανήκει στο σχήμα του πίνακα. Τα πεδία *όνομα_στήλης* και *τύπος_στήλης* κάθε τριπλέτας προσδιορίζουν το όνομα και τον τύπο της στήλης, αντίστοιχα, ενώ το πεδίο *ιδιότητα* μπορεί είτε να είναι κενό, είτε να έχει την τιμή *NOT NULL*, ανάλογα με το αν ο χρήστης ορίζει ότι η στήλη επιτρέπεται να δέχεται τιμές *NULL* ή όχι.
3. *στήλες_κανονικοποίησης*. Το πεδίο αυτό είναι μία λίστα, η οποία περιέχει ονόματα στηλών που αναφέρονται στην πρόταση *NORMALISED* της εντολής *CREATE TABLE*. Αν η πρόταση *NORMALISED* δεν είναι παρούσα, η λίστα είναι κενή.

4. *πρωτεύον_κλειδί*. Το πεδίο αυτό έχει τη μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(όνομα_στήλης, τύπος_κλειδιού)*. Η λίστα περιέχει ένα ζεύγος για κάθε στήλη που αναφέρεται στην πρόταση *PRIMARY KEY* της πρότασης *CREATE TABLE*. Το πεδίο *όνομα_στήλης* περιέχει το όνομα της στήλης, ενώ το πεδίο *τύπος_κλειδιού* μπορεί είτε να είναι κενό, ή να έχει μία από τις τιμές *POINT* και *INTERVAL*, ανάλογα με τα προσδιοριζόμενα στην πρόταση *PRIMARY KEY*. Αν η πρόταση *PRIMARY KEY* δεν είναι παρούσα, η λίστα είναι κενή.

ii) Η εντολή DROP TABLE.

Η δομή δεδομένων που περιγράφει την εντολή *DROP TABLE* αποτελείται από ένα μόνο πεδίο, το *όνομα_πίνακα*, το οποίο περιέχει το όνομα του υπό διαγραφή πίνακα.

iii) Η εντολή SELECT.

Η δομή δεδομένων που περιγράφει την εντολή *SELECT* αποτελείται από τα ακόλουθα πεδία:

1. *πρώτη_ερώτηση*, το οποίο είναι μία δομή δεδομένων που περιγράφει την πρώτη επεκταμένη ερώτηση της εντολής *SELECT*. Η δομή που χρησιμοποιείται για την περιγραφή των επεκταμένων ερωτήσεων περιγράφεται στη συνέχεια.
2. *είδος_συνδυασμού*. Το πεδίο αυτό μπορεί να είναι κενό, αν δεν υπάρχει δεύτερη ερώτηση, ή να έχει μία από τις τιμές *UNION*, *UNION ALL* και *EXCEPT*, ανάλογα με την πράξη της ΣΑΧΕ που πρέπει να χρησιμοποιηθεί για να συνδυαστεί το αποτέλεσμα της πρώτης ερώτησης με το αποτέλεσμα της δεύτερης.
3. *στήλες_συνδυασμού*. Το πεδίο αυτό είναι μία λίστα από προσδιορισμούς στηλών αποτελέσματος, που παρατίθενται μετά τον τελεστή συνδυασμού. Αν δεν παρατίθενται στήλες, τότε η λίστα είναι κενή, ενώ αν το πεδίο *είδος_συνδυασμού* είναι κενό, τότε η τιμή του πεδίου *στήλες_συνδυασμού* αγνοείται.
4. *δεύτερη_ερώτηση*, το οποίο είναι μία δομή δεδομένων που περιγράφει την δεύτερη επεκταμένη ερώτηση της εντολής *SELECT*. Αν το πεδίο *είδος_συνδυασμού* είναι κενό, τότε η τιμή του πεδίου *δεύτερη_ερώτηση* αγνοείται.
5. *στήλες_ταξινόμησης*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(στήλη_αποτελέσματος, διάταξη)*. Η λίστα περιέχει ένα ζεύγος για κάθε προσδιορισμό ταξινόμησης που παρατίθεται στην πρόταση *ORDER BY*. Το πεδίο *στήλη_αποτελέσματος* περιέχει τον προσδιορισμό στήλης αποτελέσματος, ενώ το πεδίο *διάταξη* είτε είναι κενό, είτε έχει μία από τις τιμές *ASC* και *DESC*. Αν η πρόταση *ORDER BY* δεν είναι παρούσα, τότε η λίστα είναι κενή.

Η δομή που χρησιμοποιείται για να περιγραφεί μία επεκταμένη ερώτηση περιέχει τα ακόλουθα πεδία:

1. *ποσοδείκτης*. Το πεδίο αυτό είτε είναι κενό, είτε έχει μία από τις τιμές *ALL* και *DISTINCT*, αν κάποια από αυτές τις δεσμευμένες λέξεις χρησιμοποιείται στην επεκταμένη ερώτηση.
2. *στήλες_αποτελέσματος*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(όνομα_στήλης, ορισμός_στήλης)*, όπου *ορισμός_στήλης* είναι η έκφραση που θα αποτιμηθεί για να παραχθεί η τιμή της στήλης για κάθε πλειάδα του αποτελέσματος, μετασχηματισμένη σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Το υποπεδίο *όνομα_στήλης* περιέχει το όνομα που δίδεται στην στήλη αποτελέσματος μέσω του συμβολισμού

όνομα_στήλης = ορισμός_στήλης

ή του συμβολισμού

ορισμός_στήλης = όνομα_στήλης

Αν δεν χρησιμοποιείται κανένας από τους δύο συμβολισμούς, αλλά παρατίθεται μόνο ο ορισμός της στήλης, το υποπεδίο *όνομα_στήλης* θα είναι κενό.

3. *πίνακες*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(όνομα_πίνακα, ψευδώνυμο)*. Η λίστα περιέχει έναν κόμβο για κάθε σχέση ή ψευδώνυμο που παρατίθεται στην πρόταση *FROM*. Το πεδίο *όνομα_πίνακα* περιέχει το πραγματικό όνομα του πίνακα, ενώ το πεδίο *ψευδώνυμο* περιέχει το ψευδώνυμο που αποδίδεται στον πίνακα μέσω της σύνταξης

όνομα_πίνακα ψευδώνυμο

Αν δεν χρησιμοποιείται αυτή η σύνταξη, αλλά παρατίθεται μόνο το πραγματικό όνομα του πίνακα, τότε το πεδίο *ψευδώνυμο* θα είναι κενό.

4. *συνθήκη*, που περιέχει τη συνθήκη της πρότασης *WHERE*, μετασχηματισμένη σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *WHERE* δεν είναι παρούσα, τότε το πεδίο είναι κενό.
5. *προσδιορισμός_ομαδοποίησης*. Το πεδίο αυτό περιέχει τις προτάσεις *GROUP BY* και *HAVING*. Η συνθήκη της πρότασης *HAVING* είναι μετασχηματισμένη σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν καμία από τις προτάσεις *GROUP BY* και *HAVING* δεν είναι παρούσα, το πεδίο είναι κενό.
6. *προσδιορισμοί_αναμόρφωσης*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(πράξη, στήλες)*. Η λίστα περιέχει έναν κόμβο για κάθε υποπρόταση

FOLD, *UNFOLD* και *UNFOLD ALL* της πρότασης *REFORMAT AS*. Το πεδίο *πράξη* υποδεικνύει ποια από τις πράξεις της ΣΑΧΕ θα εφαρμοστεί, ενώ το πεδίο *στήλες* είναι μία λίστα που περιέχει τις στήλες πάνω στις οποίες θα εφαρμοστεί η πράξη. Αν η πρόταση *REFORMAT AS* δεν είναι παρούσα, τότε το πεδίο *προσδιορισμοί_αναμόρφωσης* θα έχει ως τιμή το κενό.

7. *προσδιορισμοί_κανονικοποίησης*. Το πεδίο αυτό είναι μία λίστα που περιέχει τις στήλες αποτελέσματος που αναφέρονται στην πρόταση *NORMALISE ON*. Αν η πρόταση *NORMALISE ON* δεν είναι παρούσα, τότε η λίστα είναι κενή.

iv) Η εντολή INSERT.

Η δομή δεδομένων που περιγράφει την εντολή *INSERT* αποτελείται από τα ακόλουθα πεδία:

1. *όνομα_πίνακα*, που περιέχει το όνομα του πίνακα από τον οποίο θα εισαχθούν δεδομένα.
2. *στήλες_εισαγωγής*. Το πεδίο αυτό είναι μία λίστα, που περιέχει τα ονόματα των στηλών που παρατίθενται ανάμεσα στις παρενθέσεις που ακολουθούν το όνομα του πίνακα στη σύνταξη της εντολής *INSERT*. Αν δεν παρατίθενται ονόματα στηλών, τότε η λίστα είναι κενή.
3. *εισαγόμενες_τιμές*. Το πεδίο αυτό είναι μία λίστα που περιέχει τις εκφράσεις που παρατίθενται στην πρόταση *VALUES*, μετασχηματισμένες σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *VALUES* δεν είναι παρούσα, αλλά οι προς εισαγωγή τιμές προσδιορίζονται μέσω ερώτησης, τότε η λίστα είναι κενή.
4. *ερώτηση*. Το πεδίο αυτό είναι μία δομή δεδομένων περιγραφής επεκταμένης ερώτησης, η οποία περιγράφει την ερώτηση που θα παράγει τα δεδομένα που θα εισαχθούν στον πίνακα. Αν οι προς εισαγωγή τιμές προσδιορίζονται μέσω της πρότασης *VALUES* (στην οποία περίπτωση η τιμή του πεδίου *εισαγόμενες_τιμές* δεν θα είναι κενή), τότε η τιμή του πεδίου αγνοείται.

v) Η εντολή DELETE.

Η δομή δεδομένων που περιγράφει την εντολή *DELETE* αποτελείται από τα ακόλουθα πεδία:

1. *όνομα_πίνακα*, που περιέχει το όνομα του πίνακα από τον οποίο θα διαγραφούν δεδομένα.
2. *προσδιορισμός_τμημάτων*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(στήλη, έκφραση)*. Η λίστα περιέχει έναν κόμβο για κάθε εκχώρηση της μορφής *στήλη = έκφραση* που περιέχεται στην πρόταση *PORTION*. Το πεδίο *έκφραση*

είναι μετασχηματισμένο, σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *PORTION* δεν είναι παρούσα, τότε η λίστα είναι κενή.

3. *συνθήκη*, που περιέχει τη συνθήκη της πρότασης *WHERE*, μετασχηματισμένη σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *WHERE* δεν είναι παρούσα, τότε το πεδίο είναι κενό.

vi) Η εντολή UPDATE.

Η δομή δεδομένων που περιγράφει την εντολή *UPDATE* αποτελείται από τα ακόλουθα πεδία:

1. *όνομα_πίνακα*, το οποίο περιέχει το όνομα του πίνακα που θα ενημερωθεί.
2. *προσδιορισμός_τμημάτων*. Το πεδίο αυτό έχει τη μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(στήλη, έκφραση)*. Η λίστα περιέχει έναν κόμβο για κάθε εκχώρηση της μορφής *στήλη = έκφραση* που περιέχεται στην πρόταση *PORTION*. Το πεδίο *έκφραση* είναι μετασχηματισμένο, σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *PORTION* δεν είναι παρούσα, τότε η λίστα είναι κενή.
3. *εκχωρήσεις*. Το πεδίο αυτό έχει μορφή λίστας, κάθε κόμβος της οποίας είναι ένα ζεύγος *(στήλη, έκφραση)*. Η λίστα περιέχει έναν κόμβο για κάθε εκχώρηση της μορφής *στήλη = έκφραση* που περιέχεται στην πρόταση *SET*. Το πεδίο *έκφραση* είναι μετασχηματισμένο, σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5.
4. *συνθήκη*, που περιέχει τη συνθήκη της πρότασης *WHERE*, μετασχηματισμένη σύμφωνα με όσα αναφέρονται στην παράγραφο 4.3.5. Αν η πρόταση *WHERE* δεν είναι παρούσα, τότε το πεδίο είναι κενό.

vii) Η εντολή CREATE INDEX.

Η δομή δεδομένων που περιγράφει την εντολή *CREATE INDEX* αποτελείται από τα ακόλουθα πεδία:

1. *όνομα_δείκτη*, που περιέχει το όνομα του υπό δημιουργία δείκτη.
2. *όνομα_πίνακα*, το οποίο περιέχει το όνομα του πίνακα πάνω στον οποίο θα δημιουργηθεί η δομή δείκτη.
3. *προσδιορισμός_μοναδικότητας*. Το πεδίο αυτό είτε έχει την τιμή *UNIQUE* ή είναι κενό, ανάλογα με το αν ο προσδιοριστής *UNIQUE* είναι παρών στην εντολή *CREATE INDEX* ή όχι, αντίστοιχα.
4. *στήλες_δείκτη*. Το πεδίο αυτό είναι μία λίστα, η οποία περιέχει ονόματα στηλών που ορίζεται ότι θα μετέχουν στη δομή δείκτη.

viii) Η εντολή DROP INDEX.

Η δομή δεδομένων που περιγράφει την εντολή *DROP INDEX* αποτελείται από ένα μόνο πεδίο, το *όνομα_δείκτη*, το οποίο περιέχει το όνομα του υπό διαγραφή δείκτη.

ix) Η εντολή HELP.

Στην υλοποίηση της SQL-XE έχει συμπεριληφθεί η εντολή *HELP*, η οποία συντάσσεται ως

HELP

ή

HELP όνομα_αντικειμένου

Στην πρώτη της μορφή, η εντολή *HELP* πληροφορεί τον χρήστη για τα ονόματα των υπαρχόντων πινάκων, ενώ στη δεύτερη παρέχει πληροφορίες για το σχήμα του προσδιοριζόμενου αντικειμένου (πίνακα ή δείκτη). Η δομή δεδομένων που περιγράφει την εντολή *HELP* αποτελείται από ένα μόνο πεδίο, το *όνομα_αντικειμένου*, το οποίο είτε είναι κενό, αν χρησιμοποιείται η πρώτη μορφή της εντολής, είτε περιέχει το όνομα του πίνακα, αν χρησιμοποιείται η δεύτερη σύνταξη.

x) Οι διαφεύγουσες εντολές.

Στην υλοποίηση της SQL-XE έχει συμπεριληφθεί η δυνατότητα να δίνονται εντολές, οι οποίες δεν θα τύχουν επεξεργασίας ή ανάλυσης από τον χρονολογικό φλοιό, αλλά θα προωθηθούν προς εκτέλεση από το σχεσιακό ΣΔΒΔ. Προκειμένου να προωθηθεί μία εντολή απ' ευθείας στο σχεσιακό ΣΔΒΔ, πρέπει να ξεκινά με το χαρακτήρα *!*, δηλαδή η σύνταξή της θα πρέπει να είναι

! εντολή_ΣΔΒΔ

Οι διαφεύγουσες εντολές υποστηρίζονται προκειμένου να είναι δυνατή η αξιοποίηση εντολών που παρέχει το ΣΔΒΔ, αλλά δεν αποτελούν τμήμα της SQL-XE (π.χ. στο Ingres η εντολή MODIFY μπορεί να χρησιμοποιηθεί για τον προσδιορισμό της φυσικής δομής αποθήκευσης των πινάκων (ISAM, B-δένδρο, ποσοστά πλήρωσης σελίδων δίσκου κ.λπ.)).

Ο χρονολογικός φλοιός απαγορεύει τη διαφυγή των εντολών *INSERT*, *DELETE*, *UPDATE* και *DROP* διότι μπορούν να παραβιάσουν την ακεραιότητα της βάσης, καθώς επίσης και των εντολών *SELECT* και *HELP*, διότι αυτές υπερκαλύπτονται από τις αντίστοιχες της SQL-XE.

4.3.6. Η βιβλιοθήκη της ΣΑΧΕ.

Στον χρονολογικό φλοιό είναι ενσωματωμένη μία *βιβλιοθήκη αλγορίθμων* της ΣΑΧΕ[4], η οποία περιέχει μία διαδικασία για κάθε μία από τις πράξεις *διαφορά*, *σύμπτυξη*, *ανάπτυξη*, *κανονικοποίηση*, *σημειακή ένωση* και *σημειακή διαφορά* της ΣΑΧΕ, καθώς επίσης και μία διαδικασία που υλοποιεί την πράξη *κατάτμηση* (split). Η πράξη *κατάτμηση* δεν ανήκει στον ορισμό της ΣΑΧΕ, αλλά χρησιμοποιείται εσωτερικά από την πράξη *κανονικοποίηση* για λόγους βελτιστοποίησης. Για τις πράξεις *επιλογή*, *προβολή*, *σύνδεση*, *ένωση* και *υπολογισμός* της ΣΑΧΕ δεν έχουν δημιουργηθεί διαδικασίες εκτέλεσης, διότι για την εκτέλεση αυτών των πράξεων χρησιμοποιούνται οι δυνατότητες του Ingres. Όλες οι πράξεις της ΣΑΧΕ επεξεργάζονται έναν ή δύο πίνακες εισόδου και παράγουν ως αποτέλεσμα έναν πίνακα εξόδου. Στις επόμενες παραγράφους περιγράφονται οι διαδικασίες της βιβλιοθήκης της ΣΑΧΕ, καθώς και η πράξη *κατάτμηση*.

I. Η πράξη *διαφορά*.

Η πράξη *διαφορά* υλοποιείται μέσω μιας αντισύνδεσης. Για να υπολογιστεί το αποτέλεσμα της πράξης *Π EXCEPT P*, όπου *Π* και *P* είναι πίνακες με σχήματα (π_1, ..., π_v) και (ρ_1, ..., ρ_v), αντίστοιχα και να αποθηκευθεί στον πίνακα *Σ*, σχηματίζεται η εντολή

```
CREATE TABLE Σ AS
SELECT * FROM Π
WHERE NOT EXISTS (SELECT * FROM P
                  WHERE Π.π1 = P.ρ1
                  AND ...
                  AND Π.πv = P.ρv)
```

η οποία και προωθείται προς εκτέλεση στο Ingres.

II. Η πράξη *σύμπτυξη*.

Για να υπολογιστεί το αποτέλεσμα της πράξης *FOLD [π_1, ..., π_κ](Π)* (το σχήμα του πίνακα *Π* θεωρείται ότι είναι (π_1, ..., π_v)), ο πίνακας *Π* συμπτύσσεται πρώτα ως προς τη στήλη π_1, κατόπιν το αποτέλεσμα της πράξης συμπτύσσεται ως προς τη στήλη π_2 κ.ο.κ. Το αποτέλεσμα της τελευταίας σύμπτυξης (πάνω στη στήλη π_κ) αποτελεί και το τελικό αποτέλεσμα της πράξης. (Η υπόθεση ότι η πράξη *σύμπτυξη* εφαρμόζεται στις πρώτες *κ* στήλες και με τη σειρά

---

[4] Η βιβλιοθήκη της ΣΑΧΕ αναπτύχθηκε στα πλαίσια του προγράμματος ORES, από την ομάδα του Τμήματος Πληροφορικής του Πανεπιστημίου Αθηνών που συμμετείχε σ' αυτό.

1, …, *κ*, γίνεται χωρίς βλάβη της γενικότητας.) Για να συμπτυχθεί ο πίνακας *Π* πάνω σε μία στήλη π_t, και να αποθηκευθεί το αποτέλεσμα στον πίνακα *P*, ακολουθείται ο κάτωθι αλγόριθμος:

1) ο τύπος της στήλης π_t ελέγχεται για να διαπιστωθεί ότι είναι τύπου *χρονικό διάστημα* ή *χρονικό σημείο*. Αν ο έλεγχος αποτύχει, η διαδικασία εκτέλεσης της πράξης *σύμπτυξη* αποτυγχάνει και επιστρέφει έναν κωδικό λάθους.
2) ο πίνακας *Π* ταξινομείται βάσει όλων των στηλών π_1, …, π_{t-1}, π_{t+1}, .., π_v, π_t, με αυτή τη σειρά.
3) διαβάζεται η πρώτη πλειάδα από τον ταξινομημένο πίνακα *Π* και αποθηκεύεται σε μία δομή στη μνήμη του χρονολογικού φλοιού με το όνομα *πλειάδα_εργασίας*. Αν ο τύπος της στήλης π_t του πίνακα *Π* είναι *χρονικό σημείο* τότε στη στήλη π_t της δομής *πλειάδα_εργασίας* αποθηκεύεται το αποτέλεσμα της συνάρτησης *tointerv(Π.*π_t*)*.
4) η επόμενη πλειάδα διαβάζεται από τον πίνακα *Π* και αποθηκεύεται σε μία δομή στη μνήμη του χρονολογικού φλοιού με το όνομα *τρέχουσα_πλειάδα*. Αν τα δεδομένα του πίνακα *Π* έχουν εξαντληθεί, η *πλειάδα_εργασίας* εισάγεται στον πίνακα *P* και ο αλγόριθμος τερματίζει. Αν ο τύπος της στήλης π_t του πίνακα *Π* είναι *χρονικό σημείο* τότε στη στήλη π_t της δομής *τρέχουσα_πλειάδα* αποθηκεύεται το αποτέλεσμα της συνάρτησης *tointerv(Π.*π_t*)*. Κατόπιν διενεργούνται οι εξής έλεγχοι:
   i) αν οποιαδήποτε από τις στήλες π_1, …, π_{t-1}, π_{t+1}, .., π_v της δομής *τρέχουσα_πλειάδα* έχει διαφορετική τιμή από την αντίστοιχη στήλη της δομής *πλειάδα_εργασίας* ή το πέρας της στήλης π_t της δομής *πλειάδα_εργασίας* είναι μικρότερο από την αρχή της ίδιας στήλης στη δομή *τρέχουσα_πλειάδα*, τότε η *πλειάδα_εργασίας* εισάγεται στον πίνακα *P*, οι τιμές των στηλών της δομής *τρέχουσα_πλειάδα* αντικαθιστούν τις τιμές των στηλών της δομής *πλειάδα_εργασίας* και το βήμα (4) επαναλαμβάνεται.
   ii) το πέρας της στήλης π_t της δομής *πλειάδα_εργασίας* τίθεται ίσο με το μέγιστο της τρέχουσας τιμής του και του πέρατος της στήλης π_t της δομής *τρέχουσα_πλειάδα*. Κατόπιν το βήμα (4) επαναλαμβάνεται.

III. Η πράξη *ανάπτυξη*.

Για να υπολογιστεί το αποτέλεσμα της πράξης *UNFOLD[*π_1, ..., π_κ*](Π)*, το οποίο θα αποθηκευθεί στον πίνακα *P*, ακολουθείται ο εξής αλγόριθμος (το σχήμα του πίνακα *Π* θεωρείται ότι είναι (π_1, …, π_v). Η υπόθεση ότι η ανάπτυξη γίνεται στις πρώτες *κ* στήλες

γίνεται χωρίς βλάβη της γενικότητας. Αν κάποια από τις στήλες π_1, ..., π_κ είναι τύπου *χρονικό σημείο*, τότε αφαιρείται από τις στήλες ανάπτυξης.):

1) ελέγχεται ότι κάθε μία από τις στήλες π_1, ..., π_κ είναι τύπου *χρονικό διάστημα*. Αν κάποια από αυτές είναι διαφορετικού τύπου, η πράξη *ανάπτυξη* αποτυγχάνει και επιστρέφει έναν κατάλληλο κωδικό λάθους.
2) διαβάζεται μία πλειάδα π του πίνακα Π. Αν τα δεδομένα του πίνακα έχουν εξαντληθεί, τότε ο αλγόριθμος τερματίζει, αλλιώς οι αρχές των διαστημάτων π_1, ..., π_κ αποθηκεύονται στη δομή *τρέχουσες_τιμές*.
3) εισάγεται στον πίνακα P μία πλειάδα, για την οποία οι στήλες π_1, ..., π_κ έχουν τιμές ίσες με τα αντίστοιχα πεδία της δομής *τρέχουσες_τιμές*, ενώ οι στήλες $\pi_{\kappa+1}$, ..., π_ν έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας π.
4) τα πεδία της δομής *τρέχουσες_τιμές* διατρέχονται, ξεκινώντας από το πρώτο. Για κάθε πεδίο εκτελούνται τα εξής βήματα:
    i) η τιμή του πεδίου αντικαθίσταται από το *αμέσως επόμενο* χρονικό σημείο. Αν η νέα τιμή είναι μικρότερη από το πέρας της αντίστοιχης στήλης, ο αλγόριθμος συνεχίζεται στο βήμα (3).
    ii) η τιμή του πεδίου αντικαθίσταται από την αρχή της αντίστοιχης στήλης. Αν το πεδίο είναι το τελευταίο της δομής *τρέχουσες_τιμές*, τότε ο αλγόριθμος συνεχίζεται στο βήμα (2), αλλιώς τα βήματα (i) και (ii) επαναλαμβάνονται για το επόμενο πεδίο της δομής *τρέχουσες_τιμές*.

Ο αλγόριθμος ολοκληρώνεται με την απομάκρυνση των διπλοτύπων πλειάδων από τον πίνακα P, εκτός αν έχει προσδιορισθεί (μέσω κάποιας ένδειξης που μεταβιβάζεται ως παράμετρος στη διαδικασία που υλοποιεί την πράξη) ότι οι διπλότυπες πλειάδες πρέπει να διατηρηθούν. Η απομάκρυνση των διπλοτύπων πλειάδων επιτυγχάνεται με την αντικατάσταση του πίνακα P από το αποτέλεσμα της ερώτησης

SELECT DISTINCT * FROM *P*

IV. Η πράξη *κατάτμηση*.

Η πράξη *κατάτμηση* συμβολίζεται με $P = SPLIT[\pi_1, ..., \pi_\kappa] (\Pi)$, όπου Π είναι ένας πίνακας και π_1, ..., π_κ είναι οι *στήλες κατάτμησης*, κάθε μία από τις οποίες πρέπει να είναι τύπου *χρονικό διάστημα* ή *χρονικό σημείο* και να ανήκει στο σχήμα του πίνακα Π. Το πλήρες σχήμα του πίνακα Π θεωρείται ότι είναι (π_1, ..., π_ν). Η υπόθεση ότι η πράξη εφαρμόζεται στις κ πρώτες στήλες του πίνακα γίνεται χωρίς βλάβη της γενικότητας. Αν κάποια από τις στήλες π_1,

..., π_κ είναι τύπου *χρονικό σημείο*, τότε αφαιρείται από τις στήλες κατάτμησης. Για τον πίνακα *P*, το αποτέλεσμα δηλαδή της πράξης *κατάτμηση*, ισχύουν τα ακόλουθα:

1) το σχήμα του είναι ίδιο με το σχήμα του πίνακα *Π*.
2) UNFOLD $[\pi_1, ..., \pi_\kappa](\Pi)$ = UNFOLD $[\pi_1, ..., \pi_\kappa](P)$
3) για κάθε πλειάδα χ του πίνακα *P* και κάθε πλειάδα ψ του πίνακα *Π* ισχύει ένα από τα ακόλουθα:
   i) $\chi.\pi_\iota \neq \psi.\pi_\iota$ για κάποιο $\iota = \kappa + 1, ..., \nu$
   ii) $\chi.\pi_\iota$ *subinterv* $\psi.\pi_\iota \vee$ NOT$(\chi_1.\pi_\iota$ *cp* $\chi_2.\pi_\iota) \forall = 1, ..., \kappa$
4) αν χ_1 και χ_2 πλειάδες του πίνακα *P*, τέτοιες ώστε:
   i) $\chi_1.\pi_\iota = \chi_2.\pi_\iota \forall \iota = \kappa + 1, ..., \nu$
   ii) $\exists \lambda \in [1, \kappa]$: $\chi_1.\pi_\iota$ *cp* $\chi_2.\pi_\iota \forall \iota = 1, ..., \kappa, \iota \neq \lambda \wedge \chi_1.\pi_\lambda$ *adjacent* $\chi_2.\pi_\lambda$

τότε υπάρχει πλειάδα ψ στον πίνακα *Π*, τέτοια ώστε:

   i) $\chi_1.\pi_\iota = \psi.\pi_\iota \forall \iota = \kappa + 1, ..., \nu$
   ii) merge$(\chi_1.\pi_\iota, \chi_2.\pi_\iota)$ *cp* $\psi.\pi_\iota \wedge$ NOT (merge$(\chi_1.\pi_\iota, \chi_2.\pi_\iota)$ *subinterv* $\psi.\pi_\iota)$ για κάποιο $\iota = 1, ..., \kappa$

Η δεύτερη ιδιότητα εξασφαλίζει ότι όλα τα χρονικά σημεία τα οποία υπάρχουν στη σχέση εισόδου με την πράξη *κατάτμηση* δεν δημιουργείται αλλά ούτε και χάνεται πληροφορία. Η τρίτη ιδιότητα εξασφαλίζει ότι η κατάτμηση του αρχικού πίνακα γίνεται κατά τρόπο ώστε ανάμεσα σε οποιαδήποτε πλειάδα χ του αρχικού πίνακα και οποιαδήποτε πλειάδα ψ του αποτελέσματος ισχύει μία από τις ακόλουθες συνθήκες:

1. είτε η πληροφορία που φέρει η πλειάδα ψ είναι ξένη προς την πληροφορία που φέρει η πλειάδα χ.
2. είτε η πληροφορία που φέρει η πλειάδα ψ είναι υποσύνολο της πληροφορίας που φέρει η πλειάδα χ.

Τέλος, η τέταρτη ιδιότητα εξασφαλίζει ότι η κατάτμηση αυτή γίνεται με τον ελάχιστο αριθμό πλειάδων.

Απόρροια της τρίτης ιδιότητας είναι ότι το αποτέλεσμα της πράξης *σύμπτυξη* πάνω σε οποιαδήποτε διάταξη των στηλών $\pi_1, ..., \pi_\kappa$ του πίνακα *Σ* είναι ίσο με το αποτέλεσμα της πράξης *κανονικοποίηση* πάνω στην ίδια διάταξη στηλών. Κατά συνέπεια, η πράξη *κανονικοποίηση* μπορεί να υλοποιηθεί ως

NORMALISE$[\pi_1, ..., \pi_\kappa]$ (Π) = FOLD$[\pi_1, ..., \pi_\kappa]$ (SPLIT$[\pi_1, ..., \pi_\kappa]$ (Π))

αντί για

NORMALISE[π_1, ..., π_κ] (Π) = FOLD[π_1, ..., π_κ] (UNFOLD[π_1, ..., π_κ] (Π))

Η υλοποίηση αυτή θα είναι πιο αποδοτική, καθώς η πράξη *κατάτμηση* παράγει σημαντικά μικρότερο αριθμό πλειάδων απ' ότι η πράξη *ανάπτυξη*. Ο αριθμός των πλειάδων στο αποτέλεσμα της πράξης *κατάτμηση* θα είναι το πολύ $(2 * N - 1)^\kappa$, όπου *N* είναι ο αριθμός πλειάδων στη σχέση *Π* και *κ* το πλήθος στηλών πάνω στις οποίες εφαρμόζεται η πράξη *κατάτμηση*. Όπως παρατηρούμε, ο αριθμός πλειάδων στο αποτέλεσμα της πράξης *κατάτμηση* είναι ανεξάρτητος του αριθμού χρονικών σημείων που περιέχονται στα διαστήματα της σχέσης *Π*.

Για την εκτέλεση της πράξης $P = SPLIT[\pi_1, ..., \pi_\kappa](\Pi)$ ακολουθείται ο κάτωθι αλγόριθμος:

1) ο πίνακας *Π* ταξινομείται ως προς τις στήλες $\pi_{\kappa+1}$, ..., π_ν.
2) διαβάζεται η επόμενη πλειάδα από τον ταξινομημένο πίνακα *Π*. Οι στήλες $\pi_{\kappa+1}$, ..., π_ν αποθηκεύονται στη δομή *τρέχουσες_τιμές*, ενώ κάθε μία από τις στήλες π_1, ..., π_κ αρχικοποιεί μία λίστα στη δομή *διαστήματα_εξόδου*. Κάθε λίστα στη δομή *διαστήματα_εξόδου* είναι μία ταξινομημένη λίστα σημείων, κάθε ένα από τα οποία έχει το χαρακτηρισμό *αρχή* ή *πέρας*. Η αρχή κάθε στήλης π_i εισάγεται στην αντίστοιχη λίστα με το χαρακτηρισμό *αρχή*, ενώ το πέρας της εισάγεται στην ίδια λίστα με το χαρακτηρισμό *πέρας*.
3) διαβάζεται η επόμενη πλειάδα από τον ταξινομημένο πίνακα *Π*. Αν τα δεδομένα του πίνακα έχουν εξαντληθεί, τότε τίθεται μία ένδειξη τέλους δεδομένων εισόδου και ο αλγόριθμος συνεχίζεται στο βήμα (5). Αν οποιαδήποτε από τις στήλες $\pi_{\kappa+1}$, ..., π_ν της πλειάδας έχει τιμή διαφορετική από το αντίστοιχο πεδίο της δομής *τρέχουσες_τιμές*, τότε η πλειάδα αποθηκεύεται σε μία δομή δεδομένων στη μνήμη του χρονολογικού φλοιού και ο αλγόριθμος συνεχίζεται στο βήμα (5).
4) για κάθε μία από τις στήλες π_1, ..., π_κ της πλειάδας εκτελούνται τα εξής βήματα:
   i) εντοπίζεται η θέση στην οποία πρέπει να προστεθεί η αρχή της στήλης στην αντίστοιχη λίστα της δομής *διαστήματα_εξόδου*. Αν το χρονικό αυτό σημείο είναι ήδη καταχωρισμένο στη λίστα, τότε σημειώνεται ότι ο χαρακτηρισμός του πρέπει να τροποποιηθεί σε *αρχή* (αν ο χαρακτηρισμός του είναι ήδη *αρχή*, τότε η τροποποίηση δεν θα επιφέρει καμία αλλαγή), ενώ στην αντίθετη περίπτωση σημειώνεται ότι πρέπει να προστεθεί με τον χαρακτηρισμό *αρχή*.
   ii) εντοπίζεται η θέση στην οποία πρέπει να προστεθεί το πέρας της στήλης στην αντίστοιχη λίστα. Αν το χρονικό αυτό σημείο είναι ήδη καταχωρισμένο στη λίστα,

τότε ο αλγόριθμος συνεχίζεται στο βήμα (iv), αλλιώς σημειώνεται ότι το χρονικό σημείο πρέπει να προστεθεί με το χαρακτηρισμό *πέρας*.

iii) αν το πέρας της στήλης θα προστεθεί αμέσως μετά από κόμβο που είναι χαρακτηρισμένος ως *αρχή*, τότε σημειώνεται ότι και το ίδιο πρέπει να προστεθεί με τον χαρακτηρισμό *αρχή*.

iv) εκτελούνται οι ενέργειες που έχουν σημειωθεί (προσθήκες στη λίστα και αλλαγές χαρακτηρισμών) και ο χαρακτηρισμός όλων των σημείων που βρίσκονται ανάμεσα από την αρχή και το πέρας της στήλης τροποποιείται σε *αρχή*.

Όταν εκτελεστούν τα βήματα (i) έως (iv) για όλες τις στήλες π_1, ..., π_κ, το βήμα (3) επαναλαμβάνεται.

5) οι λίστες της δομής *διαστήματα_εξόδου* διατρέχονται και εντοπίζονται ζεύγη διαδοχικών σημείων (σ_1, σ_2), για οποία ο χαρακτηρισμός του πρώτου είναι *αρχή*. Για κάθε συνδυασμό τέτοιων ζευγών εισάγεται στον πίνακα Σ μία πλειάδα, της οποίας οι στήλες π_1 έως π_κ έχουν ως τιμές τα διαστήματα $[\sigma_1, \sigma_2)$ των αντίστοιχων λιστών της δομής *διαστήματα_εξόδου*, ενώ οι στήλες $\pi_{\kappa+1}$, ..., π_ν έχουν τιμές ίσες με τα αντίστοιχα πεδία της δομής *τρέχουσες_τιμές*.

6) αν έχει τεθεί η ένδειξη τέλους δεδομένων εισόδου τότε ο αλγόριθμος τερματίζει, αλλιώς η πλειάδα που έχει αποθηκευθεί στο βήμα (3) αρχικοποιεί τις δομές *τρέχουσες_τιμές* και *διαστήματα_εξόδου*, όπως περιγράφεται στο βήμα (2) και το βήμα (3) εκτελείται ξανά.

V. Η πράξη *κανονικοποίηση*.

Για να υπολογιστεί το αποτέλεσμα της πράξης $NORMALISE[\pi_1, ..., \pi_\kappa](\Pi)$ δημιουργείται αρχικά ένας προσωρινός πίνακας *T1* που περιέχει το αποτέλεσμα της πράξης $SPLIT[\pi_1, ..., \pi_\kappa](\Pi)$, και στη συνέχεια εκτελείται η πράξη $FOLD[\pi_1, ..., \pi_\kappa](T1)$, η οποία παράγει και το τελικό αποτέλεσμα. Η πράξη *κανονικοποίηση* ολοκληρώνεται με την καταστροφή του προσωρινού πίνακα *T1*.

VI. Η πράξη *σημειακή ένωση*.

Για να υπολογιστεί το αποτέλεσμα της πράξης $\Pi\ PUNION\ [\pi_1, ..., \pi_\kappa]\ P$ δημιουργείται αρχικά ένας προσωρινός πίνακας *T1* στον οποίο αποθηκεύεται η ένωση των δύο πινάκων. Στη συνέχεια, εκτελείται η πράξη $NORMALISE[\pi_1, ..., \pi_\kappa](T1)$, η οποία παράγει το τελικό αποτέλεσμα. Η πράξη *σημειακή ένωση* ολοκληρώνεται με την καταστροφή του προσωρινού πίνακα *T1*.

VII. Η πράξη *σημειακή διαφορά*.

Σημασιολογικά, η πράξη *σημειακή διαφορά* μεταξύ δύο πινάκων αναπτύσσει κάθε έναν από τους πίνακες στις καθοριζόμενες στήλες, στη συνέχεια υπολογίζει τη διαφορά των δύο αναπτυγμένων πινάκων και, τέλος, συμπτύσσει το αποτέλεσμα στις καθοριζόμενες στήλες. Ο αλγόριθμος αυτός δεν είναι κατάλληλος για την υλοποίηση της πράξης *σημειακή διαφορά*, μια και η ανάπτυξη ενός πίνακα με *ν* πλειάδες ως προς μία στήλη θα παράγει έναν πίνακα με *ν* * *M* πλειάδες, όπου *M* είναι ο μέσος αριθμός σημείων που περιέχονται στη συγκεκριμένη στήλη κάθε πλειάδας. Για υψηλές τιμές του *M*, το μέγεθος του πίνακα που θα προκύψει μπορεί να είναι υπερβολικά μεγάλο, επηρεάζοντας τόσο τον χώρο που απαιτείται για την αποθήκευση των ενδιαμέσων αποτελεσμάτων, όσο και τον χρόνο που απαιτείται για να υπολογιστεί το τελικό αποτέλεσμα. Ένας πιο αποτελεσματικός αλγόριθμος για την υλοποίηση της πράξης *σημειακή διαφορά* είναι ο ακόλουθος (στα επόμενα, η πρώτη σχέση θα συμβολίζεται με *Π* και το σχήμα της με (π_1, ..., π_ν), ενώ η δεύτερη σχέση θα συμβολίζεται με *P* και το σχήμα της με (ρ_1, ..., ρ_ν). Ο πίνακας στον οποίο θα αποθηκευθεί το αποτέλεσμα θα συμβολίζεται με *Σ* και το σχήμα του είναι ίδιο με το σχήμα του πίνακα *Π*. Χωρίς βλάβη της γενικότητας, θεωρούμε ότι η πράξη αυτή εφαρμόζεται στις στήλες π_1, ..., π_κ):

1. εκτελείται η πράξη *T1 = SPLIT'[*π_1*, ..., *π_κ*] (Π, P)*. Η πράξη *SPLIT'* είναι μία παραλλαγή της πράξης *κατάτμηση*, η οποία διαμορφώνει τις λίστες της δομής *διαστήματα εξόδου* (βλέπε *Η πράξη κατάτμηση.*) λαμβάνοντας υπόψη και τις πλειάδες του πίνακα *P*.
2. σχηματίζεται η εντολή

CREATE TABLE *T2* AS
SELECT * FROM *T1*
WHERE NOT EXISTS (SELECT * FROM *P*
WHERE (cp(*T1*.π_1, *P*.ρ_1) = 1) AND ... AND (cp(*T1*.π_κ, *P*.ρ_κ) = 1)
AND *T1*.$\pi_{\kappa+1}$ = *P*.$\rho_{\kappa+1}$ AND ... AND *T1*.π_ν = *P*.ρ_ν)

η οποία και προωθείται στο Ingres για εκτέλεση.

3. εκτελείται η πράξη *Σ = FOLD [*π_1*, ..., *π_κ*](T2)*, η οποία παράγει το τελικό αποτέλεσμα (ο πίνακας *T2* έχει δημιουργηθεί στο βήμα 2). Ο αλγόριθμος ολοκληρώνεται με την καταστροφή των προσωρινών πινάκων *T1* και *T2*.

4.3.7. Οι διαδικασίες εκτέλεσης εντολών.

Ο χρονολογικός φλοιός περιέχει μία διαδικασία εκτέλεσης για κάθε εντολή της SQL-XE. Στις επόμενες παραγράφους παρουσιάζονται οι διαδικασίες εκτέλεσης των εντολών.

I. Η εντολή *CREATE TABLE*.

Η διαδικασία εκτέλεσης της εντολής *CREATE TABLE* ελέγχει αν εντολή που δόθηκε πληρεί τους σημασιολογικούς κανόνες που ορίζονται από την SQL-XE, δημιουργεί τον πίνακα και τις απαραίτητες δευτερεύουσες δομές στο επίπεδο του Ingres και εισάγει καταχωρίσεις στο επεκταμένο λεξικό, αν αυτό είναι απαραίτητο. Αναλυτικότερα, για τη δημιουργία ενός πίνακα ακολουθείται ο κάτωθι αλγόριθμος:

1. ελέγχεται αν στο πεδίο *στήλες_κανονικοποίησης* (βλέπε σελ. 62, *Η εντολή CREATE TABLE.*) περιέχεται το πολύ μία στήλη. Αν το πεδίο περιέχει πάνω από μία στήλη, τότε η εντολή διαδικασία εκτέλεσης της εντολής τερματίζεται, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος.
2. αν το πεδίο *στήλες_κανονικοποίησης* δεν είναι κενό, ελέγχεται αν η στήλη που προσδιορίζεται υπάρχει στο πεδίο *στήλες_πίνακα* και δηλώνεται ως τύπου *DATEINTERVAL*. Αν οποιοσδήποτε έλεγχος αποτύχει, η εντολή διαδικασία εκτέλεσης της εντολής τερματίζεται, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος. Στην αντίθετη περίπτωση, το πεδίο *ιδιότητα* της τριπλέτας της δομής *στήλες_πίνακα* που αντιστοιχεί στην προσδιοριζόμενη στήλη αποκτά την τιμή *NOT NULL*.
3. ελέγχεται ότι όλες οι στήλες που περιέχονται στο πεδίο *πρωτεύον_κλειδί* υπάρχουν στο πεδίο *στήλες_πίνακα*. Αν ο έλεγχος αποτύχει, η εντολή διαδικασία εκτέλεσης της εντολής τερματίζεται, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος. Στην αντίθετη περίπτωση, τα πεδία *ιδιότητα* των τριπλετών της δομής *στήλες_πίνακα* που αντιστοιχούν στις προσδιοριζόμενες στήλες, αποκτούν την τιμή *NOT NULL*. Τέλος, ελέγχεται αν ο προσδιορισμός τύπου κλειδιού *POINT* χρησιμοποιείται μόνο για τη στήλη που περιέχεται στο πεδίο *στήλες_κανονικοποίησης*.
4. ο πίνακας δημιουργείται στο επίπεδο του Ingres, με την εκτέλεση της κατάλληλης εντολής SQL και εισάγονται στους πίνακες *vtsql_keys* και *vtsql_norms* καταχωρίσεις για τις στήλες που αναφέρονται στα πεδία *πρωτεύον_κλειδί* και *στήλες_κανονικοποίησης*. Τέλος, αν το πεδίο *πρωτεύον_κλειδί* δεν είναι κενό, τότε δημιουργείται στο επίπεδο του Ingres ένας δείκτης πάνω σε όλες τις στήλες που αναφέρονται σ' αυτό. Αυτός ο δείκτης δηλώνεται ως *μοναδικός* (unique), εξασφαλίζοντας τη διατήρηση της μοναδικότητας των κλειδιών τύπου *διάστημα*.

II. Η εντολή *DROP TABLE*.

Η διαδικασία εκτέλεσης της εντολής *DROP TABLE* καταστρέφει τον προσδιοριζόμενο πίνακα στο επίπεδο του Ingres, προωθώντας την κατάλληλη εντολή *DROP TABLE* στο Ingres. Επίσης, φροντίζει για την αφαίρεση όλων των καταχωρίσεων που σχετίζονται με τον πίνακα που καταστράφηκε από τους πίνακες *vtsql_keys* και *vtsql_norms*, προωθώντας στο Ingres τις κατάλληλες εντολές *DELETE*.

III. Η εντολή *SELECT*.

Η διαδικασία εκτέλεσης της εντολής *SELECT* αναλύει τις δομές δεδομένων με τις οποίες εφοδιάζεται από τον συντακτικό αναλυτή, προκειμένου να διαμορφώσει το σχέδιο εκτέλεσης της ερώτησης. Αναλυτικά, ο αλγόριθμος εκτέλεσης της εντολής *SELECT* έχει ως ακολούθως:

1) αν το πεδίο *είδος_συνδυασμού* (βλέπε σελ. 63, *Η εντολή SELECT*.) είναι κενό και τα πεδία *προσδιορισμός_αναμόρφωσης* και *προσδιορισμός_κανονικοποίησης* της δομής *πρώτη_ερώτηση* είναι κενά ή το πεδίο *είδος_συνδυασμού* έχει μία από τις τιμές *UNION* και *UNION ALL*, το πεδίο *στήλες_συνδυασμού* είναι κενό και τα πεδία *προσδιορισμός_αναμόρφωσης* και *προσδιορισμός_κανονικοποίησης* των δομών *πρώτη_ερώτηση* και *δεύτερη_ερώτηση* είναι κενά, τότε η ερώτηση μπορεί να αποτιμηθεί άμεσα από το Ingres. Η εντολή *SELECT* ανασυντίθεται στην αρχική μορφή της (στη μορφή δηλαδή που δόθηκε από τον χρήστη), προωθείται στο Ingres για εκτέλεση και η διαδικασία εκτέλεσης της εντολής *SELECT* τερματίζει.
2) αν το πεδίο *είδος_συνδυασμού* είναι κενό και κάποιο από τα πεδία *προσδιορισμός_αναμόρφωσης* και *προσδιορισμός_κανονικοποίησης* της δομής *πρώτη_ερώτηση* (ενδεχομένως και τα δύο) δεν είναι κενό, τότε ακολουθείται η κάτωθι διαδικασία:
    i) τα πεδία *ποσοδείκτης*, *στήλες_αποτελέσματος*, *πίνακες*, *συνθήκη* και *προσδιορισμός_ομαδοποίησης* της δομής *πρώτη_ερώτηση* συνδυάζονται για να δημιουργηθεί μία εντολή *SELECT* που είναι σύμφωνη με το πρότυπο SQL89. Κατόπιν, υπολογίζεται το σχήμα του αποτελέσματος της ερώτησης αυτής και ελέγχεται ότι όλες οι πράξεις που προσδιορίζονται στα πεδία *προσδιορισμός_αναμόρφωσης* και *προσδιορισμός_κανονικοποίησης* εφαρμόζονται σε στήλες τύπου *DATE* ή *DATEINTERVAL*. Αν ο έλεγχος αποτύχει, η εντολή διαδικασία εκτέλεσης της εντολής τερματίζεται, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος. Στην αντίθετη περίπτωση, η εντολή *SELECT* που σχηματίστηκε

σ' αυτό το βήμα προωθείται προς εκτέλεση στο Ingres και τα αποτελέσματα αποθηκεύονται σε έναν προσωρινό πίνακα.

ii) αν το πεδίο *προσδιορισμός_αναμόρφωσης* δεν είναι κενό, τότε καλείται το *υποσύστημα βελτιστοποίησης* για να απαλειφθούν περιττές πράξεις (το υποσύστημα βελτιστοποίησης περιγράφεται στη συνέχεια). Στη συνέχεια, εκτελούνται οι πράξεις που υποδεικνύονται από τον κάθε κόμβο της λίστας *προσδιορισμός_αναμόρφωσης*, κάθε μία από τις οποίες εφαρμόζεται στον τελευταίο προσωρινό πίνακα που δημιουργήθηκε και παράγει έναν νέο προσωρινό πίνακα. Μετά την ολοκλήρωση κάθε πράξης, ο προσωρινός πίνακας πάνω στον οποίο η πράξη εφαρμόστηκε, καταστρέφεται.

iii) αν το πεδίο *προσδιορισμός_κανονικοποίησης* δεν είναι κενό, τότε καλείται το *υποσύστημα βελτιστοποίησης* (αν δεν κλήθηκε στο προηγούμενο βήμα) για να απαλειφθούν περιττές πράξεις. Στη συνέχεια, εφαρμόζεται η πράξη *κανονικοποίηση* πάνω στον τελευταίο προσωρινό πίνακα που έχει προκύψει από τα βήματα (i) και (ii) και ο πίνακας αυτός καταστρέφεται. Το αποτέλεσμα της πράξης αποθηκεύεται σε έναν νέο προσωρινό πίνακα.

iv) οι πλειάδες του τελευταίου προσωρινού πίνακα που έχει προκύψει από τα βήματα (i) έως (iii) ανακτώνται, ενδεχομένως ταξινομημένες σύμφωνα με τα προσδιοριζόμενα στο πεδίο *στήλες_ταξινόμησης*, αν αυτό δεν είναι κενό, παρουσιάζονται στον χρήστη και ο πίνακας αυτός καταστρέφεται.

3) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *EXCEPT* ή/και το πεδίο *στήλες_συνδυασμού* δεν είναι κενό, τότε:

i) υπολογίζονται τα σχήματα των αποτελεσμάτων των ερωτήσεων που περιγράφονται από τις δομές *πρώτη_ερώτηση* και *δεύτερη_ερώτηση* και ελέγχονται για συμβατότητα. Αν τα σχήματα είναι ασύμβατα, τότε η διαδικασία εκτέλεσης της εντολής *SELECT* τερματίζει επιστρέφοντας έναν κατάλληλο κωδικό λάθους. Αν το πεδίο *στήλες_συνδυασμού* δεν είναι κενό, τότε ελέγχεται ότι οι στήλες αποτελέσματος που αναφέρονται στο πεδίο αυτό είναι όλες τύπου *DATE* ή *DATEINTERVAL* και, αν ο έλεγχος αποτύχει, η διαδικασία εκτέλεσης της εντολής *SELECT* τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

ii) η ερώτηση που περιγράφεται από τη δομή *πρώτη_ερώτηση* αποτιμάται, βάσει των βημάτων (i) έως (iii) της περίπτωσης (2) και τα αποτελέσματα αποθηκεύονται σε έναν προσωρινό πίνακα *T1*. Αν κατά τη διάρκεια αποτίμησης της ερώτησης

προκύψει σφάλμα, τότε η διαδικασία εκτέλεσης της εντολής *SELECT* τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

iii) η ερώτηση που περιγράφεται από τη δομή *δεύτερη_ερώτηση* αποτιμάται, βάσει των βημάτων (i) έως (iii) της περίπτωσης (2) και τα αποτελέσματα αποθηκεύονται σε έναν δεύτερο προσωρινό πίνακα *T2*. Αν κατά τη διάρκεια αποτίμησης της ερώτησης προκύψει σφάλμα, τότε ο προσωρινός πίνακας *T1* καταστρέφεται και η διαδικασία εκτέλεσης της εντολής *SELECT* τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

iv) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *UNION* ή *UNION ALL* και το πεδίο *στήλες_συνδυασμού* είναι κενό, τότε σχηματίζεται μία ερώτηση SQL που ανακτά τις πλειάδες που περιέχονται στην ένωση των προσωρινών πινάκων *T1* και *T2*, ταξινομημένες σύμφωνα με τα προσδιοριζόμενα στο πεδίο *στήλες_ταξινόμησης* (αν το πεδίο αυτό δεν είναι κενό). Η ερώτηση αυτή προωθείται στο Ingres για εκτέλεση και ο αλγόριθμος τερματίζει με την καταστροφή των προσωρινών πινάκων *T1* και *T2*.

v) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *EXCEPT* ή/και το πεδίο *στήλες_συνδυασμού* δεν είναι κενό, τότε καλείται μία διαδικασία της βιβλιοθήκης αλγορίθμων της ΣΑΧΕ, η οποία επεξεργάζεται τους προσωρινούς πίνακες *T1* και *T2*, παράγοντας έναν νέο προσωρινό πίνακα *T3*. Η διαδικασία βιβλιοθήκης που θα κληθεί, επιλέγεται σύμφωνα με τα ακόλουθα κριτήρια:

   a) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *EXCEPT* και το πεδίο *στήλες_συνδυασμού* είναι κενό, τότε επιλέγεται η διαδικασία βιβλιοθήκης *EXCEPT*.

   b) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *UNION* ή *UNION ALL* και το πεδίο *στήλες_συνδυασμού* δεν είναι κενό, τότε επιλέγεται η διαδικασία βιβλιοθήκης *PUNION*. Η πράξη *σημειακή ένωση* θα εφαρμοστεί πάνω στις στήλες που αναφέρονται στο πεδίο *στήλες_συνδυασμού*.

   c) αν το πεδίο *είδος_συνδυασμού* έχει τιμή *EXCEPT* και το πεδίο *στήλες_συνδυασμού* δεν είναι κενό, τότε επιλέγεται η διαδικασία βιβλιοθήκης *PEXCEPT*. Η πράξη *σημειακή διαφορά* θα εφαρμοστεί πάνω στις στήλες που αναφέρονται στο πεδίο *στήλες_συνδυασμού*.

Τέλος, οι προσωρινοί πίνακες *T1* και *T2* καταστρέφονται, τα περιεχόμενα του προσωρινού πίνακα *T3* ανακτώνται και τυπώνονται, ταξινομημένα σύμφωνα με τα

προσδιοριζόμενα στο πεδίο *στήλες_ταξινόμησης* και ο προσωρινός πίνακας *T3* καταστρέφεται.

Όπως αναφέρθηκε, οι δομές που περιγράφουν τις προτάσεις *REFORMAT AS* και *NORMALISE* υπόκεινται σε βελτιστοποίηση, πριν εκτελεστούν οι πράξεις που αναφέρονται σ' αυτές, προκειμένου να απαλειφθούν περιττές πράξεις. Η διαδικασία βελτιστοποίησης έχει ως ακολούθως:

1) οι κόμβοι της λίστας *προσδιορισμοί_αναμόρφωσης* που είναι τύπου *UNFOLD ALL* και ακολουθούνται (όχι απαραίτητα άμεσα) από κόμβους τύπου *UNFOLD* ή *FOLD*, μετατρέπονται σε κόμβους τύπου *UNFOLD*. Η μετατροπή αυτή δεν επηρεάζει το τελικό αποτέλεσμα, δεδομένου ότι η πράξη *UNFOLD* ή *FOLD* θα αφαιρούσε τις διπλότυπες πλειάδες που ενδεχομένως θα παρήγαγε η πράξη *UNFOLD ALL*.
2) αν δύο διαδοχικοί κόμβοι της λίστας *προσδιορισμοί_αναμόρφωσης* είναι ίδιου τύπου, τότε συμπτύσσονται σε έναν μόνο κόμβο του ιδίου τύπου. Το πεδίο *στήλες* του κόμβου αυτού σχηματίζεται από την επισύναψη του πεδίου *στήλες* του δεύτερου κόμβου στο τέλος του πεδίου *στήλες* του πρώτου κόμβου.
3) οι κόμβοι της λίστας *προσδιορισμοί_αναμόρφωσης* που είναι τύπου *UNFOLD* και ακολουθούνται από κόμβους τύπου *FOLD* μετατρέπονται σε κόμβους τύπου *UNFOLD ALL*. Η μετατροπή αυτή δεν επηρεάζει το τελικό αποτέλεσμα, καθώς οι διπλότυπες πλειάδες που θα παραχθούν από την πράξη *UNFOLD ALL* θα αφαιρεθούν από την πράξη *FOLD*.
4) διαδοχικές εμφανίσεις μιας στήλης στο πεδίο *στήλες* ενός κόμβου τύπου *FOLD* της λίστας *προσδιορισμοί_αναμόρφωσης*, αντκαθίστανται με μία μόνο εμφάνιση. Επίσης, πολλαπλές (όχι απαραίτητα διαδοχικές) αναφορές μιας στήλης στο πεδίο *στήλες* ενός κόμβου τύπου *UNFOLD* ή *UNFOLD ALL* της λίστας *προσδιορισμοί_αναμόρφωσης*, αντικαθίστανται με την πρώτη εμφάνιση.
5) αν κάποια στήλη αναφέρεται στη λίστα *στήλες_κανονικοποίησης* περισσότερες από μία φορές, τότε διατηρείται μόνο η πρώτη εμφάνισή της.

Οι μετασχηματισμοί (1) και (2) μειώνουν τον αριθμό κλήσεων σε διαδικασίες της βιβλιοθήκης της ΣΑΧΕ που πρέπει να πραγματοποιηθούν. Η μείωση του αριθμού κλήσεων μπορεί να έχει ως αποτέλεσμα και τη μείωση των ενδιάμεσων πινάκων που πρέπει να παραχθούν, αν οι καλούμενες διαδικασίες έχουν κωδικοποιηθεί κατά τρόπο ώστε να υπολογίζουν το τελικό αποτέλεσμα με μία μόνο ανάγνωση του πίνακα εισόδου (όπως, π.χ. οι διαδικασίες *UNFOLD* και *SPLIT*). Ο μετασχηματισμός (3) εξαλείφει περιττές απομακρύνσεις

διπλοτύπων πλειάδων (οι οποίες συνήθως υλοποιούνται μέσω αλγορίθμων ταξινόμησης, που είναι αρκετά χρονοβόροι) και οι μετασχηματισμοί (4) και (5) απαλείφουν πράξεις, των οποίων το αποτέλεσμα θα ήταν ίσο με την είσοδό τους.

Στο σχήμα 4.2 παρουσιάζεται ένα παράδειγμα εφαρμογής των μετασχηματισμών (1) έως (4), οι οποίοι αποσκοπούν στη βελτιστοποίηση των πράξεων που προσδιορίζονται στη λίστα *προσδιορισμοί_αναμόρφωσης*. Σημειώνουμε, τέλος, ότι υπάρχουν περαιτέρω δυνατότητες βετλιστοποίησης, οι οποίες δεν έχουν υλοποιηθεί.

fold 1	fold 1	fold 1, 1, 2	fold 1, 1, 2	fold 1, 2
fold 1, 2	fold 1, 2	unfold 3, 4, 5, 4, 3	unfold all 3, 4, 5, 4, 3	unfold all 3, 4, 5
unfold 3, 4	unfold 3, 4	fold 6	fold 6	fold 6
unfold all 5, 4	unfold 5, 4	unfold 7, 8, 8, 9	unfold all 7, 8, 8, 9	unfold all 7, 8, 9
unfold all 3	unfold 3	fold 10	fold 10	fold 10
fold 6	fold 6	unfold all 11	unfold all 11	unfold all 11
unfold all 7, 8	unfold 7, 8			
unfold 8, 9	unfold 8, 9			
fold 10	fold 10			
unfold all 11	unfold all 11			
Αρχική μορφή	Βήμα (1)	Βήμα (2)	Βήμα (3)	Βήμα (4)

Σχήμα 4.2 - Οι μετασχηματισμοί βελτιστοποίησης για τις πράξεις αναμόρφωσης.

Ο αλγόριθμος αποτίμησης του αποτελέσματος της εντολής *SELECT* απαιτεί, σε αρκετά σημεία, τη δημιουργία ενδιάμεσων πινάκων στους οποίους θα αποθηκεύεται το αποτέλεσμα κάποιας ερώτησης. Κατά τη δημιουργία των προσωρινών πινάκων πρέπει να λαμβάνεται πρόνοια ώστε τα ονόματα των στηλών του προσωρινού πίνακα να είναι μοναδικά, ειδάλλως το Ingres δεν θα μπορέσει να δημιουργήσει τον ζητούμενο πίνακα. Σύμφωνα με αυτά, το αποτέλεσμα μιας ερώτησης *ε1* δεν μπορεί να αποθηκευθεί στον προσωρινό πίνακα *π1* απλά προωθώντας στο Ingres την εντολή

CREATE TABLE π1 AS ε1

διότι αν στο σχήμα του αποτελέσματος της ερώτησης *ε1* περιλαμβάνονται δύο στήλες με το ίδιο όνομα, το Ingres θα απορρίψει την εντολή.

Προκειμένου να αποφευχθούν προβλήματα αυτού του είδους, οι στήλες του σχήματος του αποτελέσματος της ερώτησης *ε1* μετονομάζονται και τους αποδίδονται μοναδικά ονόματα, ενώ τα αρχικά τους ονόματα φυλάσσονται για να χρησιμοποιηθούν κατά την παρουσίαση των αποτελεσμάτων στον χρήστη.

Δεδομένου ότι οι στήλες στους ενδιάμεσους πίνακες έχουν διαφορετικά ονόματα, απ' ότι οι αντίστοιχες στήλες στους αρχικούς πίνακες, όταν καλείται κάποια διαδικασία της βιβλιοθήκης της ΣΑΧΕ που δέχεται ως παραμέτρους ονόματα στηλών (π.χ. *FOLD*, *PUNION*), η καλούσα διαδικασία την εφοδιάζει με το μοναδικό όνομα που έχει αποδοθεί στη στήλη αντί για το όνομα που έχει δώσει ο χρήστης. Επίσης, οι αναφορές σε στήλες που περιέχονται στην πρόταση *ORDER BY* μετατρέπονται σε αύξοντες αριθμούς στήλης. Στο σχήμα 4.3 παρουσιάζεται η λειτουργικότητα του σχήματος μετονομασίας στηλών.

```
select s.sno, sp.sno, time = sp.period
from s, sp
where s.sno > sp.sno

reformat as unfold time

order by s.sno, sp.sno
```

(α)

```
create table temp1 as
select attr0 = s.sno, attr1 = sp.sno, attr2 = sp.period
from s, sp
where s.sno > sp.sno

temp2 = unfold [attr2] (temp1)

select sno = attr0, sno = attr1, time = attr2
from temp2 order by 1, 2
```

(β)

Σχήμα 4.3 - (α) Μία ερώτηση, της οποίας η αποτίμηση απαιτεί δημιουργία προσωρινού πίνακα. (β) Οι πράξεις που εκτελούνται, βάσει του σχήματος μετονομασίας στηλών.

IV. Η εντολή *INSERT*.

Η διαδικασία εκτέλεσης της εντολής *INSERT* αρχικά ελέγχει αν ο πίνακας που αναφέρεται στο πεδίο *όνομα_πίνακα* (βλέπε σελ. 65, *Η εντολή INSERT*.) είναι κανονικοποιημένος ή όχι, εξετάζοντας τον πίνακα *vtsql_norms*. Κατόπιν διακρίνονται οι ακόλουθες περιπτώσεις (στα επόμενα ο πίνακας στον οποίο θα εισαχθούν τα δεδομένα θα συμβολίζεται με Π. Το σχήμα του πίνακα Π θα συμβολίζεται με Π_1, ..., Π_v, $\Pi_{\chi\varepsilon}$, όπου $\Pi_{\chi\varepsilon}$ είναι η στήλη κανονικοποίησης του πίνακα (αν ο πίνακας δεν είναι κανονικοποιημένος, τότε η στήλη $\Pi_{\chi\varepsilon}$ δεν θεωρείται μέρος του σχήματος του πίνακα). Οι στήλες του πίνακα Π_1 έως Π_v χωρίζονται σε δύο ξένα μεταξύ τους σύνολα, τα $\Pi_{κλειδί}$ και $\Pi_{\neg κλειδί}$, το πρώτο από τα οποία περιέχει τις στήλες που ανήκουν στο πρωτεύον κλειδί, ενώ το δεύτερο αυτές που δεν μετέχουν. Αν δεν έχει ορισθεί πρωτεύον κλειδί πάνω στον πίνακα Π, το σύνολο $\Pi_{κλειδί}$ θα είναι κενό, ενώ το σύνολο $\Pi_{\neg κλειδί}$ θα περιλαμβάνει όλες τις στήλες Π_1, ..., Π_v):

1) *ο πίνακας Π δεν είναι κανονικοποιημένος και το πεδίο εισαγόμενες_τιμές δεν είναι κενό.*
   Σ' αυτή την περίπτωση, η διαδικασία εκτέλεσης της εντολής *INSERT* χρησιμοποιεί τα

πεδία *όνομα_πίνακα*, *στήλες_εισαγωγής* και *εισαγόμενες_τιμές* για να ανασυνθέσει την αρχική εντολή *INSERT*, στη μορφή που δόθηκε από τον χρήστη. Κατόπιν, η εντολή που σχηματίζεται προωθείται στο Ingres για εκτέλεση. Αν στον πίνακα *Π* έχει ορισθεί μοναδικό κλειδί, η μοναδικότητά του εξασφαλίζεται από τη δομή μοναδικού δείκτη που έχει δημιουργηθεί από τη διαδικασία εκτέλεσης της εντολής *CREATE TABLE*, κατά τη δημιουργία του πίνακα.

2) *ο πίνακας Π δεν είναι κανονικοποιημένος και το πεδίο εισαγόμενες_τιμές είναι κενό.* Αν τα υποπεδία *προσδιορισμοί_αναμόρφωσης* και *προσδιορισμοί_κανονικοποίησης* του πεδίου *υποερώτηση* είναι κενά, τότε η διαδικασία εκτέλεσης της εντολής *INSERT* ανασυνθέτει την αρχική εντολή, στη μορφή που δόθηκε από τον χρήστη και την προωθεί στο Ingres για εκτέλεση. Στην αντίθετη περίπτωση, η επεκταμένη ερώτηση που περιγράφεται από το πεδίο *υποερώτηση* αποτιμάται (σύμφωνα με τον αλγόριθμο που περιγράφηκε για την εκτέλεση επεκταμένων ερωτήσεων) και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο θα συμβολίζουμε με *P1*. Στη συνέχεια, σχηματίζεται η εντολή

INSERT INTO *όνομα_πίνακα* (*στήλες_εισαγωγής*) SELECT * FROM *P1*

η οποία προωθείται στο Ingres για εκτέλεση και ο αλγόριθμος τερματίζει με την καταστροφή του προσωρινού πίνακα *P1*.

3) *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και το πεδίο εισαγόμενες_τιμές δεν είναι κενό.* Στην περίπτωση αυτή, η διαδικασία εκτέλεσης της εντολής *INSERT* δημιουργεί μία *πλειάδα εισαγωγής*, την οποία θα συμβολίζουμε με *πε*. Το σχήμα της πλειάδας *πε* είναι ίδιο με αυτό του πίνακα *Π* και οι τιμές των στηλών της προκύπτουν από τα πεδία *στήλες_εισαγωγής* και *εισαγόμενες_τιμές* (αν κάποια στήλη της πλειάδας εισαγωγής δεν λαμβάνει τιμή από τα πεδία αυτά, τότε παίρνει τιμή *NULL*). Κατόπιν, ορίζεται ένας *δρομέας* (cursor) (ο δρομέας θα συμβολίζεται με *δρομέας_εισαγωγής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* για τις οποίες οι τιμές όλων των στηλών του συνόλου $\Pi_{\text{κλειδί}}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *merges* $\pi\varepsilon.\Pi_{\chi\varepsilon}$ είναι αληθές. Για κάθε μία από τις επιλεγόμενες πλειάδες (την οποία θα συμβολίζουμε με *τρέχουσα_πλειάδα*), προσκομίζεται στη μνήμη του χρονολογικού φλοιού η στήλη $\text{τρέχουσα\_πλειάδα}.\Pi_{\chi\varepsilon}$ και ακολουθούνται τα εξής βήματα:

   i) η τιμή της στήλης $\pi\varepsilon.\Pi_{\chi\varepsilon}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης $merge(\pi\varepsilon.\Pi_{\chi\varepsilon}, \text{τρέχουσα\_πλειάδα}.\Pi_{\chi\varepsilon})$.

ii) η *τρέχουσα_πλειάδα* διαγράφεται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π* WHERE CURRENT OF *δρομέας_εισαγωγής*

της εμφυτευμένης (embedded) SQL.

Όταν προσκομισθούν όλες οι πλειάδες του πίνακα *Π* που πληρούν τα κριτήρια επιλογής του δρομέα *δρομέας_εισαγωγής*, τότε η πλειάδα *πε* προστίθεται στον πίνακα *Π*.

4) *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και το πεδίο εισαγόμενες_τιμές δεν είναι κενό.* Στην περίπτωση αυτή η διαδικασία εκτέλεσης της εντολής *INSERT* δημιουργεί μία *πλειάδα εισαγωγής*, όπως περιγράφεται στην περίπτωση (3). Κατόπιν ορίζεται ένας *δρομέας* (ο δρομέας θα συμβολίζεται με *δρομέας_εισαγωγής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* για τις οποίες οι τιμές όλων των στηλών του συνόλου $Π_{κλειδί}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $π.Π_{χε}$ *merges* $πε.Π_{χε}$ είναι αληθές. Για κάθε μία από τις επιλεγόμενες πλειάδες (την οποία θα συμβολίζουμε με *τρέχουσα_πλειάδα*), προσκομίζονται στη μνήμη του χρονολογικού φλοιού οι στήλες του συνόλου $Π_{-κλειδί}$ και η στήλη $Π_{χε}$ και ακολουθούνται τα εξής βήματα:

i) αν το κατηγόρημα $τρέχουσα\_πλειάδα.Π_{χε}$ *cp* $πε.Π_{χε}$ αληθεύει, τότε η εντολή *INSERT* παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Οι αλλαγές που έχουν επέλθει στον πίνακα *Π* από τη διαδικασία εκτέλεσης της εντολής *INSERT* αναιρούνται και η διαδικασία εκτέλεσης της εντολής τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

ii) αν όλες οι στήλες της πλειάδας *τρέχουσα_πλειάδα* που ανήκουν στο σύνολο $Π_{-κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *πε*, τότε η τιμή της στήλης $πε.Π_{χε}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης *merge*($πε.Π_{χε}$, $τρέχουσα\_πλειάδα.Π_{χε}$) και η τρέχουσα πλειάδα διαγράφεται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π* WHERE CURRENT OF *δρομέας_εισαγωγής*

της εμφυτευμένης SQL.

iii) σε όλες τις άλλες περιπτώσεις, δηλαδή αν το κατηγόρημα $τρέχουσα\_πλειάδα.Π_{χε}$ *cp* $πε.Π_{χε}$ είναι ψευδές και τουλάχιστον μία από τις στήλες της πλειάδας *τρέχουσα_πλειάδα* που ανήκουν στο σύνολο $Π_{-κλειδί}$ έχει τιμή διαφορετική από την αντίστοιχη στήλη της πλειάδας *πε*, η πλειάδα *τρέχουσα_πλειάδα* αγνοείται.

Όταν προσκομισθούν όλες οι πλειάδες του πίνακα *Π* που πληρούν τα κριτήρια επιλογής του δρομέα *δρομέας_εισαγωγής*, τότε η πλειάδα *πε* προστίθεται στον πίνακα *Π*.

5) *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και το πεδίο εισαγόμενες_τιμές είναι κενό.* Σ' αυτή την περίπτωση, η επεκταμένη ερώτηση που περιγράφεται από το πεδίο *υποερώτηση* αποτιμάται (σύμφωνα με τον αλγόριθμο που περιγράφηκε για την εκτέλεση επεκταμένων ερωτήσεων) και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο θα συμβολίζουμε με *P1*. Το σχήμα του πίνακα *P1* θα είναι ίδιο με το σχήμα του πίνακα *Π*, με πιθανή αναδιάταξη των στηλών του αποτελέσματος της ερώτησης σύμφωνα με τα προσδιοριζόμενα στο πεδίο *στήλες_εισαγωγής* ή/και πλήρωση των παραλειπομένων στηλών με στήλες που περιέχουν τιμές *NULL*. Αν ο πίνακας *P1* περιέχει μόνο μία πλειάδα, τότε οι τιμές των στηλών της ανακτώνται, ο πίνακας *P1* καταστρέφεται και ακολουθείται ο αλγόριθμος της περίπτωσης (3). Στην αντίθετη περίπτωση, ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_εισαγωγής*) μέσω του οποίου επιλέγονται οι πλειάδες *π* του πίνακα *Π* για τις οποίες υπάρχει πλειάδα *ρ* στον πίνακα *P1* τέτοια ώστε όλες οι στήλες της πλειάδας *Π* που ανήκουν στο σύνολο $\Pi_{\neg κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *ρ* και το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *merges* $\rho.\Pi_{\chi\varepsilon}$ αληθεύει. Για κάθε επιλεγόμενη πλειάδα (την οποία θα συμβολίζουμε με *τρέχουσα_πλειάδα*) προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της και ακολουθούνται τα εξής βήματα:

   i) η πλειάδα *τρέχουσα_πλειάδα* διαγράφεται από τον πίνακα *Π* μέσω της εντολής

   DELETE FROM *Π* WHERE CURRENT OF *δρομέας_εισαγωγής*

   της εμφυτευμένης SQL.

   ii) η πλειάδα *τρέχουσα_πλειάδα* εισάγεται στον πίνακα *P1*.

Στη συνέχεια, ο πίνακας *P1* συμπτύσσεται πάνω στη στήλη $\Pi_{\chi\varepsilon}$ και οι πλειάδες του αποτελέσματος της πράξη *σύμπτυξη* εισάγονται στον πίνακα *Π*. Ο αλγόριθμος τερματίζει με την καταστροφή του προσωρινού πίνακα *P1*.

6) *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και το πεδίο εισαγόμενες_τιμές είναι κενό.* Σ' αυτή την περίπτωση, η επεκταμένη ερώτηση που περιγράφεται από το πεδίο *υποερώτηση* αποτιμάται (σύμφωνα με τον αλγόριθμο που περιγράφηκε για την εκτέλεση επεκταμένων ερωτήσεων) και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο θα συμβολίζουμε με *P1*. Το σχήμα του πίνακα *P1* θα είναι ίδιο με το σχήμα του πίνακα *Π*, με πιθανή αναδιάταξη των στηλών του αποτελέσματος της ερώτησης σύμφωνα με τα προσδιοριζόμενα στο πεδίο *στήλες_εισαγωγής* ή/και πλήρωση των παραλειπομένων στηλών με στήλες που περιέχουν

τιμές *NULL*. Αν ο πίνακας *P1* περιέχει μόνο μία πλειάδα, τότε οι τιμές των στηλών της ανακτώνται, ο πίνακας *P1* καταστρέφεται και ακολουθείται ο αλγόριθμος της περίπτωσης (4). Στην αντίθετη περίπτωση, ο αλγόριθμος συνεχίζεται ως ακολούθως:

i) ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_εισαγωγής*), ο οποίος επιλέγει τις πλειάδες *(π, ρ)* του αποτελέσματος της σύνδεσης των πινάκων *Π* και *P1*, για τις οποίες όλες οι στήλες της πλειάδας *Π* που ανήκουν στο σύνολο $\Pi_{κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *ρ* και το κατηγόρημα $\pi.\Pi_{χε}$ *merges* $\rho.\Pi_{χε}$ αληθεύει. Για κάθε επιλεγόμενη πλειάδα του αποτελέσματος της σύνδεσης, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της πλειάδας *π*, οι στήλες της πλειάδας *ρ* που ανήκουν στο σύνολο $\Pi_{-κλειδί}$ και η στήλη $\rho.\Pi_{χε}$ και ακολουθούνται τα εξής βήματα:

   a) αν το κατηγόρημα $\pi.\Pi_{χε}$ *cp* $\rho.\Pi_{χε}$ αληθεύει, τότε η εντολή *INSERT* παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Οι αλλαγές που έχει επιφέρει η διαδικασία εκτέλεσης της εντολής *INSERT* στον πίνακα *Π* αναιρούνται, ο πίνακας *P1* καταστρέφεται και η διαδικασία εκτέλεσης της εντολής τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος.

   b) αν όλες οι στήλες της πλειάδας *π*, που ανήκουν στο σύνολο $\Pi_{-κλειδί}$, έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *ρ* και το κατηγόρημα $\pi.\Pi_{χε}$ *adjacent* $\rho.\Pi_{χε}$ αληθεύει, τότε η πλειάδα *π* διαγράφεται από τον πίνακα *Π* μέσω της εντολής

   DELETE FROM *Π* WHERE CURRENT OF *δρομέας_εισαγωγής*

   της εμφυτευμένης SQL και η ίδια πλειάδα εισάγεται στον πίνακα *P1*.

   c) σε όλες τις άλλες περιπτώσεις, δηλαδή αν το κατηγόρημα $\pi.\Pi_{χε}$ *cp* $\rho.\Pi_{χε}$ είναι ψευδές και τουλάχιστον μία από τις στήλες της πλειάδας *π* που ανήκουν στο σύνολο $\Pi_{-κλειδί}$ έχει τιμή διαφορετική από την αντίστοιχη στήλη της πλειάδας *ρ*, η πλειάδα *(π, ρ)* αγνοείται.

ii) ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ελέγχου*), ο οποίος επιλέγει τις πλειάδες *ρ* του πίνακα *P1* ταξινομημένες βάσει των στηλών που περιέχονται στο σύνολο $\Pi_{κλειδί}$ και της στήλης $\Pi_{χε}$, με αυτή τη σειρά. Κατόπιν ακολουθούνται τα εξής βήματα:

   a) για την πρώτη πλειάδα που ανακτάται μέσω του δρομέα *δρομέας_ελέγχου*, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της και αποθηκεύονται στη δομή *πλειάδα_ελέγχου*.

b) για την επόμενη πλειάδα που ανακτάται μέσω του δρομέα *δρομέας_ελέγχου*, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της, αποθηκεύονται στη δομή *νέα_πλειάδα* και διενεργούνται οι ακόλουθοι έλεγχοι:

- αν όλες οι στήλες της δομής *νέα_πλειάδα* που περιέχονται στο σύνολο $\Pi_{κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της δομής *πλειάδα_ελέγχου* και το κατηγόρημα *πλειάδα_ελέγχου*.$\Pi_{χε}$ *cp νέα_πλειάδα*.$\Pi_{χε}$ αληθεύει, τότε η λειτουργία εισαγωγής παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Οι αλλαγές που έχουν επέλθει στον πίνακα *Π* αναιρούνται, ο πίνακας *P1* καταστρέφεται και ο αλγόριθμος τερματίζει επιστρέφοντας έναν κωδικό λάθους.
- αν όλες οι στήλες της δομής *νέα_πλειάδα*, εκτός της στήλης *νέα_πλειάδα*.$\Pi_{χε}$, έχουν τιμές ίσες με τις αντίστοιχες στήλες της δομής *πλειάδα_ελέγχου* και το κατηγόρημα *πλειάδα_ελέγχου*.$\Pi_{χε}$ *adjacent νέα_πλειάδα*.$\Pi_{χε}$ αληθεύει, τότε η στήλη *πλειάδα_ελέγχου*$_{χε}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης *merge(πλειάδα_ελέγχου*.$\Pi_{χε}$, *νέα_πλειάδα*.$\Pi_{χε}$*)* και το βήμα (b) επαναλαμβάνεται.
- σε όλες τις άλλες περιπτώσεις, η πλειάδα *πλειάδα_ελέγχου* εισάγεται στον πίνακα *Π*, οι τιμές των στηλών της δομής *πλειάδα_ελέγχου* αντικαθίστανται από τις αντίστοιχες τιμές της δομής *νέα_πλειάδα* και το βήμα (b) επαναλαμβάνεται έως ότου εξετασθούν όλες οι πλειάδες του πίνακα *P1*.

Πρακτικά, στο βήμα (ii) ενσωματώνονται ο έλεγχος παραβίασης της μοναδικότητας του κλειδιού και η σύμπτυξη των πλειάδων. Ο αλγόριθμος ολοκληρώνεται με την καταστροφή του πίνακα *P1*.

Για τις περιπτώσεις (5) και (6) πρέπει να ληφθεί υπόψη η συμπεριφορά του σχεσιακού ΣΔΒΔ, όταν εισάγονται πλειάδες σε κάποιον πίνακα που μετέχει σε σύνδεση. Αν η εισαγωγή πλειάδων δεν επηρεάζει το αποτέλεσμα της σύνδεσης, τότε οι αλγόριθμοι μπορούν να εφαρμοστούν όπως περιγράφονται στα ανωτέρω. Στην αντίθετη περίπτωση, οι πλειάδες που διαγράφονται από τον πίνακα *Π* πρέπει να εισαχθούν στον πίνακα *P1* όταν έχει ολοκληρωθεί η αποτίμηση της σύνδεσης. Στο ενδιάμεσο χρονικό διάστημα, οι πλειάδες αυτές πρέπει να αποθηκευθούν είτε στη μνήμη του χρονολογικού φλοιού, είτε σε κάποιον προσωρινό πίνακα, αν ο όγκος δεδομένων που περιέχονται σ' αυτές είναι υπερβολικά μεγάλος.

Ένα άλλο σημείο που χρήζει προσοχής στην περίπτωση (6) είναι το αν το σχεσιακό ΣΔΒΔ επιτρέπει τη διαγραφή πλειάδων ενός πίνακα μέσω δρομέα, όταν η πρόταση *FROM* της εντολής *SELECT* που περιέχεται στον ορισμό του δρομέα αναφέρει περισσότερους από έναν πίνακες (το ΣΔΒΔ Sybase υποστηρίζει τη διαγραφή αν ο πίνακας έχει ορισθεί με το χαρακτηριστικό *TIMESTAMP* ([Sybase90])· το ΣΔΒΔ ORACLE επιτρέπει τη διαγραφή αν ο ορισμός του δρομέα αναφέρει σαφώς ότι μόνο ένας πίνακας από τους μετέχοντες στη σύνδεση μπορεί να ενημερωθεί μέσω του δρομέα ([Oracle90])· το ΣΔΒΔ Ingres δεν επιτρέπει τέτοιες διαγραφές ([Ingres91])). Στην περίπτωση που οι διαγραφές δεν επιτρέπονται ή δεν πληρούνται οι κατάλληλες προϋποθέσεις για να υποστηρίζεται η διαγραφή, το βήμα (i) του αλγορίθμου της περίπτωσης (6) τροποποιείται ως ακολούθως:

i) αρχικά σχηματίζεται ο πίνακας *P1* και ορίζεται ένας δρομέας, ο οποίος επιλέγει τις πλειάδες π του πίνακα Π για τις οποίες υπάρχει πλειάδα ρ στον πίνακα *P1* τέτοια ώστε όλες οι στήλες της πλειάδας π που ανήκουν στο σύνολο $\Pi_{κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας ρ και το κατηγόρημα $\pi.\Pi_{χε}$ *merges* $\rho.\Pi_{χε}$ αληθεύει. Κάθε επιλεγόμενη πλειάδα διαγράφεται από τον πίνακα Π και εισάγεται στον πίνακα *P1*.

Η τροποποιημένη μορφή του αλγορίθμου παρουσιάζει το μειονέκτημα ότι μερικές από τις πλειάδες του πίνακα Π, οι οποίες δεν μπορούν να συμπτυχθούν με πλειάδες του πίνακα *P1* (διότι κάποια από τις στήλες της πλειάδας π που ανήκουν στο σύνολο $\Pi_{-κλειδί}$ έχει διαφορετική τιμή από την αντίστοιχη στήλη στον πίνακα *P1*) μετακινούνται από τον πίνακα Π στον πίνακα *P1*. Το πλήθος των πλειάδων που θα μετακινηθούν άσκοπα θα είναι το πολύ ίσο με το διπλάσιο του αριθμού των πλειάδων του πίνακα *P1*.

V. Η εντολή *DELETE*.

Η διαδικασία εκτέλεσης της εντολής *INSERT* αρχικά ελέγχει αν ο πίνακας που αναφέρεται στο πεδίο *όνομα_πίνακα* (βλέπε σελ. 65, *Η εντολή DELETE.*) είναι κανονικοποιημένος ή όχι, εξετάζοντας τον πίνακα *vtsql_norms*. Κατόπιν διακρίνονται οι ακόλουθες περιπτώσεις (στα επόμενα ο πίνακας από τον οποίο θα διαγραφούν τα δεδομένα θα συμβολίζεται με Π. Το σχήμα του πίνακα Π θα συμβολίζεται με Π_1, ..., $\Pi_ν$, $\Pi_{χε}$, όπου $\Pi_{χε}$ είναι η στήλη κανονικοποίησης του πίνακα (αν ο πίνακας δεν είναι κανονικοποιημένος, τότε η στήλη $\Pi_{χε}$ δεν θεωρείται μέρος του σχήματος του πίνακα):

1) αν το πεδίο *προσδιορισμός_τμημάτων* είναι κενό, τότε η εντολή προωθείται απευθείας στο Ingres για εκτέλεση, ανεξάρτητα από το αν ο πίνακας Π είναι κανονικοποιημένος ή όχι και η διαδικασία εκτέλεσης της εντολής *DELETE* τερματίζει.

2) αν το πεδίο *προσδιορισμός_τμημάτων* δεν είναι κενό και ο πίνακας *Π* δεν είναι κανονικοποιημένος, τότε προκύπτει σημασιολογικό σφάλμα, καθώς δεν επιτρέπεται η χρήση της πρότασης *PORTION* σε μη κανονικοποιημένους πίνακες. Η διαδικασία εκτέλεσης της εντολής *DELETE* αποτυγχάνει, επιστρέφοντας έναν κωδικό λάθους. Αντίστοιχη αποτυχία της διαδικασίας εκτέλεσης και επιστροφή κωδικού λάθους έχουμε αν στο πεδίο *προσδιορισμός_τμημάτων* περιέχονται περισσότερα από ένα ζεύγη της μορφής *(στήλη, έκφραση)* ή αν η *στήλη* που αναφέρεται στο μοναδικό ζεύγος που περιέχεται στη δομή *προσδιορισμός_τμημάτων* δεν είναι η στήλη $\Pi_{\chi\varepsilon}$.

3) ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_διαγραφής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* οι οποίες πληρούν τη συνθήκη που περιέχεται στο πεδίο *συνθήκη* και για τις οποίες το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *cp έκφραση* αληθεύει (*έκφραση* είναι ο δεύτερος όρος του μοναδικού ζεύγους που περιέχεται στη δομή *προσδιορισμός_τμημάτων*). Για κάθε επιλεγόμενη πλειάδα προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλα τα πεδία της, καθώς και η τιμή της έκφρασης *έκφραση* και ακολουθούνται τα κάτωθι βήματα:

   i) αν το κατηγόρημα *έκφραση supinterv* $\pi.\Pi_{\chi\varepsilon}$ αληθεύει, τότε η πλειάδα *π* διαγράφεται από τον πίνακα *Π* μέσω της εντολής

   DELETE FROM *Π* WHERE CURRENT OF *δρομέας_διαγραφής*

   της εμφυτευμένης SQL.

   ii) αν τα χρονικά σημεία που περιέχονται στη στήλη $\pi.\Pi_{\chi\varepsilon}$ και δεν περιέχονται στην έκφραση *έκφραση* είναι συνεχόμενα, και άρα μπορούν να αναπαρασταθούν με ένα μόνο χρονικό διάστημα έστω δ_1, τότε η τιμή της στήλης $\Pi_{\chi\varepsilon}$ της πλειάδας *π* τίθεται ίση με δ_1, χρησιμοποιώντας την εντολή

   UPDATE *Π* SET $\Pi_{\chi\varepsilon} = \delta_1$ WHERE CURRENT OF *δρομέας_διαγραφής*

   της εμφυτευμένης SQL.

   iii) αν τα χρονικά σημεία που περιέχονται στη στήλη $\pi.\Pi_{\chi\varepsilon}$ και δεν περιέχονται στην έκφραση *έκφραση* δεν είναι συνεχόμενα, με αποτέλεσμα να απαιτούνται *δύο* χρονικά διαστήματα για την αναπαράστασή τους, έστω δ_1 και δ_2, τότε τιμή της στήλης $\Pi_{\chi\varepsilon}$ της πλειάδας *π* τίθεται ίση με δ_1, όπως στην περίπτωση (ii) και μία νέα πλειάδα εισάγεται στον πίνακα *Π*, της οποίας οι στήλες Π_1, …, Π_v έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *π*, ενώ η στήλη $\Pi_{\chi\varepsilon}$ έχει τιμή ίση με δ_2.

Για την υλοποίηση της εντολής *DELETE* πρέπει να ληφθεί υπόψη η συμπεριφορά του ΣΔΒΔ, σε σχέση με την ανάκτηση πλειάδων μέσω δρομέων: αν ένας δρομέας επιλέγει πλειάδες που

έχουν εισαχθεί στον πίνακα *μετά* το άνοιγμά του, τότε όλες οι εισαγωγές πλειάδων που ορίζονται από την περίπτωση (iii) του βήματος (3) πρέπει να γίνουν μόνον αφότου κλεισθεί ο δρομέας *δρομέας_διαγραφής*, αλλιώς τα αποτελέσματα θα είναι απρόβλεπτα. Μία περίπτωση ανεπιθύμητης συμπεριφοράς παρουσιάζεται στο ακόλουθο παράδειγμα: ας υποθέσουμε ότι υπάρχει ο πίνακας *Ιεράρχης*, με σχήμα *(Όνομα, Διάστημα)* και ότι περιέχει τις πλειάδες που φαίνονται στο σχήμα 4.4.

Όνομα	Διάστημα
Βασίλειος	[330, 380)
Γρηγόριος	[326, 391)
Ιωάννης	[344, 508)

Σχήμα 4.4 - Ο πίνακας *Ιεράρχης*.

Θέλουμε να διαγράψουμε την πληροφορία για τη δεκαετία γύρω από το μέσον της ζωής του Βασιλείου, οπότε δίνουμε την εντολή

```
DELETE FROM Ιεράρχης
PORTION PERIOD = interv(succ(middle(Διάστημα), -5), succ(middle(Διάστημα), 5))
WHERE NAME = 'Βασίλειος'
```

Σύμφωνα με τον ανωτέρω αλγόριθμο, η πλειάδα που περιέχει δεδομένα για τον Βασίλειο θα επιλεχθεί από τον δρομέα *δρομέας_διαγραφής*, θα προσκομισθεί στη μνήμη του χρονολογικού φλοιού, η τιμή του πεδίου *Διάστημα* θα τροποποιηθεί σε *[330, 350)* και θα εισαχθεί μία νέα πλειάδα *(Βασίλειος, [360, 380))*. Η πλειάδα που θα εισαχθεί, πληρεί τις προϋποθέσεις επιλογής της από τον δρομέα *δρομέας_διαγραφής*, οπότε θα εφαρμοστεί και σ' αυτήν ο αλγόριθμος διαγραφής, με αποτέλεσμα να τροποποιηθεί η τιμή *[360, 380)* σε *[360, 365)* και να εισαχθεί μία νέα πλειάδα *(Βασίλειος, [375, 380))*. Ο αλγόριθμος διαγραφής θα εφαρμοστεί ξανά για την τελευταία πλειάδα, κ.ο.κ.

Λαμβάνοντας υπόψη αυτό το πρόβλημα, αν η σημασιολογία των δρομέων του σχεσιακού ΣΔΒΔ ορίζει ότι ένας δρομέας επιλέγει *μόνο* πλειάδες που είχαν εισαχθεί στον πίνακα *πριν* ο δρομέας ανοιχθεί, τότε η εισαγωγές των πλειάδων που ορίζονται από την περίπτωση (iii) του βήματος (3) μπορούν να γίνουν όταν το βήμα αυτό εκτελείται. Στην αντίθετη περίπτωση, οι πλειάδες πρέπει να εισαχθούν μόνο αφότου ο δρομέας έχει κλεισθεί (δηλαδή όταν έχει τερματιστεί η εκτέλεση του βήματος (3)). Ο ετεροχρονισμός αυτός μπορεί να συνεπάγεται τη δημιουργία ενός προσωρινού πίνακα, στον οποίο θα αποθηκευθούν οι προς εισαγωγή

πλειάδες, αν ο όγκος δεδομένων είναι υπερβολικά μεγάλος για να αποθηκευθεί στη μνήμη του χρονολογικού φλοιού.

VI. Η εντολή *UPDATE*.

Η διαδικασία εκτέλεσης της εντολής *UPDATE* αρχικά ελέγχει αν ο πίνακας που αναφέρεται στο πεδίο *όνομα_πίνακα* (βλέπε σελ. 66, *Η εντολή UPDATE.*)είναι κανονικοποιημένος ή όχι, εξετάζοντας τον πίνακα *vtsql_norms*. Κατόπιν διακρίνονται οι ακόλουθες περιπτώσεις (στα επόμενα ο πίνακας στον οποίο θα εισαχθούν τα δεδομένα θα συμβολίζεται με *Π*. Το σχήμα του πίνακα *Π* θα συμβολίζεται με $Π_1$, ..., $Π_ν$, $Π_{χε}$, όπου $Π_{χε}$ είναι η στήλη κανονικοποίησης του πίνακα (αν ο πίνακας δεν είναι κανονικοποιημένος, τότε η στήλη $Π_{χε}$ δεν θεωρείται μέρος του σχήματος του πίνακα). Οι στήλες του πίνακα $Π_1$ έως $Π_ν$ χωρίζονται σε δύο ξένα μεταξύ τους σύνολα, τα $Π_{κλειδί}$ και $Π_{\neg κλειδί}$, το πρώτο από τα οποία περιέχει τις στήλες που ανήκουν στο πρωτεύον κλειδί, ενώ το δεύτερο αυτές που δεν μετέχουν. Αν δεν έχει ορισθεί πρωτεύον κλειδί πάνω στον πίνακα *Π*, το σύνολο $Π_{κλειδί}$ θα είναι κενό, ενώ το σύνολο $Π_{\neg κλειδί}$ θα περιλαμβάνει όλες τις στήλες $Π_1$, ..., $Π_ν$):

1) *ο πίνακας Π δεν είναι κανονικοποιημένος*. Στην περίπτωση αυτή ελέγχεται αν το πεδίο *προσδιορισμός_τμημάτων* είναι κενό. Αν ο έλεγχος αποτύχει, τότε προκύπτει σημασιολογικό σφάλμα, καθώς δεν επιτρέπεται η χρήση της πρότασης *PORTION* σε μη κανονικοποιημένους πίνακες· η διαδικασία εκτέλεσης της εντολής *UPDATE* αποτυγχάνει, επιστρέφοντας έναν κωδικό λάθους. Στην αντίθετη περίπτωση, η αρχική εντολή ανασυντίθεται, στη μορφή που δόθηκε από τον χρήστη και προωθείται στο Ingres προς εκτέλεση, τερματίζοντας τη διαδικασία εκτέλεσης της εντολής *UPDATE*.
2) *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί κλειδί πάνω σ' αυτόν και το πεδίο προσδιορισμός_τμημάτων είναι κενό*. Σ' αυτή την περίπτωση ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ενημέρωσης*), ο οποίος επιλέγει τις πλειάδες του πίνακα *Π* οι οποίες πληρούν τη συνθήκη που περιέχεται στο πεδίο *συνθήκη*. Για κάθε επιλεγόμενη πλειάδα, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της. Αν κάποια στήλη αναφέρεται ως πρώτος όρος ενός ζεύγους *(στήλη, έκφραση)* της δομής *εκχωρήσεις*, τότε αντί της πραγματικής τιμής της προσκομίζεται η τιμή της έκφρασης *έκφραση* του ίδιου ζεύγους. Η επιλεχθείσα πλειάδα διαγράφεται από τον πίνακα με μία εντολή

DELETE FROM *Π* WHERE CURRENT OF *δρομέας_ενημέρωσης*

της εμφυτευμένης SQL και μία πλειάδα που αποτελείται από τις προσκομισθείσες τιμές εισάγεται σε έναν προσωρινό πίνακα *P1*. Όταν τα δεδομένα που μπορούν να προσκομισθούν μέσω του δρομέα *δρομέας_ενημέρωσης* εξαντληθούν, τότε οι πλειάδες που περιέχονται στον πίνακα *P1* εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (5) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

3) *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί κλειδί πάνω σ' αυτόν και το πεδίο προσδιορισμός_τμημάτων δεν είναι κενό*. Σ' αυτή την περίπτωση αρχικά ελέγχεται ότι το πεδίο *προσδιορισμός_τμημάτων* περιέχει μόνο ένα ζεύγος της μορφής *(στήλη, έκφραση)* και ότι η *στήλη* που αναφέρεται στο μοναδικό ζεύγος που περιέχεται στη δομή *προσδιορισμός_τμημάτων* είναι η στήλη $\Pi_{\chi\varepsilon}$. Αν οποιοσδήποτε από τους δύο ελέγχους αποτύχει, τότε η διαδικασία εκτέλεσης της εντολής *UPDATE* τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους. Στη συνέχεια, ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ενημέρωσης*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* οι οποίες πληρούν τη συνθήκη που περιέχεται στο πεδίο *συνθήκη* και για τις οποίες το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *cp έκφραση* αληθεύει (*έκφραση* είναι ο δεύτερος όρος του μοναδικού ζεύγους που περιέχεται στη δομή *προσδιορισμός_τμημάτων*). Για κάθε επιλεγόμενη πλειάδα, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλα τα πεδία της, οι τιμές όλων των *εκφράσεων* που εμφανίζονται ως δεύτεροι όροι στα ζεύγη της δομής *εκχωρήσεις*, καθώς και η τιμή της έκφρασης *έκφραση* που εμφανίζεται ως δεύτερος όρος του μοναδικού ζεύγους που περιέχεται στη δομή *προσδιορισμός_τμημάτων* και ακολουθούνται τα κάτωθι βήματα:

   i) εκτελείται το βήμα (3) του αλγορίθμου διαγραφής, προκειμένου να απομακρυνθεί από τον πίνακα *Π* η πληροφορία που αφορά τον χρόνο εγκυρότητας που προσδιορίζεται από την πρόταση *PORTION*.

   ii) εισάγεται σε έναν προσωρινό πίνακα *P1* μία πλειάδα, οι τιμές των στηλών της οποίας καθορίζονται σύμφωνα με τα ακόλουθα κριτήρια:

      a) αν το όνομα της στήλης εμφανίζεται ως πρώτος όρος σε ένα ζεύγος *(στήλη, έκφραση)* της δομής *εκχωρήσεις*, τότε η τιμή της συγκεκριμένης στήλης στη νέα πλειάδα θα είναι ίση με την τιμή της έκφρασης *έκφραση*, του δεύτερου δηλαδή όρου του ζεύγους.

b) για τις υπόλοιπες στήλες, χρησιμοποιείται η τιμή της αντίστοιχης στήλης στην πλειάδα *π*, εκτός από τη στήλη $\Pi_{\chi\varepsilon}$, της οποίας η τιμή τίθεται ίση με την τιμή του δεύτερου όρου του μοναδικού ζεύγους που περιέχεται στη δομή *προσδιορισμός_τμημάτων*.

Όταν τα δεδομένα που μπορούν να προσκομισθούν μέσω του δρομέα *δρομέας_ενημέρωσης* εξαντληθούν, τότε οι πλειάδες που περιέχονται στον πίνακα *P1* εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (5) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

4) *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί κλειδί πάνω σ' αυτόν και το πεδίο προσδιορισμός_τμημάτων είναι κενό*. Στην περίπτωση αυτή εφαρμόζεται ο αλγόριθμος που περιγράφηκε για την περίπτωση (2), τροποποιημένος ώστε οι πλειάδες του πίνακα *P1* να εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (6) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

5) *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί κλειδί πάνω σ' αυτόν και το πεδίο προσδιορισμός_τμημάτων δεν είναι κενό*. Στην περίπτωση αυτή εφαρμόζεται ο αλγόριθμος που περιγράφηκε για την περίπτωση (3), τροποποιημένος ώστε οι πλειάδες του πίνακα *P1* να εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (6) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

VII. Η εντολή *CREATE INDEX*.

Η διαδικασία εκτέλεσης της εντολής *CREATE INDEX* ανασυνθέτει από τα πεδία *όνομα_δείκτη*, *όνομα_πίνακα*, *προσδιορισμός_μοναδικότητας* και *στήλες_δείκτη* (βλέπε σελ. 66, *Η εντολή CREATE INDEX*.) την εντολή, στη μορφή που δόθηκε από τον χρήστη και την προωθεί προς εκτέλεση στο Ingres.

VIII. Η εντολή *DROP INDEX*.

Η διαδικασία εκτέλεσης της εντολής *DROP INDEX* ανασυνθέτει από το πεδίο *όνομα_δείκτη* (βλέπε σελ. 67, *Η εντολή DROP INDEX*.) την εντολή, στη μορφή που δόθηκε από τον χρήστη και την προωθεί στο Ingres για εκτέλεση.

IX. Η εντολή *HELP*.

Η διαδικασία εκτέλεσης της εντολής *HELP* εξετάζει αν το πεδίο *όνομα_αντικειμένου* (βλέπε σελ. 67, *Η εντολή HELP.*) είναι κενό ή όχι. Στην πρώτη περίπτωση ανακτώνται και τυπώνονται τα ονόματα των πινάκων οι οποίοι είτε ανήκουν στον χρήστη είτε έχει δοθεί σ' αυτόν δικαίωμα προσπέλασής τους. Στη δεύτερη περίπτωση, τυπώνονται πληροφορίες για το σχήμα του προσδιοριζόμενου αντικειμένου, συμπεριλαμβανομένων των στηλών κανονικοποίησης και των πρωτευόντων κλειδιών, αν έχουν ορισθεί.

X. Οι διαφεύγουσες εντολές.

Η διαδικασία εκτέλεσης των διαφευγουσών εντολών ελέγχει αρχικά αν η διαφεύγουσα εντολή είναι μία από τις *DROP*, *SELECT, INSERT, DELETE, UPDATE* και *HELP*. Αν ο έλεγχος δείξει ότι δεν πρόκειται για κάποια από αυτές τις εντολές, τότε η διαφεύγουσα εντολή προωθείται προς εκτέλεση στο Ingres, ενώ στην αντίθετη περίπτωση η διαδικασία εκτέλεσης των διαφευγουσών εντολών τερματίζεται επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

4.4. Απόδοση των βελτιστοποιημένων αλγορίθμων.

Οι αλγόριθμοι υλοποίησης των εντολών *INSERT*, *DELETE* και *UPDATE* που περιγράφηκαν στις προηγούμενες παραγράφους είναι *βελτιστοποιημένοι*, με την έννοια ότι αξιοποιούν τις δυνατότητες που προσφέρει η εμφυτευμένη SQL προκειμένου να επιτύχουν βελτιωμένη απόδοση. Για την εκτέλεση των εντολών αυτών, είχαν χρησιμοποιηθεί στο παρελθόν (στα πλαίσια του προγράμματος ORES) *αλγεβρικοί αλγόριθμοι*[5], οι οποίοι βασιζόταν στις πράξεις της ΣΑΧΕ, καθώς και σε εντολές SQL. Στις επόμενες παραγράφους περιγράφονται οι αλγεβρικοί αλγόριθμοι, παρουσιάζεται ένα μαθηματικό μοντέλο κόστους εκτέλεσης των εντολών, βάσει του οποίου συγκρίνονται οι δύο υλοποιήσεις και, τέλος, παρατίθενται συγκριτικά διαγράμματα απόδοσης των αλγεβρικών και των βελτιστοποιημένων αλγορίθμων.

4.4.1. Οι αλγεβρικοί αλγόριθμοι.

I. Η εντολή *INSERT*.

Για την εισαγωγή δεδομένων σε έναν κανονικοποιημένο πίνακα *Π*, του οποίου το σχήμα είναι (Π_1, ..., Π_v, $\Pi_{\chi\varepsilon}$), όπου $\Pi_{\chi\varepsilon}$ είναι η στήλη κανονικοποίησης του πίνακα (η εισαγωγή

---

[5] Ο όρος *αλγεβρικοί* δεν χρησιμοποιείται εδώ με την αυστηρή του έννοια, δεδομένου ότι στη ΣΑΧΕ δεν ορίζονται πρωτεύοντα κλειδιά, δεν υπάρχουν διπλότυπες πλειάδες κ.λπ. Οι *αλγεβρικοί αλγόριθμοι* αντιστοιχούν σε μεθόδους υλοποίησης των εντολών, οι οποίες βασίζονται στις πράξεις της ΣΑΧΕ και εντολές SQL.

δεδομένων σε μη κανονικοποιημένο πίνακα αντιμετωπίζεται άμεσα από το σχεσιακό ΣΔΒΔ, οπότε η περίπτωση αυτή δεν χρήζει ανάλυσης), δημιουργείται ένας προσωρινός πίνακας *P1*, ο οποίος θα περιέχει τα προς εισαγωγή δεδομένα. Αν τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω της πρότασης *VALUES*, τότε ο πίνακας *P1* θα περιέχει μία μόνο πλειάδα, οι τιμές των στηλών της οποίας θα είναι αυτές που παρατίθενται στην πρόταση *VALUES*. Αν τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω μιας επεκταμένης ερώτησης, τότε ο πίνακας *P1* θα περιέχει το αποτέλεσμα της επεκταμένης ερώτησης. Στη συνέχεια διακρίνονται δύο περιπτώσεις:

1) *στον πίνακα Π δεν έχει ορισθεί πρωτεύον κλειδί*: οι πλειάδες του πίνακα *Π* που μπορούν να συμπτυχθούν με οποιαδήποτε πλειάδα του πίνακα *P1* αποθηκεύονται σε έναν προσωρινό πίνακα *P2*. Αυτό επιτυγχάνεται μέσω της εντολής

CREATE TABLE *P2* AS
SELECT * FROM *Π*
WHERE EXISTS (SELECT * FROM *P1*
WHERE $\Pi.\Pi_1 = P1.\Pi_1$ AND
… AND
$\Pi.\Pi_v = P1.\Pi_v$ AND
$cp(\Pi.\Pi_{\chi\varepsilon}, P1.\Pi_{\chi\varepsilon}) = 1$

Ακολούθως, οι πλειάδες αυτές διαγράφονται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π*
WHERE EXISTS (SELECT * FROM *P2*
WHERE $\Pi.\Pi_1 = P2.\Pi_1$ AND
… AND
$\Pi.\Pi_v = P2.\Pi_v$ AND
$\Pi.\Pi_{\chi\varepsilon} = P2.\Pi_{\chi\varepsilon}$)

Τέλος, εφαρμόζεται η πράξη $P3 = P1\ PUNION\ [\Pi_{\chi\varepsilon}]\ P2$ και οι πλειάδες του πίνακα *P3* εισάγονται στον πίνακα *Π*. Ο αλγόριθμος εισαγωγής ολοκληρώνεται με την καταστροφή των προσωρινών πινάκων *P1*, *P2* και *P3*.

2) *έχει ορισθεί κλειδί πάνω στον πίνακα Π (το οποίο υποθέτουμε ότι αποτελείται από τις στήλες* $\Pi_1, ..., \Pi_\kappa, \Pi_{\chi\varepsilon}$). Στην περίπτωση αυτή αρχικά πιστοποιείται ότι ο πίνακας *P1* δεν περιέχει κάποιο ζεύγος διαφορετικών πλειάδων *(χ, ψ)* τέτοιες ώστε $\chi.\Pi_\iota = \psi.\Pi_\iota\ \forall\ \iota = 1, ..., \kappa$ και $\chi.\Pi_{\chi\varepsilon}\ cp\ \psi.\Pi_{\chi\varepsilon}$. Ο έλεγχος αυτός διενεργείται σε δύο φάσεις:

i) στην πρώτη φάση πιστοποιείται ότι ο πίνακας *P1* δεν περιέχει κάποιο ζεύγος διαφορετικών πλειάδων *(χ, ψ)* τέτοιες ώστε $\chi.\Pi_\iota = \psi.\Pi_\iota \ \forall\ \iota = 1, \ldots, \kappa$ και $\chi.\Pi_{\chi\varepsilon} = \psi.\Pi_{\chi\varepsilon}$. Αυτό επιτυγχάνεται προωθώντας την εντολή

CREATE TABLE *P2* AS SELECT DISTINCT $\Pi_1, \ldots, \Pi_\kappa, \Pi_{\chi\varepsilon}$

FROM *P1*

στο σχεσιακό ΣΔΒΔ και συγκρίνοντας τον αριθμό πλειάδων που περιέχονται στους πίνακες *P1* και *P2*. Αν ο πίνακας *P2* περιέχει λιγότερες πλειάδες από τον πίνακα *P1*, τότε ο πίνακας *P1* περιέχει πλειάδες με ίσες τιμές στα πεδία που συμμετέχουν στο κλειδί. Οι πίνακες *P1* και *P2* καταστρέφονται και η διαδικασία εισαγωγής τερματίζεται.

ii) στη δεύτερη φάση, ελέγχεται ότι ο πίνακας *P1* δεν περιέχει κάποιο ζεύγος διαφορετικών πλειάδων *(χ, ψ)* τέτοιες ώστε $\chi.\Pi_\iota = \psi.\Pi_\iota \ \forall\ \iota = 1, \ldots, \kappa$ και $\chi.\Pi_{\chi\varepsilon} \neq \psi.\Pi_{\chi\varepsilon}$ και $\chi.\Pi_{\chi\varepsilon}$ *cp* $\psi.\Pi_{\chi\varepsilon}$. Αυτό επιτυγχάνεται εκτελώντας την εντολή

SELECT count(*) FROM *P1*, *P1 tmp*

WHERE $P1.\Pi_1 = tmp.\Pi_1$ AND

... AND

$P1.\Pi_\kappa = tmp.\Pi_\kappa$ AND

$cp(P1.\Pi_{\chi\varepsilon}, tmp.\Pi_{\chi\varepsilon}) = 1$ AND

$P1.\Pi_{\chi\varepsilon} <> tmp.\Pi_{\chi\varepsilon}$

Αν το αποτέλεσμα είναι μη μηδενικό, τότε ο πίνακας *P1* περιέχει κάποιο ζεύγος διαφορετικών πλειάδων με ίσες τιμές στις στήλες Π_1 έως Π_κ και επικαλυπτόμενες τιμές στη στήλη $\Pi_{\chi\varepsilon}$. Ο πίνακας *P1* καταστρέφεται και η διαδικασία εισαγωγής τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος.

Ακολούθως οι πλειάδες π του πίνακα *Π* για τις οποίες υπάρχει πλειάδα ρ του πίνακα *P1* τέτοια ώστε $\pi.\Pi_\iota = \rho.\Pi_\iota \ \forall\ \iota = 1, \ldots, \kappa$ και $\pi.\Pi_{\chi\varepsilon}$ *merges* $\rho.\Pi_{\chi\varepsilon}$, αποθηκεύονται σε έναν προσωρινό πίνακα *P3*. Αυτό επιτυγχάνεται μέσω της εντολής

CREATE TABLE *P3* AS

SELECT * FROM *Π*

WHERE EXISTS (SELECT * FROM *P1*

WHERE *Π*.Π_1 = *P1*.Π_1 AND

… AND

Π.Π_κ = *P1*.Π_κ AND

cp(*Π*.$\Pi_{\chi\varepsilon}$, *P1*.$\Pi_{\chi\varepsilon}$) = 1

Οι πλειάδες αυτές διαγράφονται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π*

WHERE EXISTS (SELECT * FROM *P3*

WHERE *Π*.Π_1 = *P3*.Π_1 AND

… AND

Π.Π_ν = *P3*.Π_ν AND

Π.$\Pi_{\chi\varepsilon}$ = *P3*.$\Pi_{\chi\varepsilon}$)

Στη συνέχεια πιστοποιείται ότι στον πίνακα *P1* δεν υπάρχει πλειάδα χ τέτοια ώστε να υπάρχει στον πίνακα *P3* πλειάδα ψ με $\chi.\Pi_\iota = \psi.\Pi_\iota \ \forall\ \iota = 1, \ldots, \kappa$ και $\chi.\Pi_{\chi\varepsilon}$ *cp* $\psi.\Pi_{\chi\varepsilon}$. Αυτό επιτυγχάνεται εκτελώντας την εντολή

SELECT count(*) FROM *P1*, *P3*

WHERE *P1*.Π_1 = *P3*.Π_1 AND

… AND

P1.Π_κ = *P3*.Π_κ AND

cp(*P1*.$\Pi_{\chi\varepsilon}$, *P3*.$\Pi_{\chi\varepsilon}$) = 1

και ελέγχοντας την τιμή του αποτελέσματος. Αν το αποτέλεσμα είναι μη μηδενικό, τότε η πράξη της εισαγωγής οδηγεί σε παραβίαση της μοναδικότητας του πρωτεύοντος κλειδιού. Οι αλλαγές που έχουν επέλθει στον πίνακα *Π* αναιρούνται, οι πίνακες *P1* και *P3* καταστρέφονται και η διαδικασία εισαγωγής τερματίζεται, επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος.

Τέλος, εφαρμόζεται η πράξη *P4 = P1 PUNION [*$\Pi_{\chi\varepsilon}$*] P3* και οι πλειάδες του πίνακα *P4* εισάγονται στον πίνακα *Π*. Ο αλγόριθμος εισαγωγής ολοκληρώνεται με την καταστροφή των προσωρινών πινάκων *P1*, *P3* και *P4*.

II. Η εντολή *DELETE*.

Για τη διαγραφή δεδομένων από έναν κανονικοποιημένο πίνακα *Π*, μέσω μιας εντολής *DELETE* που περιέχει την πρόταση *PORTION* ακολουθούνται τα κάτωθι βήματα (η διαγραφή δεδομένων με εντολές *DELETE* που δεν περιέχουν την πρόταση *PORTION* αντιμετωπίζεται άμεσα από το σχεσιακό ΣΔΒΔ, οπότε η περίπτωση αυτή χρήζει ανάλυσης). Το σχήμα του πίνακα *Π* είναι (Π_1, ..., Π_v, $\Pi_{\chi\varepsilon}$), όπου $\Pi_{\chi\varepsilon}$ είναι η στήλη κανονικοποίησης του πίνακα.):

1. δημιουργείται ένας προσωρινός πίνακας *P1*, ο οποίος περιέχει τις πλειάδες του πίνακα *Π*, οι οποίες θα επηρεαστούν από την εντολή διαγραφής. Αυτό επιτυγχάνεται προωθώντας στο σχεσιακό ΣΔΒΔ την εντολή

CREATE TABLE *P1* AS
SELECT * FROM *Π*
WHERE (*συνθήκη*) AND
cp($\Pi_{\chi\varepsilon}$, *έκφραση*) = 1

όπου *συνθήκη* είναι η συνθήκη που εμφανίζεται στην πρόταση *WHERE* της εντολής *DELETE* που έδωσε ο χρήστης και *έκφραση* είναι το δεξί μέλος της εκχώρησης $\Pi_{\chi\varepsilon}$ = *έκφραση* στην πρόταση *PORTION*. Ακολούθως, οι επιλεχθείσες πλειάδες διαγράφονται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π*
WHERE EXISTS (SELECT * FROM *P1*
WHERE *Π*.Π_1 = *P1*.Π_1 AND
... AND
Π.Π_v = *P1*.Π_v AND
Π.$\Pi_{\chi\varepsilon}$ = *P1*.$\Pi_{\chi\varepsilon}$)

2. δημιουργείται ένας δεύτερος προσωρινός πίνακας *P2*, ο οποίος είναι όμοιος με τον πίνακα *P1*, εκτός της στήλης $\Pi_{\chi\varepsilon}$, η τιμή της οποίας θα είναι ίση με το κοινό τμήμα της τρέχουσας τιμής της και της έκφρασης *έκφραση*, που βρίσκεται στο δεξί μέλος της εκχώρησης $\Pi_{\chi\varepsilon}$ = *έκφραση* στην πρόταση *PORTION*. Ο πίνακας *P2* δημιουργείται με την εντολή

CREATE TABLE *P2* AS
SELECT Π_1, ..., Π_v, $\Pi_{\chi\varepsilon}$ = intervsect($\Pi_{\chi\varepsilon}$, *έκφραση*)
FROM *P1*

3. εκτελείται η πράξη *P3 = P1 PEXCEPT[$\Pi_{\chi\varepsilon}$] P2*, οι πλειάδες του πίνακα *P3* εισάγονται στον πίνακα *Π* και ο αλγόριθμος ολοκληρώνεται με την καταστροφή των προσωρινών πινάκων *P1*, *P2* και *P3*.

III. Η εντολή *UPDATE*.

Για την ενημέρωση των δεδομένων ενός κανονικοποιημένου πίνακα *Π*, διακρίνονται οι κάτωθι περιπτώσεις (η ενημέρωση δεδομένων σε μη κανονικοποιημένους πίνακες αντιμετωπίζεται άμεσα από το σχεσιακό ΣΔΒΔ, οπότε η περίπτωση αυτή δεν χρήζει ανάλυσης). Το σχήμα του πίνακα *Π* είναι (Π_1, ..., Π_ν, $\Pi_{\chi\varepsilon}$), όπου $\Pi_{\chi\varepsilon}$ είναι η στήλη κανονικοποίησης του πίνακα. Αν έχει ορισθεί κλειδί πάνω στον πίνακα *Π*, τότε αυτό αποτελείται από τις στήλες Π_1, ..., Π_κ, $\Pi_{\chi\varepsilon}$.):

1) *δεν χρησιμοποιείται η πρόταση PORTION*. Σ' αυτή την περίπτωση, για την εκτέλεση της εντολής *UPDATE* ακολουθούνται τα εξής βήματα:

   i) δημιουργείται ένας προσωρινός πίνακας *P1*, ο οποίος περιέχει τις πλειάδες του πίνακα *Π* οι οποίες θα επηρεαστούν από την ενημέρωση. Αυτό επιτυγχάνεται προωθώντας στο σχεσιακό ΣΔΒΔ την εντολή

   CREATE TABLE *P1* AS
   SELECT * FROM *Π*
   WHERE (*συνθήκη*)

   όπου συνθήκη είναι η συνθήκη που εμφανίζεται στην πρόταση *WHERE* της εντολής *UPDATE* που έδωσε ο χρήστης. Ακολούθως, οι επιλεχθείσες πλειάδες διαγράφονται από τον πίνακα Π μέσω της εντολής

   DELETE FROM *Π*
   WHERE EXISTS (SELECT * FROM *P1*
   WHERE *Π*.Π_1 = *P1*.Π_1 AND
   ... AND
   Π.Π_ν = *P1*.Π_ν AND
   Π.$\Pi_{\chi\varepsilon}$ = *P1*.$\Pi_{\chi\varepsilon}$)

   ii) οι πλειάδες του πίνακα *P1* ενημερώνονται, όπως ορίζεται από την πρόταση *SET* της εντολής *UPDATE*.

   iii) οι πλειάδες του πίνακα *P1* εισάγονται στον πίνακα *Π*, ακολουθώντας τον αλγεβρικό αλγόριθμο εισαγωγής δεδομένων (βλέπε *Η εντολή INSERT.*, σελ. 93).

2) *η πρόταση PORTION χρησιμοποιείται.* Στην περίπτωση αυτή, για την εκτέλεση της εντολής *UPDATE* ακολουθούνται τα εξής βήματα:

i) οι πλειάδες του πίνακα *Π* που θα επηρεαστούν από την εντολή *UPDATE* αποθηκεύονται σε έναν προσωρινό πίνακα *P1* μέσω της εντολής

CREATE TABLE *P1* AS

SELECT Π_1, …, Π_v, $\Pi_{\chi\varepsilon}$, νέα_στήλη = $\Pi_{\chi\varepsilon}$

FROM *Π*

WHERE (*συνθήκη*) AND

cp($\pi_{\chi\varepsilon}$, *έκφραση*) = 1

όπου *συνθήκη* είναι η συνθήκη που εμφανίζεται στην πρόταση *WHERE* της εντολής *UPDATE* που έδωσε ο χρήστης και *έκφραση* είναι το δεξί μέλος της εκχώρησης $\Pi_{\chi\varepsilon}$ = *έκφραση* στην πρόταση *PORTION*. Ο πίνακας *P1* περιέχει μία επιπλέον στήλη, σε σχέση με τον πίνακα *Π* (τη στήλη *νέα_στήλη*), η τιμή της οποίας είναι ίση με την τιμή της στήλης $\Pi_{\chi\varepsilon}$. Ακολούθως, οι πλειάδες που υπάρχουν στον πίνακα *P1* διαγράφονται από τον πίνακα *Π* μέσω της εντολής

DELETE FROM *Π*

WHERE EXISTS (SELECT * FROM *P1*

WHERE *Π*.Π_1 = *P1*.Π_1 AND

… AND

Π.Π_v = *P1*.Π_v AND

Π.$\Pi_{\chi\varepsilon}$ = *P1*.$\Pi_{\chi\varepsilon}$)

ii) δημιουργείται ένας δεύτερος προσωρινός πίνακας *P2*, ο οποίος είναι όμοιος με τον πίνακα *P1*, εκτός της στήλης *νέα_στήλη*, η τιμή της οποίας θα είναι ίση με το κοινό τμήμα της τρέχουσας τιμής της και της έκφρασης *έκφραση*, που βρίσκεται στο δεξί μέλος της εκχώρησης $\Pi_{\chi\varepsilon}$ = *έκφραση* στην πρόταση *PORTION*. Ο πίνακας *P2* δημιουργείται με την εντολή

CREATE TABLE *P2* AS

SELECT Π_1, …, Π_v, $\Pi_{\chi\varepsilon}$, νέα_στήλη = intervsect(νέα_στήλη, έκφραση)

FROM *P1*

Στη συνέχεια, εκτελείται η πράξη *P1 PEXCEPT[νέα_στήλη] P2* και το αποτέλεσμα αποθηκεύεται σε έναν προσωρινό πίνακα *P3*. Ο πίνακας *P3* περιέχει πλέον τα τμήματα των πλειάδων του πίνακα *Π* που *δεν* θα επηρεαστούν από την εντολή *UPDATE*.

iii) η πρόταση *SET* της εντολής *UPDATE* που έδωσε ο χρήστης τροποποιείται, έτσι ώστε αν περιέχει κάποια εκχώρηση της μορφής $\Pi_{\chi\varepsilon}$ = *έκφραση*, αυτή αντικαθίσταται από την εκχώρηση *νέα_στήλη* = *έκφραση*. Στη συνέχεια, η εντολή

UPDATE *P2 τροποποιημένη_πρόταση_SET*

προωθείται στο σχεσιακό ΣΔΒΔ, ενημερώνοντας τον πίνακα *P2*.

iv) οι πλειάδες του πίνακα *P3* εισάγονται στον πίνακα *Π*, προωθώντας στο σχεσιακό ΣΔΒΔ την εντολή

INSERT INTO *Π* SELECT Π_1, …, Π_v, νέα_στήλη FROM *P3*

v) δημιουργείται ένας προσωρινός πίνακας *P4*, ο οποίος είναι όμοιος με τον πίνακα *P2*, με τη διαφορά ότι δεν περιέχει τη στήλη $\Pi_{\chi\varepsilon}$. Οι πλειάδες του πίνακα *P4* εισάγονται στον πίνακα *Π*, ακολουθώντας τον αλγεβρικό αλγόριθμο εισαγωγής δεδομένων (βλέπε *Η εντολή INSERT.*, σελ. 93).

4.4.2. Μοντέλο κόστους.

Στις επόμενες παραγράφους παρουσιάζεται ένα μαθηματικό μοντέλο κόστους, βάσει του οποίου συγκρίνονται οι επιδόσεις των βελτιστοποιημένων και των αλγεβρικών αλγορίθμων για τις πράξεις *INSERT*, *DELETE* και *UPDATE*. Για τον υπολογισμό του κόστους των αλγορίθμων λαμβάνεται υπόψη το κόστος μεταφοράς των δεδομένων από και προς το δίσκο, καθώς και το κόστος μεταφοράς των δεδομένων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό και αντίστροφα. Μόνο το σχεσιακό ΣΔΒΔ έχει τη δυνατότητα να διαβάζει και να γράφει δεδομένα στους δίσκους, κατά συνέπεια, αν ο χρονολογικός φλοιός πρέπει να διαβάσει μία πλειάδα μιας σχέσης, πρέπει η πλειάδα αυτή να διαβαστεί πρώτα από το σχεσιακό ΣΔΒΔ και κατόπιν να προωθηθεί στον χρονολογικό φλοιό μέσω των μηχανισμών διαδιεργασιακής επικοινωνίας (interprocess communication-IPC) του συστήματος.

Τα δεδομένα διαβάζονται και γράφονται στους δίσκους σε *σελίδες* και το κόστος μιας τέτοιας μεταφοράς θα συμβολίζεται με $\chi_{\sigma\varepsilon\lambda}$. Για τη μεταφορά δεδομένων από τον χρονολογικό φλοιό προς τον πυρήνα του σχεσιακού ΣΔΒΔ χρησιμοποιείται εμφυτευμένη SQL, στην οποία η μονάδα μεταφοράς είναι η *πλειάδα*. Η μεταφορά γίνεται μέσω των μηχανισμών διαδιεργασιακής επικοινωνίας, που μπορεί να είναι είτε διαμοιραζόμενη μνήμη (shared memory), συνδυαζόμενη με σημαφόρους (semaphores), είτε μεταβίβαση μηνυμάτων (message passing). Το κόστος μεταφοράς μιας πλειάδας ανάμεσα στον χρονολογικό φλοιό και τον πυρήνα του σχεσιακού ΣΔΒΔ είναι $\chi_{\pi\lambda\varepsilon\iota\acute{\alpha}\delta\alpha\varsigma} = \chi_{\varepsilon\pi\iota\beta} + \mu * \chi_{byte}$, όπου $\chi_{\varepsilon\pi\iota\beta}$ είναι η

επιβάρυνση που εισάγεται από τις διαδικασίες διαδιεργασιακής επικοινωνίας για κάθε λειτουργία μεταφοράς δεδομένων, μ είναι το μέγεθος της πλειάδας σε bytes και χ_{byte} είναι το κόστος αντιγραφής ενός byte. Στο μοντέλο κόστους θα χρησιμοποιηθούν οι ακόλουθοι συμβολισμοί:

- ΜΣ είναι το μέγεθος μιας σελίδας δίσκου.
- Μ είναι το ποσό μνήμης που είναι διαθέσιμο στο σχεσιακό ΣΔΒΔ.
- Δ είναι ο *παράγων διακλάδωσης εισόδου* (fan-in factor) της συγχωνευτικής ταξινόμησης (merge-sort).
- ΜΔ είναι το μέγεθος ενός δεδομένου τύπου *διάστημα*, σε bytes.

Για κάθε πίνακα *Π*, θα χρησιμοποιούνται οι ακόλουθοι συμβολισμοί:

- Σ_{Π} είναι το μέγεθος του πίνακα *Π* σε σελίδες.
- μ_{Π} είναι το μέγεθος της πλειάδας του πίνακα *Π* σε bytes.
- N_{Π} είναι το πλήθος των πλειάδων του πίνακα *Π*. Αν το ποσοστό πλήρωσης σελίδων είναι 100%, τότε $\Sigma_{\Pi} = \lceil (\mu_{\Pi} * N_{\Pi}) / \text{ΜΣ} \rceil$
- αν στον πίνακα *Π* έχει ορισθεί πρωτεύον κλειδί, τότε με μ'_{Π} συμβολίζουμε το μέγεθος μιας πλειάδας που αποτελείται από τις στήλες του πίνακα *Π* που μετέχουν στο κλειδί.

Τέλος, για το σχεσιακό ΣΔΒΔ γίνονται οι ακόλουθες υποθέσεις:

- δεν παρέχεται άμεσος έλεγχος για τον αλγόριθμο που θα χρησιμοποιηθεί, προκειμένου να εκτελεστούν οι ζητούμενες λειτουργίες. Έτσι, αν το ΣΔΒΔ υλοποιεί τρεις αλγορίθμους σύνδεσης, δεν παρέχεται η δυνατότητα στον χρήστη να επιλέξει τον αλγόριθμο που θα χρησιμοποιηθεί για να εκτελεστεί μία συγκεκριμένη σύνδεση.
- το σχεσιακό ΣΔΒΔ υλοποιεί έναν αλγόριθμο ταξινόμησης, ο οποίος χρησιμοποιείται για ταξινόμηση και υπολογισμό των συναθροιστικών συναρτήσεων, καθώς και έναν αλγόριθμο *ένθετων βρόχων* (nested loops), ο οποίος χρησιμοποιείται για τον υπολογισμό του αποτελέσματος των συνδέσεων. Αν και υπάρχουν πιο αποδοτικοί αλγόριθμοι για τον υπολογισμό των συνδέσεων και των συναθροιστικών συναρτήσεων (σύνδεση με συγχώνευση (merge-join), σύνδεση με κερματισμό (hash-join), σύνδεση βασισμένη σε δείκτες (pointer-based join), συνάθροιση με κερματισμό (hash-based aggregation)), δεν υλοποιούνται από όλα τα σχεσιακά ΣΔΒΔ.
- για να εισαχθεί μία πλειάδα σε κάποια σχέση, απλά προστίθεται μετά το τέλος των υπαρχόντων δεδομένων. Για να διαγραφεί μία πλειάδα, τίθεται μία ένδειξη η οποία τη

χαρακτηρίζει ως "διαγεγραμμένη". Κατά την εισαγωγή και διαγραφή πλειάδων δεν ενημερώνονται δείκτες ή άλλες δευτερεύουσες δομές δεδομένων.

I. Οι πράξεις της ΣΑΧΕ.

Οι αλγεβρικοί αλγόριθμοι εκτέλεσης των εντολών *INSERT*, *DELETE* και *UPDATE* χρησιμοποιούν τις πράξεις *FOLD*, *PUNION* και *PEXCEPT* της ΣΑΧΕ. Στις επόμενες παραγράφους υπολογίζεται το κόστος αυτών των πράξεων.

i) Η πράξη σύμπτυξη.

Προκειμένου να συμπτυχθεί ένας πίνακας *Π* σε μία στήλη του, ο πίνακας αρχικά ταξινομείται (με προαιρετική απομάκρυνση των διπλοτύπων πλειάδων). Στη συνέχεια, οι πλειάδες του διαβάζονται και υπόκεινται σε επεξεργασία και, τέλος, το αποτέλεσμα αποθηκεύεται στο δίσκο. Η ταξινόμηση γίνεται από τον πυρήνα του ΣΔΒΔ και το κόστος της είναι

$$\chi(\text{ΤΑΞΙΝΟΜΗΣΗ}(\Pi)) = 2 * \Sigma_{\Pi} * \log_{\Delta}(\Sigma_{\pi} / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} \quad (1)$$

([Graefe93]). Το κόστος ανάγνωσης της ταξινομημένης σχέσης είναι $\Sigma_{\Pi} * \chi_{\sigma\epsilon\lambda} + \Sigma_{\Pi} * M\Sigma * \chi_{byte} + N_{\Pi} * \chi_{\epsilon\pi\iota\beta}$· ο πρώτος όρος αντιστοιχεί στο χρόνο που δαπανά το σχεσιακό ΣΔΒΔ για να διαβάσει τις πλειάδες, ενώ οι επόμενοι δύο αντιπροσωπεύουν το κόστος μεταφοράς των πλειάδων από τον πυρήνα του σχεσιακού ΣΔΒΔ στον χρονολογικό φλοιό. Το μέγεθος του αποτελέσματος εξαρτάται από τον αριθμό των συμπτύξεων πλειάδων που θα λάβουν χώρα: αν δεν υπάρχουν συμπτυσσόμενες πλειάδες, το πλήθος των πλειάδων του αποτελέσματος θα είναι ίσο με το πλήθος των πλειάδων εισόδου, ενώ στην ακραία περίπτωση, όλες οι πλειάδες εισόδου μπορούν να συμπτυχθούν σε μία πλειάδα εξόδου. Αν το αποτέλεσμα αποθηκεύεται στον πίνακα *P1* και περιέχει N_{P1} πλειάδες, το κόστος εγγραφής του αποτελέσματος στο δίσκο είναι $N_{T1} * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} + N_{T1} * \mu_{\Pi} * \chi_{byte} + N_{T1} * \chi_{\epsilon\pi\alpha\beta}$. Αν δεν υπάρχει καμία ένδειξη για το μέγεθος του τελικού αποτελέσματος, τότε θα υποθέτουμε ότι είναι ίσο με το μέγεθος της σχέσης εισόδου, οπότε το κόστος εγγραφής του στο δίσκο θα είναι $\Sigma_{\Pi} * \chi_{\sigma\epsilon\lambda} + \Sigma_{\Pi} * M\Sigma * \chi_{byte} + N_{\Pi} * \chi_{\epsilon\pi\iota\beta}$.

Σύμφωνα με τα ανωτέρω, το συνολικό κόστος της πράξης *σύμπτυξη*, όταν αυτή εφαρμόζεται πάνω σε μία στήλη και παράγει αποτέλεσμα το οποίο περιέχει N_{T1} πλειάδες είναι

$$\chi(\text{FOLD}(\Pi, N_{T1})) = 2 * \Sigma_{\Pi} * \log_{\Delta}(\Sigma_{\Pi} / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + \Sigma_{\Pi} * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (N_{\Pi} + N_{T1}) * \chi_{\epsilon\pi\iota\beta} + N_{T1} * \mu_{\Pi} * (\chi_{\sigma\epsilon\lambda} / M\Sigma + \chi_{byte}) \quad (2)$$

Στις περιπτώσεις που υποθέτουμε ότι το μέγεθος του αποτελέσματος είναι ίσο με το μέγεθος της σχέσης εισόδου, το κόστος της πράξης *σύμπτυξη* δίνεται από τον τύπο

$$\chi(\text{FOLD}(\Pi)) = 2 * \Sigma_\Pi * \log_\Delta(\Sigma_\Pi / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * \Sigma_\Pi * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * N_\Pi * \chi_{\epsilon\pi\iota\beta} \quad (3)$$

ii) Η πράξη σημειακή ένωση.

Η εκτέλεση της πράξης *σημειακή ένωση* σε μία στήλη, για δύο σχέσεις *Π*, *P* συνίσταται στον υπολογισμό της ένωσης των δύο σχέσεων και την εφαρμογή της πράξης *σύμπτυξη* πάνω στο αποτέλεσμα. Ο υπολογισμός του αποτελέσματος της ένωσης γίνεται από τον πυρήνα του σχεσιακού ΣΔΒΔ και το κόστος του είναι $2 * (\Sigma_\Pi + \Sigma_P)$. Το αποτέλεσμα της πράξης *σημειακή ένωση* αποθηκεύεται στον πίνακα *T1*, ο οποίος περιέχει N_{T1} πλειάδες, με $1 \le N_{T1} \le N_\Pi + N_P$, κατά συνέπεια, σύμφωνα με την εξίσωση (2), το κόστος εφαρμογής της πράξης *σύμπτυξη* πάνω στο αποτέλεσμα της ένωσης των δύο πινάκων (το οποίο συμβολίζουμε με *T2*) είναι

$\chi(\text{Σύμπτυξη}(T2, N_{T1})) = 2 * (\Sigma_\Pi + \Sigma_P) * \log_\Delta((\Sigma_\Pi + \Sigma_P) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + (\Sigma_\Pi + \Sigma_P + N_{T1} * \mu_\Pi / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (N_\Pi + N_P + N_{T1}) * \chi_{\epsilon\pi\iota\beta}$

Αν δεν υπάρχει καμία ένδειξη σχετικά με το μέγεθος του τελικού αποτελέσματος της πράξης *σημειακή ένωση*, θα υποτίθεται ότι το μέγεθός του θα είναι ίσο με το άθροισμα των μεγεθών των σχέσεων εισόδου, συνεπώς το κόστος της πράξης *σύμπτυξη* υπολογίζεται βάσει της εξίσωσης (3) και είναι ίσο με

$\chi(\text{Σύμπτυξη}(T2)) = 2 * (\Sigma_\Pi + \Sigma_P) * \log_\Delta((\Sigma_\Pi + \Sigma_P) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * (\Sigma_\Pi + \Sigma_P) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * (N_\Pi + N_P) * \chi_{\epsilon\pi\iota\beta}$.

Σύμφωνα με τα ανωτέρω, το κόστος της πράξης *σημειακή ένωση* είναι

$$\chi(\text{PUNION}(\Pi, P, N_{T1})) = 2 * (\Sigma_\Pi + \Sigma_P) * (\log_\Delta((\Sigma_\Pi + \Sigma_P) / (2 * M) + 1) + 1) * \chi_{\sigma\epsilon\lambda} + (\Sigma_\Pi + \Sigma_P + N_{T1} * \mu_\Pi / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (N_\Pi + N_P + N_{T1}) * \chi_{\epsilon\pi\iota\beta} \quad (4)$$

Αν δεν υπάρχει καμία ένδειξη σχετικά με το μέγεθος του τελικού αποτελέσματος, τότε θα χρησιμοποιείται το άνω φράγμα του κόστους της πράξης (όπου το μέγεθος του αποτελέσματος είναι ίσο με το άθροισμα των μεγεθών των δύο σχέσεων εισόδου), το οποίο δίνεται από την εξίσωση

$$\chi(\text{PUNION}(\Pi, P)) = 2 * (\Sigma_\Pi + \Sigma_P) * \log_\Delta((\Sigma_\Pi + \Sigma_P) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * (\Sigma_\Pi + \Sigma_P) * (2 * \chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * (N_\Pi + N_P) * \chi_{\epsilon\pi\iota\beta} \quad (5)$$

iii) Η πράξη σημειακή διαφορά.

Για τον υπολογισμό του αποτελέσματος της πράξης *σημειακή διαφορά* πάνω σε μία στήλη $\pi_{\chi\epsilon}$, για δύο πίνακες *Π, P* (των οποίων το σχήμα είναι $(\pi_1, \ldots, \pi_v, \pi_{\chi\epsilon})$) εκτελείται αρχικά η πράξη *SPLIT'* (βλέπε σελ. 74), η οποία απαιτεί να είναι οι δύο σχέσεις εισόδου ταξινομημένες ως προς τις στήλες $\pi_1, \ldots, \pi_v, \pi_{\chi\epsilon}$, με αυτή τη σειρά. Το αποτέλεσμα της πράξης

αποθηκεύεται σε έναν προσωρινό πίνακα *T1*, ο οποίος θα έχει μέγεθος το πολύ $\Sigma_{\Pi} + 2 * \Sigma_{P}$ σελίδες (ισοδύναμα, $N_{\Pi} + 2 * N_{P}$ πλειάδες). Το μέγεθος του πίνακα *T1* θα φτάσει αυτό το όριο όταν πληρούνται οι κάτωθι συνθήκες:

1) ο πίνακας *Π* δεν περιέχει ζεύγος πλειάδων οι οποίες μπορούν να συμπτυχθούν.
2) ο πίνακας *P* δεν περιέχει ζεύγος πλειάδων οι οποίες μπορούν να συμπτυχθούν.
3) για κάθε πλειάδα ρ του πίνακα *P* υπάρχει πλειάδα π του πίνακα *Π* τέτοια ώστε:
   i) $\pi.\pi_{\iota} = \rho.\pi_{\iota}, \forall\ \iota = 1, \ldots, \nu$
   ii) $start(\pi.\pi_{\chi\varepsilon}) < start(\rho.\pi_{\chi\varepsilon}) \wedge stop(\pi.\pi_{\chi\varepsilon}) > stop(\rho.\pi_{\chi\varepsilon})$

Στη συνέχεια, εκτελείται μία αντισύνδεση των πινάκων *T1* και *P*, η οποία παράγει τον προσωρινό πίνακα *T2* και ο πίνακας *T2* συμπτύσσεται πάνω στη στήλη $\pi_{\chi\varepsilon}$, για να παραχθεί το τελικό αποτέλεσμα. Ο πίνακας *T2*, καθώς και το τελικό αποτέλεσμα, θα έχουν μέγεθος το πολύ $\Sigma_{\Pi} + \Sigma_{P}$ σελίδες ($N_{\Pi} + N_{P}$ πλειάδες).

Δεδομένου ότι το κόστος της αντισύνδεσης μεταξύ δύο πινάκων *Π1* και *Π2* είναι

$$(\Sigma_{\Pi 1} / (M - 1) * (\Sigma_{\Pi 2} - 1) + 1) * \chi_{\sigma\varepsilon\lambda} \qquad (6)$$

([Graefe93] - ο πίνακας *Π1* είναι ο *εξωτερικός πίνακας* της αντισύνδεσης), το συνολικό κόστος της πράξης *σημειακή διαφορά* είναι

$$\begin{aligned}\chi(\text{PEXCEPT}(\Pi, P)) = {} & 2 * \Sigma_{\Pi} * \log_{\Delta}(\Sigma_{\Pi} / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * \Sigma_{P} * \log_{\Delta}(\Sigma_{P} / (2 * \\ & M) + 1) * \chi_{\sigma\varepsilon\lambda} + ((\Sigma_{\Pi} + 2 * \Sigma_{P}) / (M - 1) * (\Sigma_{P} - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{\Pi} + \Sigma_{P}) * \chi_{\sigma\varepsilon\lambda} + \\ & 2 * (\Sigma_{\Pi} + \Sigma_{P}) * \log_{\Delta}((\Sigma_{\Pi} + \Sigma_{P}) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + (2 * \Sigma_{\Pi} + 4 * \Sigma_{P}) * \chi_{\sigma\varepsilon\lambda} + (3 \\ & * N_{\Pi} + 4 * N_{P}) * (\mu_{\Pi} * \chi_{byte} + \chi_{\varepsilon\pi\iota\beta})\end{aligned} \qquad (7)$$

II. Η εντολή *INSERT*.

Στον αλγεβρικό αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα διακρίνονται δύο περιπτώσεις (ο πίνακας στον οποίο εισάγονται τα δεδομένα θα συμβολίζεται με *Π*. Η περίπτωση εισαγωγής δεδομένων σε μη κανονικοποιημένο πίνακα καλύπτεται άμεσα από το σχεσιακό ΣΔΒΔ οπότε δεν χρήζει ανάλυσης.):

1. *δεν έχει ορισθεί κλειδί πάνω στον πίνακα Π*. Στην περίπτωση αυτή απαιτείται μία σύνδεση μεταξύ του πίνακα *Π* και του πίνακα *P1* (*P1* είναι ο πίνακας που περιέχει τα προς εισαγωγή δεδομένα). Από τη σύνδεση αυτή παράγεται ο πίνακας *P2*, ο οποίος περιέχει τις πλειάδες του πίνακα *Π* που μπορούν να συμπτυχθούν με οποιαδήποτε πλειάδα του πίνακα *P1*. Το πλήθος των πλειάδων αυτών συμβολίζεται με N_{Σ}. Το κόστος της σύνδεσης και της δημιουργίας του πίνακα *P2* είναι

   $$(\Sigma_{\Pi} / (M - 1) * (\Sigma_{P1} - 1) + 1 + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * \chi_{\sigma\varepsilon\lambda} \qquad (8)$$

Ο πίνακας *Π* χρησιμοποιείται ως εξωτερικός πίνακας στη σύνδεση, μια και, πιθανότατα, θα περιέχει περισσότερες πλειάδες από τον πίνακα *P1*. Στην περίπτωση που ο πίνακας *P1* είναι μεγαλύτερος από τον πίνακα *Π*, μπορεί να χρησιμοποιηθεί αυτός ως εξωτερικός πίνακας, οπότε ο τύπος προσαρμόζεται αναλόγως.

Κατόπιν απαιτείται μία σύνδεση μεταξύ του πίνακα *Π* και του πίνακα *P2*, με σκοπό τη διαγραφή από τον πίνακα *Π* των πλειάδων που βρίσκονται στον πίνακα *P2*. Το κόστος αυτής της σύνδεσης είναι

$$(\Sigma_{\Pi} / (M - 1) * (N_{\Sigma} * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} \qquad (9)$$

ενώ το κόστος της φυσικής διαγραφής των πλειάδων εξαρτάται από τη διασπορά των πλειάδων ανάμεσα στις σελίδες του πίνακα *Π* και είναι δύσκολο να υπολογιστεί. Όμως, οι ίδιες πλειάδες διαγράφονται και στη βελτιστοποιημένη έκδοση του αλγορίθμου, οπότε το κόστος της διαγραφής μπορεί να παραλειφθεί και από τις δύο εκδόσεις, χωρίς να επηρεάζεται το αποτέλεσμα της σύγκρισης.

Ο αλγόριθμος ολοκληρώνεται με την εκτέλεση της πράξης *PUNION* πάνω στους πίνακες *P1* και *P2* και την πρόσθεση των πλειάδων του αποτελέσματος της πράξης στον πίνακα *Π*. Δεδομένου ότι κάθε πλειάδα του πίνακα *P2* μπορεί να συμπτυχθεί με κάποια πλειάδα του πίνακα *P1*, το πλήθος των πλειάδων του αποτελέσματος θα είναι κατά μέγιστον N_{P1}. Το κόστος της εκτέλεσης της πράξης *PUNION* και της πρόσθεσης των δεδομένων του αποτελέσματος στον πίνακα *Π* είναι

$$\begin{aligned} &2 * (\Sigma_{P1} + N_{\Sigma} * \mu_{\pi} / M\Sigma) * (\log_{\Delta}((\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) / (2 * M) + 1) + 1) * \chi_{\sigma\varepsilon\lambda} \\ &+ (2 * \Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) + (2 * N_{P1} + N_{\Sigma}) * \chi_{\varepsilon\pi\iota\beta} + 2 * \Sigma_{P1} \\ &* \chi_{\sigma\varepsilon\lambda} \end{aligned} \qquad (10)$$

Το συνολικό κόστος του αλγεβρικού αλγόριθμου εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα, στον οποίο δεν έχει ορισθεί πρωτεύον κλειδί, είναι το άθροισμα των παραστάσεων (8), (9) και (10) και δίνεται από τον τύπο

$$\begin{aligned} \chi(\mathrm{INSERT}(\Pi, P1, N_{\Sigma})) = {} & (\Sigma_{\Pi} / (M - 1) * (\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma - 2) + N_{\Sigma} * \mu_{\Pi} / \\ & M\Sigma + 2 * (\Sigma_{P1} + 1)) * \chi_{\sigma\varepsilon\lambda} + 2 * (\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\log_{\Delta}((\Sigma_{P1} + N_{\Sigma} * \\ & \mu_{\Pi} / M\Sigma) / (2 * M) + 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (2 * \Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + \\ & M\Sigma * \chi_{byte}) + (2 * N_{P1} + N_{\Sigma}) * \chi_{\varepsilon\pi\iota\beta} \end{aligned} \qquad (11)$$

Αν για τον προσδιορισμό των προς εισαγωγή δεδομένων έχει χρησιμοποιηθεί η πρόταση *VALUES*, τότε $N_{P1} = 1$ και, στη γενική περίπτωση, $\Sigma_{P1} = 1$, δεδομένου ότι το σύνηθες μέγεθος των σελίδων δίσκου κυμαίνεται μεταξύ 2048 και 8192 bytes και το μέγεθος

αυτό είναι μεγαλύτερο από το μέγεθος μιας τυπικής πλειάδας. Λαμβάνοντας επίσης υπόψη ότι M » 1, η εξίσωση κόστους (11) απλοποιείται σε

$$\chi(\text{INSERT}(\Pi, N_\Sigma)) = (\Sigma_\Pi / (M - 1) * (N_\Sigma * \mu_\Pi / M\Sigma - 1) + N_\Sigma * \mu_\Pi / M\Sigma + 4) * \chi_{\sigma\epsilon\lambda} + 2 * (N_\Sigma * \mu_\Pi / M\Sigma + 1) * (\log_\Delta((N_\Sigma * \mu_\Pi / M\Sigma + 1) / (2 * M) + 1) + 1) * \chi_{\sigma\epsilon\lambda} + (N_\Sigma * \mu_\Pi / M\Sigma + 2) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (N_\Sigma + 2) * \chi_{\epsilon\pi\beta} \quad (12)$$

2. *έχει ορισθεί κλειδί στον πίνακα Π*. Στην περίπτωση αυτή, πέραν των βημάτων της περίπτωσης (1), απαιτούνται δύο έλεγχοι, προκειμένου να διασφαλιστεί η μοναδικότητα του πρωτεύοντος κλειδιού. Ο πρώτος έλεγχος διενεργείται σε δύο βήματα, το πρώτο από τα οποία απαιτεί την ταξινόμηση των στηλών του πίνακα *P1* που απαρτίζουν το κλειδί και την αποθήκευσή τους σε έναν προσωρινό πίνακα με κόστος

$$(2 * (\Sigma_{P1} * \mu'_\Pi / \mu_\Pi) * \log_\Delta(\Sigma_{P1} * (\mu'_\Pi / \mu_\Pi) / (2 * M) + 1) + N_{P1} * (\mu_\Pi - \mu'_\Pi) / M\Sigma + N_{P1} * \mu'_\Pi / M\Sigma) * \chi_{\sigma\epsilon\lambda} \quad (13)$$

Ο πρώτος όρος αντιστοιχεί στο κόστος ταξινόμησης μιας σχέσης με μέγεθος πλειάδας μ'_Π και αριθμό πλειάδων ίσο με αυτόν της σχέσης *P1*, ο δεύτερος όρος αντιστοιχεί στις επιπρόσθετες μεταφορές από το δίσκο κατά την αρχική ανάγνωση του πίνακα *P1*, ενώ ο τρίτος όρος δίνει το κόστος εγγραφής του αποτελέσματος στο δίσκο. Το δεύτερο βήμα του πρώτου ελέγχου συνίσταται σε μία σύνδεση του πίνακα *P1* με τον εαυτό του, με κόστος

$$(\Sigma_{P1} / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\epsilon\lambda} \quad (14)$$

Ο δεύτερος έλεγχος απαιτεί τη σύνδεση του πίνακα *P1* με τον πίνακα *P3* (ο πίνακας *P3* περιέχει τις πλειάδες του πίνακα *Π* των οποίων τα κλειδιά μπορούν να συμπτυχθούν με το κλειδί κάποιας πλειάδας του πίνακα *P1*). Το κόστος της σύνδεσης αυτής είναι

$$(\Sigma_{P1} / (M - 1) * (N_\Sigma * \mu_\Pi / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} \quad (15)$$

Το συνολικό κόστος της εισαγωγής δεδομένων σε έναν κανονικοποιημένο πίνακα, στον οποίο έχει ορισθεί πρωτεύον κλειδί, δίνεται από το άθροισμα των τύπων (11), (13), (14) και (15) και είναι

$$\chi(\text{INSERT}(\Pi, P1, N_\Sigma)) = (2 * (\Sigma_{P1} * \mu'_\Pi / \mu_\Pi) * \log_\Delta(\Sigma_{P1} * (\mu'_\Pi / \mu_\Pi) / (2 * M) + 1) + N_{P1} * (\mu_\Pi - \mu'_\Pi) / M\Sigma + \Sigma_{P1} / (M - 1) * (\Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma - 2) + 2) * \chi_{\sigma\epsilon\lambda} + (\Sigma_\Pi / (M - 1) * (\Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma - 2) + N_\Sigma * \mu_\Pi / M\Sigma + 2 * (\Sigma_{P1} + 1)) * \chi_{\sigma\epsilon\lambda} + 2 * (\Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma) * (\log_\Delta((\Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma) / (2 * M) + 1) + 1) * \chi_{\sigma\epsilon\lambda} + (2 * \Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (2 * N_{P1} + N_\Sigma) * \chi_{\epsilon\pi\beta} \quad (16)$$

Αν για τον προσδιορισμό των προς εισαγωγή δεδομένων έχει χρησιμοποιηθεί η πρόταση *VALUES*, τότε $N_{P1} = 1$ και, στη γενική περίπτωση, $\Sigma_{P1} = 1$. Λαμβάνοντας επίσης υπόψη ότι $M \gg 1$ και ότι $\mu'_{\Pi} < \mu_{\Pi}$, η εξίσωση κόστους (16) απλοποιείται σε

$$\chi(\text{INSERT}(\Pi, N_{\Sigma})) = ((\mu_{\Pi} - \mu'_{\Pi}) / M\Sigma + 1) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_{\Sigma} * \mu_{\Pi} / M\Sigma - 1) + N_{\Sigma} * \mu_{\Pi} / M\Sigma + 4) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_{\Sigma} * \mu_{\Pi} / M\Sigma + 1) * (\log_{\Delta}((N_{\Sigma} * \mu_{\Pi} / M\Sigma + 1) / (2 * M) + 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (2 + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) + (2 + N_{\Sigma}) * \chi_{\varepsilon\pi\iota\beta} \quad (17)$$

Στον βελτιστοποιημένο αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα διακρίνονται τέσσερις περιπτώσεις:

1. *δεν έχει ορισθεί κλειδί στον πίνακα Π και οι προς εισαγωγή τιμές προσδιορίζονται μέσω της πρότασης VALUES*. Στην περίπτωση αυτή ο αλγόριθμος διατρέχει τον πίνακα *Π* προσκομίζοντας στη μνήμη του χρονολογικού φλοιού τον χρόνο εγκυρότητας των πλειάδων που μπορούν να συμπτυχθούν με την εισαγόμενη πλειάδα, ενώ παράλληλα διαγράφει τις πλειάδες αυτές. Τέλος, η εισαγόμενη πλειάδα (της οποίας ο χρόνος εγκυρότητας ενδεχομένως έχει τροποποιηθεί) εισάγεται στον πίνακα *Π*. Το κόστος αυτής της διαδικασίας είναι

   $$\chi(\text{INSERT}(\Pi, N_{\Sigma})) = (\Sigma_{\Pi} + 1) * \chi_{\sigma\varepsilon\lambda} + (N_{\Sigma} * M\Delta + \mu_{\Pi}) * \chi_{byte} + (N_{\Sigma} + 1) * \chi_{\varepsilon\pi\iota\beta} \quad (18)$$

   Η σύγκριση των εξισώσεων κόστους (12) και (18) καταδεικνύει ότι η απόδοση του βελτιστοποιημένου αλγορίθμου εισαγωγής δεδομένων είναι σημαντικά καλύτερη, σε σχέση με την απόδοση του αλγεβρικού αλγορίθμου.

2. *έχει ορισθεί κλειδί στον πίνακα Π και οι προς εισαγωγή τιμές προσδιορίζονται μέσω της πρότασης VALUES*. Στην περίπτωση αυτή ο αλγόριθμος διατρέχει τον πίνακα *Π* επιλέγοντας τις πλειάδες των οποίων το πρωτεύον κλειδί μπορεί να συμπτυχθεί με το πρωτεύον κλειδί της εισαγόμενης πλειάδας. Για κάθε τέτοια πλειάδα προσκομίζονται στη μνήμη του χρονολογικού φλοιού ο χρόνος εγκυρότητας και οι στήλες που δεν μετέχουν στο πρωτεύον κλειδί. Οι πλειάδες που μπορούν να συμπτυχθούν με την εισαγόμενη πλειάδα διαγράφονται από τον πίνακα *Π* και, τέλος, η εισαγόμενη πλειάδα (της οποίας ο χρόνος εγκυρότητας ενδεχομένως έχει τροποποιηθεί) εισάγεται στον πίνακα *Π*. Το κόστος αυτής της διαδικασίας είναι

   $$\chi(\text{INSERT}(\Pi, N_{\Sigma})) = (\Sigma_{\Pi} + 1) * \chi_{\sigma\varepsilon\lambda} + (N_{\Sigma} * (\mu_{\Pi} - \mu'_{\Pi} + M\Delta) + \mu_{\Pi}) * \chi_{byte} + (N_{\Sigma} + 1) * \chi_{\varepsilon\pi\iota\beta} \quad (19)$$

   Η σύγκριση των εξισώσεων κόστους (17) και (19) καταδεικνύει ότι η απόδοση του βελτιστοποιημένου αλγορίθμου εισαγωγής δεδομένων είναι σημαντικά καλύτερη, σε

σχέση με την απόδοση του αλγεβρικού αλγορίθμου. Η βελτίωση στις περιπτώσεις (1) και (2) είναι σημαντική για τις εφαρμογές των οποίων το περιβάλλον διασύνδεσης με τον χρήστη λειτουργεί σε επίπεδο πλειάδας, διότι οι εφαρμογές αυτές εισάγουν τις πλειάδες μία προς μία.

3. *δεν έχει ορισθεί κλειδί στον πίνακα Π και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω ερώτησης*. Σ' αυτή την περίπτωση εκτελείται μία σύνδεση μεταξύ των πινάκων *Π* και *P1* (*P1* είναι ο πίνακας που περιέχει τα προς εισαγωγή δεδομένα), μέσω της οποίας οι πλειάδες του πίνακα *Π* που μπορούν να συμπτυχθούν με οποιαδήποτε πλειάδα του πίνακα διαγράφονται από τον πίνακα *Π* και εισάγονται στον πίνακα *P1*. Ακολούθως, ο πίνακας *P1* συμπτύσσεται πάνω στη στήλη που περιέχει τον χρόνο εγκυρότητας και οι πλειάδες που περιέχονται στο αποτέλεσμα της πράξης *σύμπτυξη* εισάγονται στον πίνακα *Π*. Σύμφωνα με τις εξισώσεις (2) και (6), το κόστος της διαδικασίας αυτής είναι

$$\begin{aligned}\chi(\text{INSERT}(\Pi, P1, N_\Sigma)) = {} & (\Sigma_\Pi / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * (\Sigma_{P1} + N_\Sigma * \\ & \mu_\Pi / M\Sigma) * \log_\Delta((\Sigma_{P1} + N_\Sigma * \mu_\Pi / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{P1} + N_\Sigma * \\ & \mu_\Pi / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) + N_{P1} * \mu_\Pi * (\chi_{\sigma\varepsilon\lambda} / M\Sigma + \chi_{byte}) + 2 * \Sigma_{P1} * \\ & \chi_{\sigma\varepsilon\lambda} + (3 * N_\Sigma + 2 * N_{P1}) * \chi_{\varepsilon\pi\iota\beta} + 2 * N_\Sigma * \mu_\Pi * \chi_{byte}\end{aligned} \qquad (20)$$

Αφαιρώντας την εξίσωση (20) από την εξίσωση (11) έχουμε τη βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγορίθμου αντί του αλγεβρικού. Το αποτέλεσμα της αφαίρεσης είναι

$$\begin{aligned}\chi_{\alpha\lambda\gamma} - \chi_{\beta\varepsilon\lambda\tau} = {} & (\Sigma_\Pi / (M - 1) * (N_\Sigma * \mu_\Pi / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + N_\Sigma * \mu_\Pi / M\Sigma * \chi_{\sigma\varepsilon\lambda} - \\ & 2 * (N_\Sigma * \mu_\Pi * \chi_{byte} + N_\Sigma * \chi_{\varepsilon\pi\iota\beta})\end{aligned} \qquad (21)$$

Ο πρώτος όρος αντιπροσωπεύει το κόστος μιας σύνδεσης μεταξύ του πίνακα *Π* και του πίνακα *P2*, ο οποίος περιέχει τις πλειάδες του πίνακα *Π* που μπορούν να συμπτυχθούν με οποιαδήποτε πλειάδα του πίνακα *P1*, ενώ ο δεύτερος όρος αντιστοιχεί στο κόστος ανάγνωσης (ή εγγραφής) από τον δίσκο του πίνακα *P2*. Τέλος, ο τρίτος όρος (που είναι αρνητικός), δίνει το κόστος μεταφοράς του πίνακα *P2* από τον πυρήνα του σχεσιακού ΣΔΒΔ στον χρονολογικό φλοιό και αντίστροφα. Δεδομένου ότι η αντιγραφή στη μνήμη είναι κατά πολύ ταχύτερη της μεταφοράς από και προς το δίσκο, η διαφορά του τρίτου όρου από τον δεύτερο είναι θετική. Σε κάθε περίπτωση, όμως, ο κυρίαρχος παράγοντας της εξίσωσης (21) είναι το κόστος της σύνδεσης, συνεπώς η βελτίωση που επέρχεται από τη χρήση του βελτιστοποιημένου αλγόριθμου είναι σημαντική.

4. *έχει ορισθεί κλειδί στον πίνακα Π και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω ερώτησης*. Σ' αυτή την περίπτωση εκτελείται μία σύνδεση μεταξύ των πινάκων *Π* και *P1*,

μέσω της οποίας επιλέγονται τα ζεύγη πλειάδων *(π, ρ)* των πινάκων *Π* και *P*, αντίστοιχα, των οποίων τα πρωτεύοντα κλειδιά μπορούν να συμπτυχθούν. Για κάθε επιλεγόμενο ζεύγος πλειάδων, προσκομίζονται στη μνήμη όλες οι στήλες της πλειάδας *Π*, ο χρόνος εγκυρότητας της πλειάδας *ρ* και οι στήλες της πλειάδας *ρ* που δεν μετέχουν στο κλειδί. Αν οι πλειάδες *π* και *ρ* μπορούν να συμπτυχθούν, τότε η πλειάδα *π* διαγράφεται από τον πίνακα *Π* και εισάγεται στον πίνακα *P1*. Το κόστος αυτής της διαδικασίας είναι

$$(\Sigma_{\Pi} / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\epsilon\lambda} + N_{\Sigma} * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} + (N'_{\Sigma} + 2 * N_{\Sigma}) * \chi_{\epsilon\pi\iota\beta} + ((N_{\Sigma} + N'_{\Sigma}) * (2 * \mu_{\Pi} - \mu'_{\Pi}) + N_{\Sigma} * \mu_{\Pi}) * \chi_{byte} \quad (22)$$

όπου N'_{Σ} είναι ο αριθμός των πλειάδων που θα προσκομιστούν στη μνήμη του χρονολογικού φλοιού, αλλά δεν μπορούν να συμπτυχθούν με κάποια πλειάδα του πίνακα *P1* (λόγω του ότι κάποια από τις στήλες που δεν μετέχουν στο κλειδί έχει διαφορετική τιμή στις πλειάδες *π* και *ρ*). Δεδομένου ότι έχει ορισθεί κλειδί στον πίνακα *Π*, το άθροισμα $N_{\Sigma} + N'_{\Sigma}$ φράσσεται άνω από το όριο $2 * N_{P1}$, κατά συνέπεια ο τύπος (22) μπορεί να απλοποιηθεί σε

$$(\Sigma_{\Pi} / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\epsilon\lambda} + N_{\Sigma} * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} + (2 * N_{P1} + N_{\Sigma}) * \chi_{\epsilon\pi\iota\beta} + (2 * N_{P1} * (2 * \mu_{\Pi} - \mu'_{\Pi}) + N_{\Sigma} * \mu_{\Pi}) * \chi_{byte} \quad (23)$$

Ακολούθως, απαιτείται ταξινόμηση του πίνακα *P1*, προσκόμιση στη μνήμη του χρονολογικού φλοιού όλων των πλειάδων του (που τώρα είναι $N_{P1} + N_{\Sigma}$) και πρόσθεση στον πίνακα *Π* των πλειάδων που θα παραχθούν ως αποτέλεσμα της διαδικασίας ελέγχου παραβίασης της μοναδικότητας του πρωτεύοντος κλειδιού και σύμπτυξης. Το κόστος του δεύτερου βήματος είναι

$$2 * (\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * \log_{\Delta}((\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * (\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * (N_{P1} + N_{\Sigma}) * \chi_{\epsilon\pi\iota\beta} \quad (24)$$

Βάσει των τύπων (23) και (24), το συνολικό κόστος εισαγωγής δεδομένων σε έναν κανονικοποιημένο πίνακα, όταν σ' αυτόν έχει ορισθεί κλειδί είναι

$$\begin{aligned}\chi(\mathrm{INSERT}(\Pi, P1, N_{\Sigma})) = {} & (\Sigma_{\Pi} / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\epsilon\lambda} + N_{\Sigma} * \mu_{\Pi} / M\Sigma * \\ & \chi_{\sigma\epsilon\lambda} + (2 * N_{P1} + N_{\Sigma}) * \chi_{\epsilon\pi\iota\beta} + (2 * N_{P1} * (2 * \mu_{\Pi} - \mu'_{\Pi}) + N_{\Sigma} * \mu_{\Pi}) * \chi_{byte} + 2 \\ & * (\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * \log_{\Delta}((\Sigma_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + \\ & 2 * (P_{P1} + N_{\Sigma} * \mu_{\Pi} / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * (N_{P1} + N_{\Sigma}) * \chi_{\epsilon\pi\iota\beta}\end{aligned} \quad (25)$$

Αφαιρώντας την εξίσωση (25) από την εξίσωση (16) έχουμε τη βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγορίθμου αντί του αλγεβρικού. Το αποτέλεσμα της αφαίρεσης είναι

$$\chi_{\alpha\lambda\gamma} - \chi_{\beta\varepsilon\lambda\tau} = (2 * (\Sigma_{P1} * \mu'_{\Pi} / \mu_{\Pi}) * \log_{\Delta}(\Sigma_{P1} * (\mu'_{\Pi} / \mu_{\Pi}) / (2 * M) + 1) + N_{P1} * (\mu_{\Pi} - \mu'_{\Pi}) / M\Sigma + \Sigma_{P1} / (M - 1) * (\Sigma_{P1} - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{P1} / (M - 1) * (N_{\Sigma} * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_{\Sigma} * \mu_{\Pi} / M\Sigma - 1) + 2 * \Sigma_{P1} + 1) * \chi_{\sigma\varepsilon\lambda} - (2 * N_{P1} * (2 * \mu_{\Pi} - \mu'_{\Pi}) + N_{\Sigma} * \mu_{\Pi}) * \chi_{byte} - 2 * N_{\Sigma} * \mu_{\Pi} / M\Sigma * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) - 2 * (N_{P1} + N_{\Sigma}) * \chi_{\varepsilon\pi\iota\beta} \quad (26)$$

Οι τρεις πρώτοι όροι αντιστοιχούν (προσεγγιστικά) σε μία ταξινόμηση (του πίνακα *P1*) και σε τρεις συνδέσεις (μία σύνδεση μεταξύ του πίνακα *P1* και του εαυτού του, μία σύνδεση μεταξύ του πίνακα *Π* και του πίνακα *P3* (ο πίνακας *P3* περιέχει τις πλειάδες του πίνακα *Π* των οποίων το κλειδί μπορεί να συμπτυχθεί με το κλειδί κάποιας πλειάδας του πίνακα *P1*) και μία σύνδεση μεταξύ των πινάκων *P1* και *P3*). Οι τρεις αρνητικοί όροι που ακολουθούν, αντιπροσωπεύουν το κόστος μετακίνησης δεδομένων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό και αντίστροφα. Η βελτίωση στην απόδοση είναι σημαντική: εκτός από το ότι οι μεταφορές στη μνήμη είναι ταχύτερες από την ανάγνωση/εγγραφή από τους δίσκους, το κόστος των συνδέσεων αυξάνεται πολυωνυμικά, σε σχέση με τον παράγοντα Σ_{P1} ($O(\Sigma_{P1}^2)$) και γραμμικά, σε σχέση με τους παράγοντες Σ_{Π} και N_{Σ}, ενώ το κόστος των μεταφορών στη μνήμη αυξάνει γραμμικά, σε σχέση με τους παράγοντες Σ_{P1} και N_{Σ}, ενώ είναι ανεξάρτητο του παράγοντα Σ_{Π}.

III. Η εντολή *DELETE*.

Ο αλγεβρικός αλγόριθμος διαγραφής δεδομένων από έναν κανονικοποιημένο πίνακα *Π* μέσω μιας εντολής *DELETE* που περιέχει την πρόταση *PORTION* (η διαγραφή δεδομένων με εντολές *DELETE* που δεν περιέχουν την πρόταση *PORTION* αντιμετωπίζεται από το σχεσιακό ΣΔΒΔ, οπότε δεν χρήζει ανάλυσης), απαιτεί αρχικά τη δημιουργία ενός προσωρινού πίνακα *P1*, ο οποίος περιέχει τις πλειάδες του πίνακα *Π* που θα επηρεαστούν από τη διαγραφή. Ακολούθως εκτελείται μία σύνδεση μεταξύ των πινάκων *Π* και *P1* για να διαγραφούν οι πλειάδες που βρίσκονται στον πίνακα *P1* από τον πίνακα *Π*. Το κόστος της δημιουργίας του πίνακα *P1* και της διαγραφής των πλειάδων από τον πίνακα *Π* είναι

$$(\Sigma_{\Pi} + N_{\Delta} * \mu_{\Pi} / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_{\Delta} * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} \quad (27)$$

όπου N_{Δ} είναι ο αριθμός πλειάδων που θα επηρεαστούν από τη διαγραφή. Στη συνέχεια, δημιουργείται ένας δεύτερος προσωρινός πίνακας *P2* που περιέχει τις πλειάδες του πίνακα *P1* τροποποιημένες ώστε ο χρόνος εγκυρότητάς τους να έχει περιοριστεί στο διάστημα που καθορίζεται από την πρόταση *PORTION*. Το κόστος δημιουργίας του πίνακα *P2* είναι

$$2 * N_{\Delta} * \mu_{\Pi} / M\Sigma * \chi_{\sigma\varepsilon\lambda} \quad (28)$$

Τέλος, εκτελείται η πράξη *P1 PEXCEPT[$\pi_{\chi\varepsilon}$] P2* (όπου $\pi_{\chi\varepsilon}$ είναι η στήλη που περιέχει τον χρόνο εγκυρότητας των πλειάδων) και το αποτέλεσμα της πράξης *σημειακή διαφορά* εισάγεται στον πίνακα *Π*. Το κόστος αυτής της διαδικασίας είναι

$$4 * (N_\Delta * \mu_\Pi / M\Sigma) * \log_\Delta((N_\Delta * \mu_\Pi / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_\Delta * \mu_\Pi / M\Sigma) * (2 * \chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * N_\Delta * \chi_{\varepsilon\pi\iota\beta} \quad (29)$$

Το συνολικό κόστος του αλγεβρικού αλγόριθμου διαγραφής δεδομένων από έναν κανονικοποιημένο πίνακα χρησιμοποιώντας εντολή *DELETE* που περιέχει την πρόταση *PORTION* δίνεται από το άθροισμα των τύπων (27), (28) και (29) και είναι

$$\chi(\text{DELETE}(\Pi, N_\Delta)) = (\Sigma_\Pi + N_\Delta * \mu_\Pi / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_\Pi / (M - 1) * (N_\Delta * \mu_\Pi / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + 4 * (N_\Delta * \mu_\Pi / M\Sigma) * \log_\Delta((N_\Delta * \mu_\Pi / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_\Delta * \mu_\Pi / M\Sigma) * (3 * \chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * N_\Delta * \chi_{\varepsilon\pi\iota\beta} \quad (30)$$

Ο βελτιστοποιημένος αλγόριθμος διαγραφής δεδομένων από έναν κανονικοποιημένο πίνακα *Π* χρησιμοποιώντας εντολή *DELETE* που περιέχει την πρόταση *PORTION*, διατρέχει τον πίνακα *Π* επιλέγοντας τις πλειάδες που θα επηρεαστούν από τη διαγραφή και, για κάθε επιλεγόμενη πλειάδα, εκτελούνται, στη χειρότερη περίπτωση, μία ενημέρωση και μία εισαγωγή. Το κόστος αυτής της διαδικασίας είναι

$$\chi(\text{DELETE}(\Pi, N_\Delta)) = (\Sigma_\Pi + N_\Delta) * \chi_{\sigma\varepsilon\lambda} + 2 * N_\Delta * (\mu_\Pi * \chi_{byte} + \chi_{\varepsilon\pi\iota\beta} + M\Delta * \chi_{byte}) \quad (31)$$

Η σύγκριση των εξισώσεων (30) και (31) μας οδηγεί στο συμπέρασμα ότι η απόδοση του βελτιστοποιημένου αλγορίθμου είναι σημαντικά καλύτερη από αυτή του αλγεβρικού αλγορίθμου.

IV. Η εντολή *UPDATE*.

Στον αλγεβρικό αλγόριθμο ενημέρωσης ενός κανονικοποιημένου πίνακα *Π*, διακρίνονται δύο περιπτώσεις (η ενημέρωση μη κανονικοποιημένων πινάκων καλύπτεται από το σχεσιακό ΣΔΒΔ οπότε δεν χρήζει ανάλυσης):

1. *δεν χρησιμοποιείται η πρόταση PORTION*. Στην περίπτωση αυτή, οι υπό ενημέρωση πλειάδες αποθηκεύονται σε έναν προσωρινό πίνακα *P1* και κατόπιν διαγράφονται από τον πίνακα *Π*, μέσω μιας σύνδεσής του με τον πίνακα *P1*. Ακολούθως, οι πλειάδες του πίνακα *P1* ενημερώνονται, όπως ορίζεται στην πρόταση *SET* της εντολής *UPDATE* και εισάγονται στον πίνακα *Π*, ακολουθώντας τον κατάλληλο αλγεβρικό αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα (ανάλογα με το αν στον πίνακα *Π* έχει ορισθεί πρωτεύον κλειδί ή όχι).

Το κόστος της δημιουργίας του πίνακα *P1*, της διαγραφής των πλειάδων του πίνακα *P1* από τον πίνακα *Π* και της ενημέρωσης του πίνακα *P1* είναι

$$(\Sigma_{\Pi} + N_E * \mu_{\Pi} / M\Sigma) * \chi_{\sigma\epsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_E * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * N_E * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} \quad (32)$$

όπου N_E είναι το πλήθος των πλειάδων που ενημερώνονται. Το κόστος της εισαγωγής των ενημερωμένων πλειάδων του πίνακα *P1* στον πίνακα *Π* δίνεται από τον τύπο (16) ή τον τύπο (11), ανάλογα με το αν έχει ορισθεί πρωτεύον κλειδί ή όχι στον πίνακα *Π*, αντίστοιχα. Σύμφωνα με τα ανωτέρω, το κόστος ενημέρωσης ενός κανονικοποιημένου πίνακα *Π* με μία εντολή *UPDATE* που περιέχει την πρόταση *PORTION* είναι

$$\begin{aligned}\chi(\text{UPDATE}(\Pi, N_E, N_\Sigma)) = {} & (\Sigma_{\Pi} + N_E * \mu_{\Pi} / M\Sigma) * \chi_{\sigma\epsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_E * \mu_{\Pi} \\ & / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * N_E * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} + ((\Sigma_{\Pi} - N_E * \mu_{\Pi} / M\Sigma) / (M \\ & - 1) * ((N_E + N_\Sigma) * \mu_{\Pi} / PG - 2) + N_\Sigma * \mu_{\Pi} / M\Sigma + 2 * (N_E * \mu_{\Pi} / M\Sigma + 1)) \\ & * \chi_{\sigma\epsilon\lambda} + 2 * (N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma * (\log_{\Delta}((N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma / (2 * M) + \\ & 1) + 1) * \chi_{\sigma\epsilon\lambda} + (2 * N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (2 * N_E + \\ & N_\Sigma) * \chi_{\epsilon\pi\iota\beta}\end{aligned} \quad (33)$$

αν δεν έχει ορισθεί κλειδί πάνω στον πίνακα *Π* (N_Σ είναι το πλήθος των πλειάδων του πίνακα *Π* οι οποίες δεν υπόκεινται σε ενημέρωση και μπορούν να συμπτυχθούν με τις *ενημερωμένες* πλειάδες του πίνακα *P1*) ή ίσο με

$$\begin{aligned}\chi(\text{UPDATE}(\Pi, N_E, N_\Sigma)) = {} & (\Sigma_{\Pi} + N_E * \mu_{\Pi} / M\Sigma) * \chi_{\sigma\epsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_E * \mu_{\Pi} \\ & / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * N_E * \mu_{\Pi} / M\Sigma * \chi_{\sigma\epsilon\lambda} + (2 * (N_E * \mu'_{\Pi} / M\Sigma) * \\ & \log_{\Delta}((N_E * \mu'_{\Pi} / M\Sigma) / (2 * M) + 1) + N_E * (\mu_{\Pi} - \mu'_{\Pi}) / M\Sigma + (N_E * \mu_{\Pi} / \\ & M\Sigma) / (M - 1) * (N_E * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + (N_E * \mu_{\Pi} / M\Sigma / (M - 1) * \\ & (N_\Sigma * \mu_{\Pi} / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + ((\Sigma_{\Pi} - N_E * \mu_{\Pi} / M\Sigma) / (M - 1) * (N_E * \mu_{\Pi} / \\ & M\Sigma + N_E * \mu_{\Pi} / M\Sigma - 2) + N_\Sigma * \mu_{\Pi} / M\Sigma + 2 * (N_E * \mu_{\Pi} / M\Sigma + 1)) * \chi_{\sigma\epsilon\lambda} \\ & + 2 * (N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma * (\log_{\Delta}((N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma / (2 * M) + 1) + \\ & 1) * \chi_{\sigma\epsilon\lambda} + (2 * N_E + N_\Sigma) * \mu_{\Pi} / M\Sigma * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + (2 * N_E + N_\Sigma) * \\ & \chi_{\epsilon\pi\iota\beta}\end{aligned} \quad (34)$$

αν έχει ορισθεί κλειδί πάνω στον πίνακα *Π*.

2. *χρησιμοποιείται η πρόταση PORTION*. Στην περίπτωση αυτή, οι υπό ενημέρωση πλειάδες αποθηκεύονται σε έναν προσωρινό πίνακα *P1* και κατόπιν διαγράφονται από τον πίνακα *Π*, μέσω μιας σύνδεσής του με τον πίνακα *P1*. Ο πίνακας *P1* περιέχει μία επιπλέον στήλη *νέα_στήλη*, σε σχέση με τον πίνακα *Π*, η οποία θα είναι ίση με τη στήλη κανονικοποίησης του πίνακα *Π*. Στη συνέχεια, δημιουργείται ένας δεύτερος προσωρινός

πίνακας *P2*, στον οποίο το χρονικό διάστημα που περιέχεται στη στήλη *νέα_στήλη* έχει αντικατασταθεί από το κοινό τμήμα της τιμής του στον πίνακα *P1* και του διαστήματος που καθορίζεται στην πρόταση *PORTION*. Ακολούθως εκτελείται η πράξη *P3 = P1 PEXCEPT[νέα_στήλη] P2*, οι πλειάδες του πίνακα *P3* εισάγονται στον πίνακα *Π* (με τη στήλη που περιέχει τον χρόνο εγκυρότητας να αντικαθίσταται από τη στήλη *νέα_στήλη*) και ο πίνακας *P2* ενημερώνεται σύμφωνα με την πρόταση *SET*. Οι ενημερωμένες πλειάδες του πίνακα *P2* εισάγονται στον πίνακα *Π* (η στήλη που περιέχει τον χρόνο εγκυρότητας αντικαθίσταται από τη στήλη *νέα_στήλη*), ακολουθώντας τον κατάλληλο αλγεβρικό αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα (ανάλογα με το αν στον πίνακα *Π* έχει ορισθεί πρωτεύον κλειδί ή όχι).

Το κόστος της δημιουργίας του πίνακα *P1* και της διαγραφής των πλειάδων του πίνακα *P1* από τον πίνακα *Π* είναι

$$(\Sigma_{\Pi} + N_E * (\mu_{\Pi} + M\Delta) / M\Sigma) * \chi_{\sigma\epsilon\lambda} + (\Sigma_{\Pi} / (M - 1) * (N_E * (\mu_{\Pi} + M\Delta) / M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} \quad (35)$$

όπου N_E είναι το πλήθος των πλειάδων του πίνακα *Π* που υπόκεινται σε ενημέρωση. Το κόστος της δημιουργίας του πίνακα *P2* και της εκτέλεσης της πράξης *PEXCEPT* δίνεται από τον τύπο

$$2 * (N_E * (\mu_{\Pi} + M\Delta) / M\Sigma) * \chi_{\sigma\epsilon\lambda} + 4 * (N_E * (\mu_{\Pi} + M\Delta) / M\Sigma) * \log_{\Delta}((N_E * (\mu_{\Pi} + M\Delta) / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + 4 * ((N_E * (\mu_{\Pi} + M\Delta) / M\Sigma) * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + 2 * N_E * \chi_{\epsilon\pi\iota\beta} \quad (36)$$

ενώ το κόστος της ενημέρωσης του πίνακα *P2* είναι

$$2 * N_E * (\mu_{\Pi} + M\Delta) / M\Sigma * \chi_{\sigma\epsilon\lambda} \quad (37)$$

Δεδομένου ότι ο πίνακας *P3* περιέχει κατά μέγιστον $2 * N_E$ πλειάδες, το κόστος της εισαγωγής των πλειάδων του πίνακα *P3* στον πίνακα *Π* φράσσεται άνω από το όριο $N_E * (2 * \mu_{\Pi} + M\Delta) / M\Sigma * \chi_{\sigma\epsilon\lambda}$. Επίσης, όταν οι ενημερωμένες πλειάδες του πίνακα *P2* εισάγονται στον πίνακα *Π*, χρησιμοποιώντας τον αλγεβρικό αλγόριθμο εισαγωγής, ο πίνακας *Π* θα περιέχει κατά μέγιστον $N_{\Pi} + N_E$ πλειάδες. Σύμφωνα με τα ανωτέρω, και λαμβάνοντας υπόψη τις εξισώσεις (11) και (16), το κόστος του αλγεβρικού αλγόριθμου ενημέρωσης ενός κανονικοποιημένου πίνακα *Π*, με μία εντολή *UPDATE* που περιέχει την πρόταση *PORTION* είναι ίσο με

$$\chi(\text{UPDATE}(\Pi, N_E, N_\Sigma)) = (\Sigma_\Pi + N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_\Pi / (M - 1) * (N_E * (\mu_\Pi + M\Delta) / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + 4 * (N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \log_\Delta((N_E * (\mu_\Pi + M\Delta) / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + 4 * ((N_E * (\mu_\Pi + M\Delta) / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{\text{byte}}) + 2 * N_E * \chi_{\varepsilon\pi\iota\beta}) + 2 * N_E * (\mu_\Pi + M\Delta) / M\Sigma * \chi_{\sigma\varepsilon\lambda} + N_E * (2 * \mu_\Pi + M\Delta) / M\Sigma * \chi_{\sigma\varepsilon\lambda} + ((\Sigma_\Pi + N_E * \mu_\Pi / M\Sigma) / (M - 1) * ((N_E + N_\Sigma) * \mu_\Pi / M\Sigma - 2) + N_\Sigma * \mu_\Pi / M\Sigma + 2 * (N_E * \mu_\Pi / M\Sigma + 1)) * \chi_{\sigma\varepsilon\lambda} + 2 * ((N_E + N_\Sigma) * \mu_\Pi / M\Sigma) * (\log_\Delta((N_E + N_\Sigma) * \mu_\Pi / M\Sigma / (2 * M) + 1) + 1) * \chi_{\sigma\varepsilon\lambda} + (2 * N_E + N_\Sigma) * \mu_\Pi / M\Sigma * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{\text{byte}}) + (2 * N_E + N_\Sigma) * \chi_{\varepsilon\pi\iota\beta} \quad (38)$$

αν δεν έχει ορισθεί κλειδί πάνω στον πίνακα *Π* (N_Σ είναι το πλήθος των πλειάδων (και των *τμημάτων* πλειάδων) του πίνακα *Π* οι οποίες δεν υπόκεινται σε ενημέρωση και μπορούν να συμπτυχθούν με τις *ενημερωμένες* πλειάδες του πίνακα *P1*) ή ίσο με

$$\chi(\text{UPDATE}(\Pi, N_E, N_\Sigma)) = (\Sigma_\Pi + N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + (\Sigma_\Pi / (M - 1) * (N_E * (\mu_\Pi + M\Delta) / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \chi_{\sigma\varepsilon\lambda} + 4 * (N_E * (\mu_\Pi + M\Delta) / M\Sigma) * \log_\Delta((N_E * (\mu_\Pi + M\Delta) / M\Sigma) / (2 * M) + 1) * \chi_{\sigma\varepsilon\lambda} + (N_E * \mu_\Pi / M\Sigma / (M - 1) * (N_\Sigma * \mu_\Pi / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + 4 * (N_E * (\mu_\Pi + M\Delta) / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{\text{byte}}) + 2 * N_E * \chi_{\varepsilon\pi\iota\beta} + 2 * N_E * (\mu_\Pi + M\Delta) / M\Sigma * \chi_{\sigma\varepsilon\lambda} + N_E * (2 * \mu_\Pi + M\Delta) / M\Sigma * \chi_{\sigma\varepsilon\lambda} + (2 * (N_E * \mu_\Pi / M\Sigma) * \log_\Delta(N_E * \mu_\Pi / PG / (2 * M) + 1) + N_E * (\mu_\Pi - \mu'_\Pi) / M\Sigma + N_E * \mu_\Pi / M\Sigma / (M - 1) * (N_E * \mu_\Pi / M\Sigma - 1) + 1) * \chi_{\sigma\varepsilon\lambda} + ((\Sigma_\Pi + N_E * \mu_\Pi / M\Sigma) / (M - 1) * ((N_E + N_\Sigma) * \mu_\Pi / M\Sigma - 2) + N_\Sigma * \mu_\Pi / M\Sigma + 2 * (N_E * \mu_\Pi / M\Sigma + 1)) * \chi_{\sigma\varepsilon\lambda} + 2 * (N_E + N_\Sigma) * \mu_\Pi / M\Sigma * (\log_\Delta((N_E + N_\Sigma) * \mu_\Pi / M\Sigma) / (2 * M) + 1) + 1) * \chi_{\sigma\varepsilon\lambda} + ((2 * N_E + N_\Sigma) * \mu_\Pi / M\Sigma) * (\chi_{\sigma\varepsilon\lambda} + M\Sigma * \chi_{\text{byte}}) + (2 * N_E + N_\Sigma) * \chi_{\varepsilon\pi\iota\beta} \quad (39)$$

αν δεν έχει ορισθεί κλειδί πάνω στον πίνακα *Π*.

Στον βελτιστοποιημένο αλγόριθμο ενημέρωσης ενός κανονικοποιημένου πίνακα *Π* διακρίνονται οι ακόλουθες τέσσερις περιπτώσεις:

1. *δεν έχει ορισθεί κλειδί πάνω στον πίνακα Π και η πρόταση PORTION δεν χρησιμοποιείται*. Σ' αυτή την περίπτωση ο πίνακας *Π* διατρέχεται και οι πλειάδες που θα ενημερωθούν διαγράφονται απ' αυτόν και εισάγονται σε έναν προσωρινό πίνακα *P1*, τροποποιημένες όπως ορίζει η πρόταση *SET* της εντολής *UPDATE*. Ακολούθως, εφαρμόζεται ο βελτιστοποιημένος αλγόριθμος εισαγωγής δεδομένων σε κανονικοποιημένους πίνακες στους οποίους δεν έχει ορισθεί κλειδί, κατά την έναρξη

εκτέλεσης του οποίου ο πίνακας *Π* περιέχει $N_Π$ - N_E πλειάδες. Σύμφωνα με τα ανωτέρω και λαμβάνοντας υπόψη την εξίσωση (20), το κόστος του βελτιστοποιημένου αλγόριθμου ενημέρωσης ενός κανονικοποιημένου πίνακα *Π*, στον οποίο δεν έχει ορισθεί κλειδί, μέσω μιας εντολής *UPDATE* που δεν περιέχει την πρόταση *PORTION*, δίνεται από τον τύπο

$$\begin{aligned}\chi(\text{UPDATE}(Π, N_E, N_Σ)) = {} & Σ_Π * χ_{σελ} + N_E * μ_Π / MΣ * χ_{σελ} + 2 * N_E * (μ_Π * χ_{byte} + \\ & χ_{επιβ}) + ((Σ_Π - N_E * μ_Π / MΣ) / (M - 1) * (N_E * μ_Π / MΣ - 1) + 1) * χ_{σελ} + 2 \\ & * ((N_E + N_Σ) * μ_Π / MΣ) * \log_Δ((N_E + N_Σ) * μ_Π / MΣ / (2 * M) + 1) * χ_{σελ} \\ & + ((N_E + N_Σ) * μ_Π / MΣ) * (χ_{σελ} + MΣ * χ_{byte}) + N_E * μ_Π * (χ_{σελ} / MΣ + χ_{byte}) \\ & + 2 * N_E * μ_Π / MΣ * χ_{σελ} + (3 * N_Σ + 2 * N_E) * χ_{επιβ} + 2 * N_Σ * μ_Π * χ_{byte}\end{aligned} \quad (40)$$

Αφαιρώντας την εξίσωση (40) από την εξίσωση (33) έχουμε τη βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγόριθμου αντί του αλγεβρικού. Η βελτίωση αυτή είναι ίση με

$$\begin{aligned}χ_{αλγ} - χ_{βελτ} = {} & 2 * N_E * μ_Π / MΣ * χ_{σελ} + (Σ_Π / (M - 1) * (N_E * μ_Π / MΣ - 1) + 1) * \\ & χ_{σελ} + ((Σ_Π - N_E * μ_Π / MΣ) / (M - 1) * N_Σ * μ_Π / MΣ - 1) + N_Σ * μ_Π / MΣ + \\ & 2) * χ_{σελ} - 2 * (N_E + N_Σ) * μ_Π * χ_{byte} - 2 * (N_Σ + N_E) * χ_{επιβ}\end{aligned} \quad (41)$$

Οι θετικοί όροι αντιπροσωπεύουν (προσεγγιστικά) το κόστος δύο συνδέσεων στις οποίες μετέχει ο πίνακας *Π* και τριών μεταφορών πινάκων από/προς το δίσκο, ενώ οι αρνητικοί όροι αντιστοιχούν σε δύο μεταφορές πινάκων στη μνήμη (από τον πυρήνα του ΣΔΒΔ προς τον χρονολογικό φλοιό ή αντίστροφα), καταδεικνύοντας ότι η διαφορά απόδοσης υπέρ του βελτιστοποιημένου αλγορίθμου είναι σημαντική.

2. *έχει ορισθεί κλειδί πάνω στον πίνακα Π και η πρόταση PORTION δεν χρησιμοποιείται*. Η εκτέλεση της εντολής γίνεται όπως στην περίπτωση (1), με τη διαφορά ότι οι πλειάδες του πίνακα *P1* εισάγονται στον πίνακα *Π* χρησιμοποιώντας τον αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένους πίνακες, στους οποίους έχει ορισθεί κλειδί. Η βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγορίθμου αντί του αλγεβρικού, είναι ίση με τη διαφορά $χ_{αλγ} - χ_{βελτ}$ της εξίσωσης (41), προσαυξημένη κατά

$$(N_E * μ_Π / MΣ / (M - 1) * ((N_Σ + N_E) * μ_Π / MΣ - 2) + 2) * χ_{σελ} - N_E * (χ_{επιβ} + μ'_Π * χ_{byte}) \quad (42)$$

(σύμφωνα με τις εξισώσεις (11), (16), (20) και (25)). Ο θετικός όρος αντιπροσωπεύει το κόστος δύο συνδέσεων (μία σύνδεση μεταξύ του πίνακα *P1* και του εαυτού του και μία σύνδεση μεταξύ του πίνακα *P1* και του πίνακα *P2* -ο πίνακας *P2* περιέχει τις πλειάδες του πίνακα *Π* που δεν υπόκεινται σε ενημέρωση και μπορούν να συμπτυχθούν με τις

ενημερωμένες πλειάδες του πίνακα *P1*), ενώ ο αρνητικός όρος αντιστοιχεί στο κόστος μεταφοράς των κλειδιών του πίνακα *P1* στη μνήμη. Σύμφωνα με τα ανωτέρω, ο βελτιστοποιημένος αλγόριθμος αποφεύγει την εκτέλεση πέντε συνδέσεων και τριών μεταφορών σχέσεων από/προς το δίσκο, εισάγοντας τρεις μεταφορές σχέσεων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό ή αντίστροφα, επιφέροντας σημαντική βελτίωση της απόδοσης.

3. *δεν έχει ορισθεί κλειδί πάνω στον πίνακα Π και η πρόταση PORTION χρησιμοποιείται.* Σ' αυτή την περίπτωση ο αλγόριθμος διατρέχει τον πίνακα *Π*, επιλέγοντας τις πλειάδες που υπόκεινται σε ενημέρωση. Για κάθε πλειάδα που επιλέγεται, διαγράφεται από τον πίνακα *Π* το *τμήμα* της πλειάδας που θα ενημερωθεί (η διαγραφή τμήματος πλειάδας μπορεί να αντιστοιχεί είτε σε μία διαγραφή, είτε σε μία ενημέρωση, είτε σε μία ενημέρωση και μία εισαγωγή) και εισάγονται σε έναν προσωρινό πίνακα *P1*. Ακολούθως, εφαρμόζεται ο βελτιστοποιημένος αλγόριθμος εισαγωγής δεδομένων σε κανονικοποιημένους πίνακες στους οποίους δεν έχει ορισθεί κλειδί, κατά την έναρξη εκτέλεσης του οποίου ο πίνακας *Π* περιέχει κατά μέγιστον $N_Π + N_E$ πλειάδες. Σύμφωνα με τα ανωτέρω και λαμβάνοντας υπόψη την εξίσωση (20), το κόστος του βελτιστοποιημένου αλγόριθμου ενημέρωσης ενός κανονικοποιημένου πίνακα *Π*, στον οποίο δεν έχει ορισθεί κλειδί, μέσω μιας εντολής *UPDATE* που περιέχει την πρόταση *PORTION*, δίνεται από τον τύπο

$$\begin{aligned}\chi(\text{UPDATE}(\Pi, N_E, N_\Sigma)) = {} & (\Sigma_\Pi + N_E * \mu_\Pi / M\Sigma) * \chi_{\sigma\epsilon\lambda} + N_E * ((2 * \mu_\Pi - \mu'_\Pi) * \chi_{byte} \\ & + \chi_{\epsilon\pi\iota\beta}) + N_E * \mu_\Pi / M\Sigma * \chi_{\sigma\epsilon\lambda} + ((\Sigma_\Pi + N_E * \mu_\Pi / M\Sigma) / (M - 1) * (N_E * \mu_\Pi / \\ & M\Sigma - 1) + 1) * \chi_{\sigma\epsilon\lambda} + 2 * (N_E + N_\Sigma) * \mu_\Pi / M\Sigma * \log_\Delta((N_E + N_\Sigma) * \mu_\Pi / M\Sigma \\ & / (2 * M) + 1) * \chi_{\sigma\epsilon\lambda} + (N_E + N_\Sigma) * \mu_\Pi / M\Sigma * (\chi_{\sigma\epsilon\lambda} + M\Sigma * \chi_{byte}) + N_E * \mu_\Pi \\ & * (\chi_{\sigma\epsilon\lambda} / M\Sigma + \chi_{byte}) + 2 * (N_E * \mu_\Pi / M\Sigma) * \chi_{\sigma\epsilon\lambda} + (3 * N_\Sigma + 2 * N_E) * \chi_{\epsilon\pi\iota\beta} + \\ & 2 * N_\Sigma * \mu_\Pi * \chi_{byte}\end{aligned} \tag{43}$$

Αφαιρώντας την εξίσωση (43) από την εξίσωση (38) έχουμε τη βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγόριθμου αντί του αλγεβρικού. Η βελτίωση αυτή είναι ίση με

$$\chi_{αλγ} - \chi_{βελτ} = 2 * N_E * \mu_Π / MΣ * \chi_{σελ} + 5 * N_E * MΔ / MΣ * \chi_{σελ} + (\Sigma_Π / (M - 1) * (N_E * (\mu_Π + MΔ) / MΣ - 1) + 1) * \chi_{σελ} + 2 * (N_E * (\mu_Π + MΔ) / MΣ) * \chi_{σελ} + 4 * (N_E * (\mu_Π + MΔ) / MΣ) * \log_Δ((N_E * (\mu_Π + MΔ) / MΣ) / (2 * M) + 1) * \chi_{σελ} + 4 * (N_E * (\mu_Π + MΔ) / MΣ) * (\chi_{σελ} + MΣ * \chi_{byte}) + N_E * \chi_{επιβ} + N_E * \mu_Π / MΣ * \chi_{σελ} + (2 * (N_E * \mu_Π / MΣ) * \log_Δ(N_E * \mu_Π / MΣ / (2 * M) + 1) + N_E * (\mu_Π - \mu'_Π) / MΣ + N_E * \mu_Π / MΣ / (M - 1) * (N_E * M_Π / MΣ - 1) + 1) * \chi_{σελ} + ((\Sigma_Π + N_E * \mu_Π / MΣ) / (M - 1) * N_Σ * \mu_Π / MΣ - 2) + N_Σ * \mu_Π / MΣ + 2 * (N_E * \mu_Π / MΣ + 1)) * \chi_{σελ} - N_E * (2 * \mu_Π - \mu'_Π) * \chi_{byte} - 2 * N_Σ * \chi_{επιβ} - 2 * N_Σ * \mu_Π * \chi_{byte} \quad (44)$$

Οι θετικοί όροι αντιπροσωπεύουν τρεις συνδέσεις (σε δύο από τις οποίες συμμετέχει ο πίνακας *Π*), τρεις ταξινομήσεις και πολυάριθμες μεταφορές πινάκων από/προς το δίσκο, ενώ οι αρνητικοί όροι αντιπροσωπεύουν τρεις μεταφορές πινάκων από το σχεσιακό ΣΔΒΔ στον χρονολογικό φλοιό (ή αντίστροφα). Είναι σαφές ότι το άθροισμα των θετικών όρων είναι κατά πολύ μεγαλύτερο του αθροίσματος των αρνητικών όρων, καταδεικνύοντας ότι η χρήση του βελτιστοποιημένου αλγορίθμου επιφέρει σημαντική βελτίωση στην απόδοση.

4. *έχει ορισθεί κλειδί πάνω στον πίνακα και χρησιμοποιείται η πρόταση PORTION*. Η εκτέλεση της εντολής γίνεται όπως στην περίπτωση (3), με τη διαφορά ότι οι πλειάδες του πίνακα *P1* εισάγονται στον πίνακα *Π* χρησιμοποιώντας τον βελτιστοποιημένο αλγόριθμο εισαγωγής δεδομένων σε κανονικοποιημένους πίνακες, στους οποίους έχει ορισθεί κλειδί. Η βελτίωση που προκύπτει από τη χρήση του βελτιστοποιημένου αλγορίθμου αντί του αλγεβρικού, είναι ίση με τη διαφορά $\chi_{αλγ} - \chi_{βελτ}$ της εξίσωσης (44), προσαυξημένη κατά

$$(N_E * \mu_Π / MΣ / (M - 1) * ((N_Σ + N_E) * \mu_Π / MΣ - 2) + 2) * \chi_{σελ} - N_E * (\chi_{επιβ} + \mu'_Π * \chi_{byte}) \quad (45)$$

(σύμφωνα με τις εξισώσεις (11), (16), (20) και (24)). Ο θετικός όρος αντιπροσωπεύει το κόστος δύο συνδέσεων (μία σύνδεση μεταξύ του πίνακα *P1* και του εαυτού του και μία σύνδεση μεταξύ του πίνακα *P1* και του πίνακα *P2* -ο πίνακας *P2* περιέχει τις πλειάδες (και τα *τμήματα* πλειάδων) του πίνακα *Π* που δεν υπόκεινται σε ενημέρωση και μπορούν να συμπτυχθούν με τις *ενημερωμένες* πλειάδες του πίνακα *P1*), ενώ ο αρνητικός όρος αντιστοιχεί στο κόστος μεταφοράς των κλειδιών του πίνακα *P1* στη μνήμη. Σύμφωνα με τα ανωτέρω, ο βελτιστοποιημένος αλγόριθμος αποφεύγει την εκτέλεση πέντε συνδέσεων, τριών ταξινομήσεων και πολυάριθμων μεταφορών σχέσεων από/προς το

δίσκο, εισάγοντας τέσσερις μεταφορές σχέσεων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό ή αντίστροφα, επιφέροντας σημαντική βελτίωση της απόδοσης.

4.4.3. Διαγράμματα απόδοσης.

Στην παράγραφο 4.4.2 παρουσιάσαμε ένα μαθηματικό μοντέλο, βάσει του οποίου συγκρίθηκαν οι αλγεβρικοί και οι βελτιστοποιημένου αλγόριθμοι εκτέλεσης των εντολών *INSERT*, *DELETE* και *UPDATE*. Στην παράγραφο αυτή, παρουσιάζουμε συγκριτικά διαγράμματα απόδοσης των αλγεβρικών και των βελτιστοποιημένων αλγορίθμων. Τα στοιχεία απόδοσης προέκυψαν από μετρήσεις που έγιναν σε υλοποιήσεις των αλγορίθμων, οι οποίες αναπτύχθηκαν σε ένα SUN SPARC IPC, με λειτουργικό σύστημα SunOS 4.1.3 και 24 MB μνήμης. Το σχεσιακό ΣΔΒΔ που χρησιμοποιήθηκε ήταν το Ingres 6.4, ο πυρήνας του οποίου είχε επεκταθεί για να υποστηρίζει τον τύπο δεδομένων DATEINTERVAL, καθώς και συναρτήσεις που δέχονται ή επιστρέφουν αποτελέσματα αυτού του τύπου. Στους πίνακες που χρησιμοποιήθηκαν στις μετρήσεις δεν είχαν ορισθεί δείκτες, ενώ το σχήμα φυσικής αποθήκευσής τους ήταν το σχήμα *σωρού* (απλό σειριακό αρχείο).

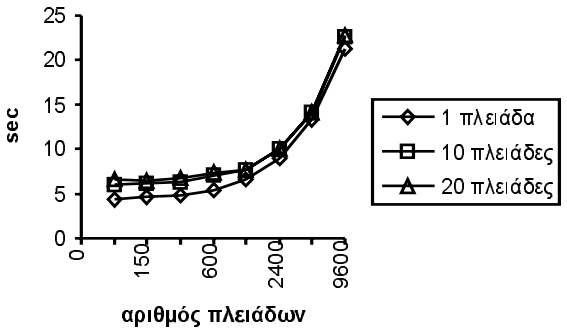
Κάθε ένα από τα διαγράμματα απεικονίζει μία οικογένεια καμπυλών. Κάθε καμπύλη αναπαριστά τον απαιτούμενο χρόνο εκτέλεσης του αλγορίθμου (κατακόρυφος άξονας) για κάποια συγκεκριμένη λειτουργία, συναρτήσει του αριθμού πλειάδων που περιέχονται στον πίνακα, πάνω στον οποίο εκτελείται η λειτουργία (οριζόντιος άξονας).

I. Εισαγωγή δεδομένων.

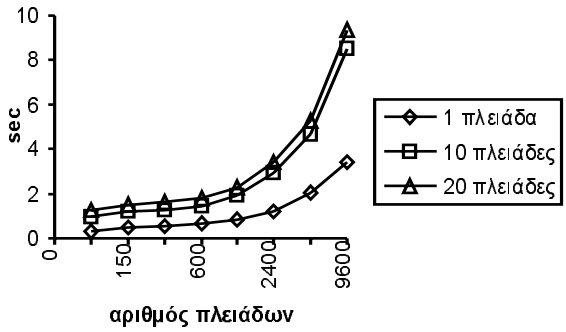
Στα σχήματα 4.5 έως 4.12 παρουσιάζονται συγκριτικά διαγράμματα απόδοσης του αλγεβρικού και του βελτιστοποιημένου αλγόριθμου εισαγωγής δεδομένων σε κανονικοποιημένο πίνακα. Τα διαγράμματα χωρίζονται σε δύο ομάδες, η πρώτη από τις οποίες (σχήματα 4.5 έως 4.8) αφορά την εισαγωγή δεδομένων σε πίνακα στον οποίο δεν έχει ορισθεί πρωτεύον κλειδί, ενώ η δεύτερη (σχήματα 4.9 έως 4.12) αφορά την εισαγωγή δεδομένων σε πίνακα στον οποίο έχει ορισθεί πρωτεύον κλειδί.

Στα δύο πρώτα διαγράμματα της κάθε ομάδας, η εισαγωγή νέων πλειάδων δεν επηρεάζει καμία από τις υπάρχουσες πλειάδες του πίνακα (δηλαδή καμία από τις εισαγόμενες πλειάδες δεν μπορεί να συμπτυχθεί με κάποια από τις πλειάδες που βρίσκονται ήδη καταχωρισμένες στον πίνακα), ενώ στα δύο επόμενα κάθε μία από τις ν εισαγόμενες πλειάδες μπορεί να συμπτυχθεί με μ από τις πλειάδες που δεν ενημερώνονται (συμβολίζεται με (ν, μ) στο υπόμνημα του διαγράμματος).

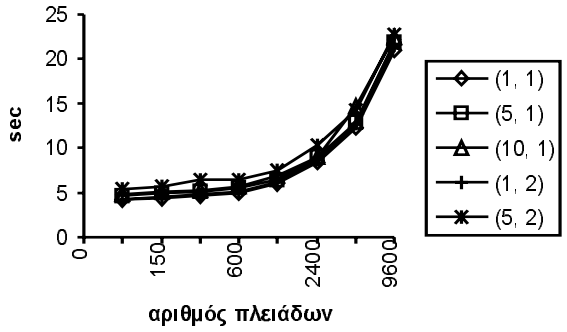
Στις περιπτώσεις όπου εισάγονται περισσότερες από μία πλειάδες (και κατά συνέπεια έχει χρησιμοποιηθεί ερώτηση για τον προσδιορισμό των προς εισαγωγή δεδομένων), στα διαγράμματα απεικονίζεται μόνο ο χρόνος εισαγωγής των δεδομένων στον πίνακα και όχι ο χρόνος αποτίμησης της ερώτησης.



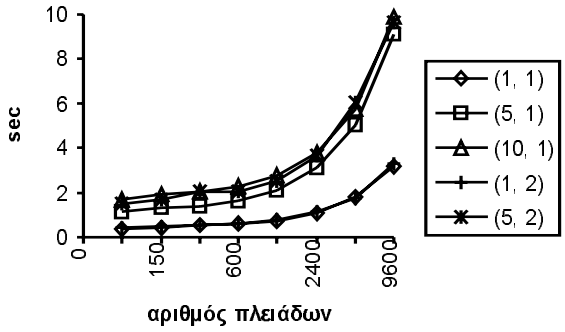
Σχήμα 4.5 - Απόδοση του αλγεβρικού αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.



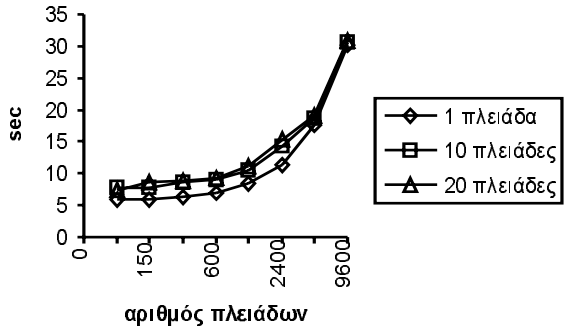
Σχήμα 4.6 - Απόδοση του βελτιστοποιημένου αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.

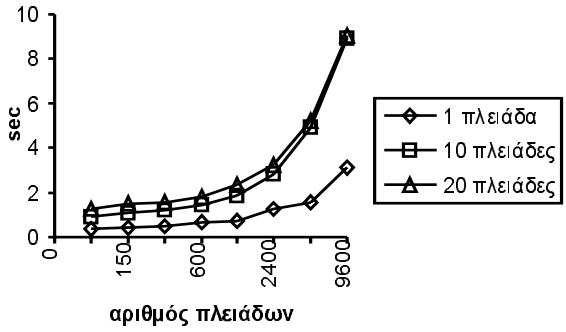


Σχήμα 4.7 - Απόδοση του αλγεβρικού αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.

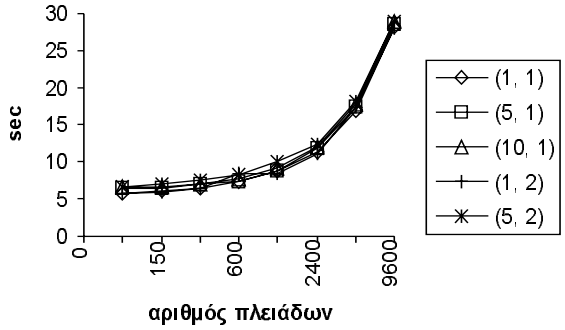


Σχήμα 4.8 - Απόδοση του βελτιστοποιημένου αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.

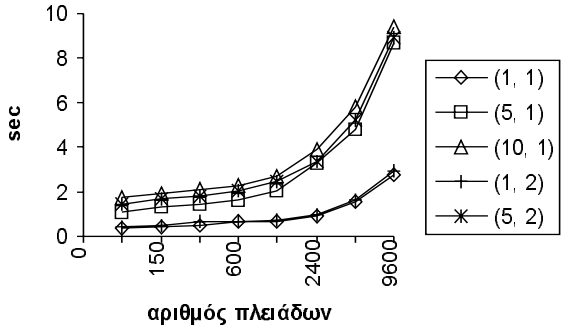




Σχήμα 4.9 - Απόδοση του αλγεβρικού αλγόριθμου εισαγωγής δεδομένων σε πίνακα με κλειδί.



Σχήμα 4.10 - Απόδοση του βελτιστοποιημένου αλγόριθμου εισαγωγής δεδομένων σε πίνακα με κλειδί.



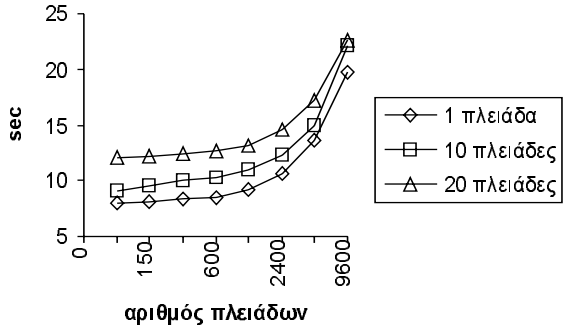
Σχήμα 4.11 - Απόδοση του αλγεβρικού αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.

Σχήμα 4.12 - Απόδοση του βελτιστοποιημένου αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.

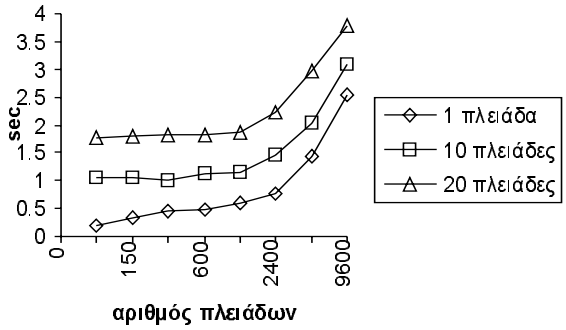
Όπως προκύπτει από τα διαγράμματα, ο βελτιστοποιημένος αλγόριθμος εισαγωγής δεδομένων παρουσιάζει βελτιωμένη απόδοση, σε σχέση με τον αλγεβρικό, σε ποσοστό που κυμαίνεται μεταξύ 220% και 1510%.

II. Διαγραφή δεδομένων.

Στα σχήματα 4.13 έως 4.14 παρουσιάζονται συγκριτικά διαγράμματα απόδοσης του αλγεβρικού και του βελτιστοποιημένου αλγόριθμου διαγραφής δεδομένων από κανονικοποιημένο πίνακα, όταν χρησιμοποιείται η πρόταση *PORTION*. Δεδομένου ότι η απόδοση των αλγορίθμων διαγραφής δεν επηρεάζεται από το αν έχει ορισθεί πρωτεύον κλειδί πάνω στον πίνακα από τον οποίο διαγράφονται τα δεδομένα, παρουσιάζεται μόνο ένα διάγραμμα για κάθε αλγόριθμο.



Σχήμα 4.13 - Απόδοση του αλγεβρικού αλγόριθμου διαγραφής δεδομένων.

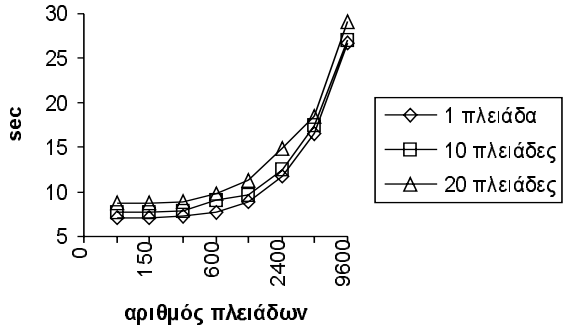


Σχήμα 4.14 - Απόδοση του βελτιστοποιημένου αλγόριθμου διαγραφής δεδομένων.

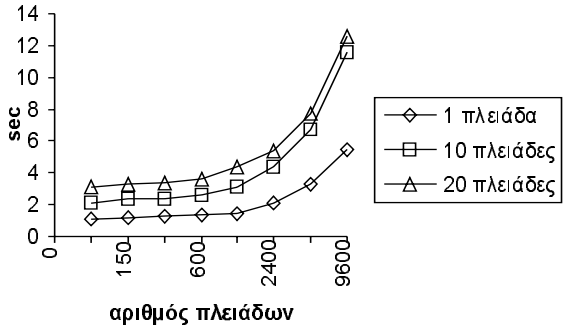
Όπως προκύπτει από τα διαγράμματα, ο βελτιστοποιημένος αλγόριθμος διαγραφής δεδομένων παρουσιάζει βελτιωμένη απόδοση, σε σχέση με τον αλγεβρικό, σε ποσοστό που κυμαίνεται μεταξύ 550% και 2420%.

III. Ενημέρωση δεδομένων.

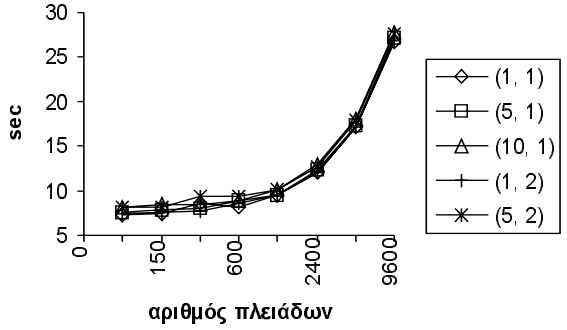
Στα σχήματα 4.15 έως 4.26 παρουσιάζονται συγκριτικά διαγράμματα απόδοσης του αλγεβρικού και του βελτιστοποιημένου αλγόριθμου ενημέρωσης κανονικοποιημένου πίνακα. Τα διαγράμματα χωρίζονται σε δύο ομάδες, η πρώτη από τις οποίες (σχήματα 4.15 έως 4.20) αφορά την ενημέρωση κανονικοποιημένου πίνακα στον οποίο δεν έχει ορισθεί πρωτεύον κλειδί ενώ η δεύτερη ομάδα (σχήματα 4.21 έως 4.26) αφορά την ενημέρωση κανονικοποιημένου πίνακα στον οποίο έχει ορισθεί πρωτεύον κλειδί. Στα δύο πρώτα διαγράμματα της κάθε ομάδας, δεν χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και οι ενημερωμένες πλειάδες δεν μπορούν να συμπτυχθούν με καμία από τις πλειάδες που δεν ενημερώνονται. Στο τρίτο και το τέταρτο διάγραμμα της κάθε ομάδας, δεν χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και κάθε μία από τις *ν* πλειάδες που ενημερώνονται, μπορεί να συμπτυχθεί με *μ* από τις πλειάδες που δεν ενημερώνονται (συμβολίζεται με *(ν, μ)* στο υπόμνημα του διαγράμματος). Στα δύο τελευταία διαγράμματα της κάθε ομάδας χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και οι ενημερωμένες πλειάδες δεν μπορούν να συμπτυχθούν με καμία από τις πλειάδες που δεν ενημερώνονται.



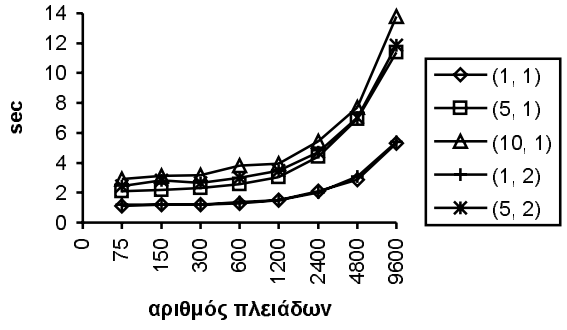
Σχήμα 4.15 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



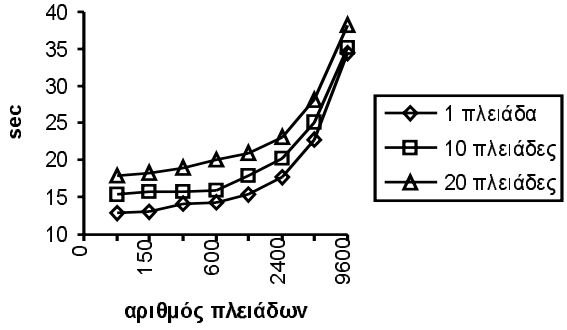
Σχήμα 4.16 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



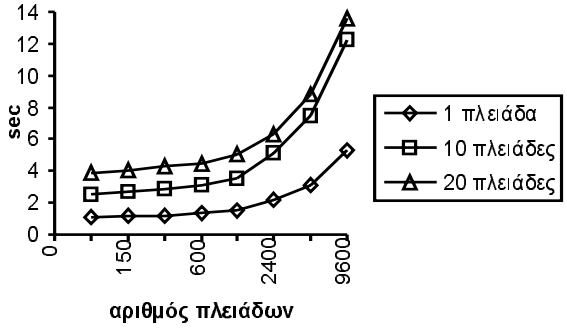
Σχήμα 4.17 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



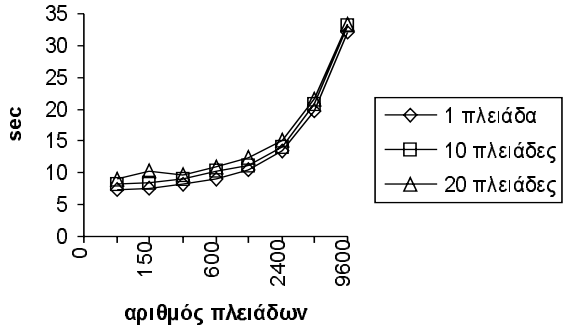
Σχήμα 4.18 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



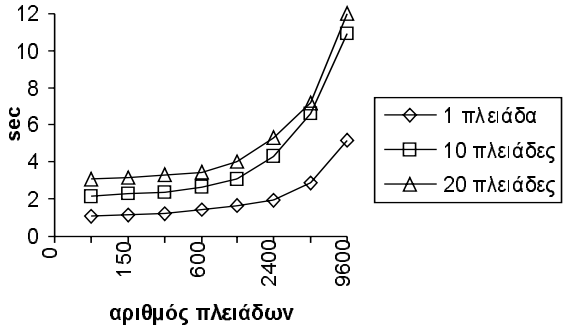
Σχήμα 4.19 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, με χρήση της πρότασης *PORTION*.



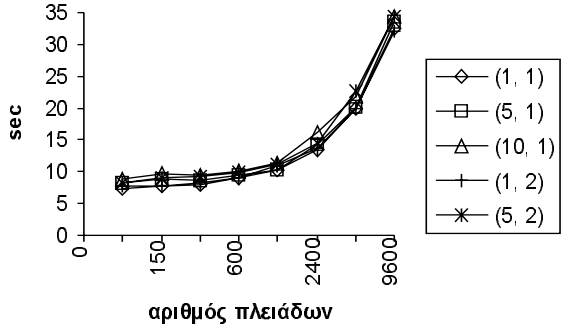
Σχήμα 4.20 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, με χρήση της πρότασης *PORTION*.



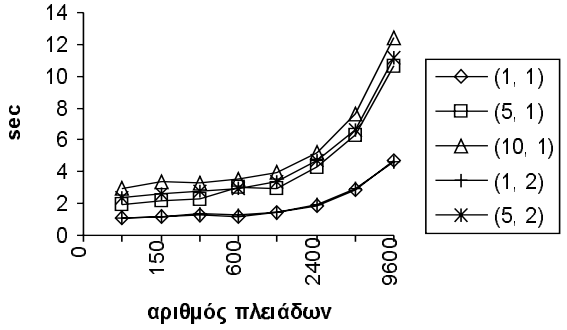
Σχήμα 4.21 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



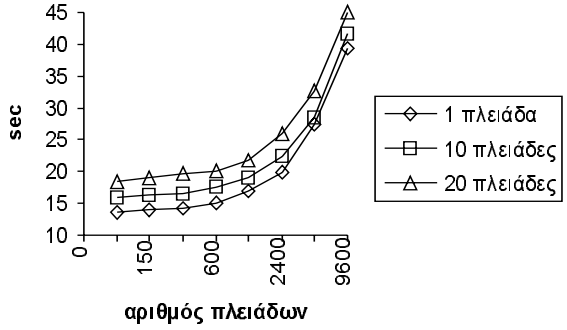
Σχήμα 4.22 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



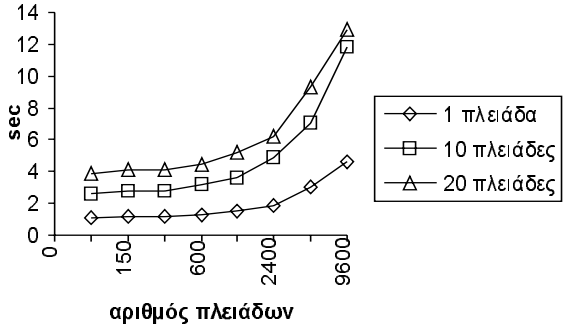
Σχήμα 4.23 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



Σχήμα 4.24 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



Σχήμα 4.25 - Απόδοση του αλγεβρικού αλγόριθμου ενημέρωσης πίνακα με κλειδί, με χρήση της πρότασης *PORTION*.



Σχήμα 4.26 - Απόδοση του βελτιστοποιημένου αλγόριθμου ενημέρωσης πίνακα με κλειδί, με χρήση της πρότασης *PORTION*.

Όπως προκύπτει από τα διαγράμματα, ο βελτιστοποιημένος αλγόριθμος ενημέρωσης κανονικοποιημένων πινάκων παρουσιάζει βελτιωμένη απόδοση, σε σχέση με τον αλγεβρικό, σε ποσοστό που κυμαίνεται μεταξύ 185% και 1230%.

4.5. Σύνοψη.

Στο κεφάλαιο αυτό παρουσιάσαμε μία βελτιστοποιημένη υλοποίηση ενός χρονολογικού ΣΔΒΔ. Το χρονολογικό ΣΔΒΔ αποτελείται από δύο τμήματα, ένα σχεσιακό ΣΔΒΔ που είναι υπεύθυνο για την αποθήκευση και την ανάκτηση δεδομένων από τους δίσκους και έναν χρονολογικό φλοιό, που έχει την ευθύνη υποστήριξης της χρονολογικής σημασιολογίας. Αναλύσαμε τους αλγόριθμους εκτέλεσης των εντολών και τους συγκρίναμε με αλγεβρικούς αλγόριθμους, τόσο βάσει ενός μαθηματικού μοντέλου, όσο και με διαγράμματα απόδοσης. Στο επόμενο κεφάλαιο θα παρουσιάσουμε μεθόδους εμπλουτισμού του χρονολογικού ΣΔΒΔ με δυνατότητες υποστήριξης δοσοληψιών (*transactions*) σε επίπεδο χρήστη και πολλών ταυτόχρονων χρηστών.

5. Εμπλουτισμός του χρονολογικού ΣΔΒΔ.

Δύο πολύ σημαντικές δυνατότητες ενός ΣΔΒΔ είναι η υποστήριξη *δοσοληψιών* (transactions) και *ταυτοχρόνων χρηστών* (concurrent users). Οι δοσοληψίες είναι θεμελιώδες συστατικό για τους ακόλουθους λόγους:

1. αποτελούν την ενότητα *ακεραιότητας* (integrity- [Date85]), επιτρέποντας στη βάση δεδομένων να μεταβεί από μία έγκυρη κατάσταση σε μία άλλη, περνώντας προσωρινά από μία μη έγκυρη κατάσταση. Για παράδειγμα, για τη μεταφορά ενός ποσού από έναν τραπεζικό λογαριασμό σε έναν άλλο, αρχικά χρεώνεται ο πρώτος λογαριασμός με το μεταφερόμενο ποσό και στη συνέχεια πιστώνεται ο δεύτερος με το ίδιο ποσό. Μεταξύ των δύο ενημερώσεων, η βάση δεδομένων βρίσκεται σε μη έγκυρη κατάσταση, η οποία όμως δεν είναι ορατή στις άλλες δοσοληψίες που τυχόν εκτελούνται.
2. συνιστούν την ενότητα *διαμοιρασμού*, δεδομένου ότι τα κλειδώματα που αποκτώνται κατά τη διάρκεια μιας συνόδου (session) απομακρύνονται στο τέλος της τρέχουσας δοσοληψίας.
3. αποτελούν την ενότητα *επανόρθωσης* (recovery), μια και, σε περίπτωση κατάρρευσης του ΣΔΒΔ (λόγω εσωτερικού προβλήματος ή εξωτερικής αιτίας, π.χ. κατάρρευσης του υπολογιστή στον οποίο αυτό εκτελείται), η βάση δεδομένων επαναφέρεται στη κατάσταση που βρισκόταν σε κάποιο σημείο επικύρωσης (commit point).
4. παρέχουν τη δυνατότητα αναίρεσης εσφαλμένων τροποποιήσεων και υλοποιούν μία αφαίρεση *ατομικών λειτουργιών*, δηλαδή λειτουργιών που εκτελούνται είτε στο σύνολό τους είτε καθόλου ([Tanenbaum92]), η οποία διευκολύνει τον προγραμματισμό.

Η δυνατότητα υποστήριξης ταυτοχρόνων χρηστών είναι επίσης σημαντική, μια και αυξάνει το βαθμό παραλληλισμού εργασιών, μειώνοντας, κατά συνέπεια, τον μέσο χρόνο απόκρισης στις ερωτήσεις, παρέχοντας προστασία από αλληλοπαρεμβολές των τροποποιήσεων, οι οποίες θα οδηγούσαν τη βάση δεδομένων σε ασυνεπή (inconsistent) κατάσταση.

Η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών, σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ, όπως αυτό που περιγράψαμε στο κεφάλαιο 4, δεν μπορεί να βασιστεί μόνο στους μηχανισμούς που παρέχονται από το σχεσιακό ΣΔΒΔ για τους εξής λόγους:

1. σε πολλές περιπτώσεις, ο χρονολογικός φλοιός πρέπει να δημιουργήσει προσωρινούς πίνακες, στους οποίους αποθηκεύονται ενδιάμεσα αποτελέσματα (π.χ. ο πίνακας που περιέχει τις προς εισαγωγή πλειάδες, κατά την εκτέλεση της εντολής *INSERT* και ο

πίνακας που περιέχει τις ενημερωμένες πλειάδες, κατά την εκτέλεση της εντολής *UPDATE*, σύμφωνα με τους αλγόριθμους που περιγράφηκαν στο κεφάλαιο 4). Σε μερικά ΣΔΒΔ (π.χ. Oracle), η εκτέλεση μιας εντολής της γλώσσας ορισμού δεδομένων (όπως οι εντολές *CREATE TABLE* και *DROP TABLE*) εισάγει ένα *σημείο αυτόματης επικύρωσης* (implicit commit point) της τρέχουσας δοσοληψίας ([Oracle90]), με συνέπεια να είναι αδύνατη η αναίρεση των τροποποιήσεων που έχουν επέλθει πριν από την εκτέλεση της εντολής στη βάση δεδομένων, με μία εντολή *ROLLBACK*. Σε άλλα ΣΔΒΔ (π.χ. Sybase), η χρήση των εντολών της γλώσσας ορισμού δεδομένων μέσα σε δοσοληψίες απαγορεύεται συνολικά ([Sybase90]).

2. η εγγραφή αποτελεσμάτων σε προσωρινούς πίνακες συνιστά τροποποίηση της βάσης και, κατά συνέπεια, οι αλλαγές αυτές καταχωρίζονται στο ημερολόγιο (log) της βάσης. Δεδομένου ότι οι αλγόριθμοι εκτέλεσης των εντολών της SQL-XE παράγουν σημαντικό όγκο ενδιαμέσων αποτελεσμάτων, το μέγεθος του ημερολογίου αυξάνεται σημαντικά, καθιστώντας απαραίτητη τη χρήση τεχνικών για μείωση των απαιτήσεων σε χώρο αποθήκευσης για το ημερολόγιο.
3. τα κλειδώματα που τίθενται κατά τη διάρκεια μιας συνόδου θα πρέπει να μην αίρονται στα σημεία αυτόματης επικύρωσης (που εισάγονται όπως περιγράψαμε ανωτέρω), αλλά ούτε και στα σημεία επικύρωσης που χρησιμοποιούνται για τον περιορισμό του μεγέθους του ημερολογίου.

Στις επόμενες παραγράφους κεφάλαιο, παρουσιάζονται τεχνικές που επιτρέπουν την υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ.

5.1. Υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών.

Οι δοσοληψίες και η ύπαρξη ταυτοχρόνων χρηστών μπορούν να υλοποιηθούν σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ, με χρήση δύο συνόδων ανάμεσα στον χρονολογικό φλοιό και το σχεσιακό ΣΔΒΔ, της *συνόδου χρήστη* και της *συνόδου συστήματος*. Η σύνοδος χρήστη χρησιμοποιείται για την προσπέλαση των πινάκων χρηστών (πίνακες που έχει δημιουργήσει ο χρήστης ή άλλοι χρήστες της ίδιας βάσης δεδομένων), τόσο για ανάγνωση όσο και για εγγραφή. Οι προσωρινοί πίνακες, που είναι απαραίτητοι για την εκτέλεση των εντολών της SQL-XE, δημιουργούνται μέσω της συνόδου συστήματος, ενώ η ίδια σύνοδος χρησιμοποιείται και για την εισαγωγή και την τροποποίηση των δεδομένων σ' αυτούς τους

πίνακες. Τα δεδομένα που περιέχονται στους προσωρινούς πίνακες μπορούν να διαβαστούν τόσο από τη σύνοδο χρήστη, όσο και από τη σύνοδο συστήματος.

Δεδομένου ότι οι επικυρώσεις που λαμβάνουν χώρα για μία σύνοδο δεν επηρεάζουν τις άλλες συνόδους, σε ό,τι αφορά τη δική τους επικύρωση, οι επικυρώσεις (αυτόματες ή όχι) που λαμβάνουν χώρα για τη σύνοδο συστήματος δεν επηρεάζουν την κατάσταση της συνόδου χρήστη. Ένα σημείο που χρήζει προσοχής, είναι ότι διαφορετικές σύνοδοι δρούν ανταγωνιστικά για τα κλειδώματα στη βάση δεδομένων: τα δεδομένα που κλειδώνονται με διαμοιραζόμενα ή αποκλειστικά κλειδώματα από μία σύνοδο, δεν είναι διαθέσιμα για ενημέρωση ή ανάγνωση, αντίστοιχα, από τις άλλες συνόδους, κατά συνέπεια πρέπει να ληφθεί πρόνοια ώστε το σχήμα προσπέλασης της βάσης δεδομένων, μέσω των δύο συνόδων, να μην οδηγεί σε αδιέξοδα.

Στις ακόλουθες παραγράφους περιγράφονται αναλυτικά η χρήση των δύο συνόδων και των σχημάτων κλειδώματος που χρησιμοποιούνται, προκειμένου να υποστηριχθούν δοσοληψίες και ταυτόχρονοι χρήστες.

5.1.1. Εκτέλεση της εντολής SELECT.

Για την εκτέλεση της εντολής *SELECT* διακρίνονται τρεις περιπτώσεις:

1. *η εντολή SELECT (στη μορφή που έχει μετασχηματιστεί από τον συντακτικό αναλυτή) δεν απαιτεί την εφαρμογή πράξεων που δεν υποστηρίζονται από το σχεσιακό ΣΔΒΔ (REFORMAT, PUNION κ.τ.λ.)*. Σ' αυτή την περίπτωση, ο χρονολογικός φλοιός ορίζει μέσω της συνόδου χρήστη έναν δρομέα, ο οποίος ανακτά τα αποτελέσματα της ερώτησης, που στη συνέχεια παρουσιάζονται στον χρήστη. Το σχεσιακό ΣΔΒΔ θα θέσει τα απαραίτητα κλειδώματα για τις ανακτηθείσες πλειάδες, προφυλάσσοντάς τες από τροποποιήσεις μέσω άλλων συνόδων (που ενδεχομένως ανήκουν σε διαφορετικούς χρήστες της βάσης), ενόσω η τρέχουσα δοσοληψία της συνόδου χρήστη είναι ενεργός (δηλαδή μέχρι να επικυρωθεί ή να ακυρωθεί). Τέλος, μια και καμία τροποποίηση δεν γίνεται στη βάση δεδομένων μέσω της συνόδου χρήστη, το μέγεθος του ημερολογίου της δεν αυξάνεται.
2. *η εντολή SELECT αποτελείται από μία μόνο επεκταμένη ερώτηση*. Στην περίπτωση αυτή, για την αποτίμηση της ερώτησης ακολουθούνται τα εξής βήματα:
   i) το τμήμα της επεκταμένης ερώτησης που είναι σύμφωνο με τις προδιαγραφές της SQL89 (δηλαδή οι προτάσεις *SELECT, FROM, WHERE, GROUP BY* και *HAVING*) χρησιμοποιείται για να κατασκευαστεί μία ερώτηση προς το σχεσιακό ΣΔΒΔ, την

οποία συμβολίζουμε με *ερώτηση-SQL*. Μέσω της συνόδου συστήματος, δημιουργείται ένας προσωρινός πίνακας (τον οποίο θα συμβολίζουμε με *P1*- το πραγματικό του όνομα είναι μία μοναδική συμβολοσειρά, για τον υπολογισμό της οποίας υπεύθυνος είναι ο χρονολογικός φλοιός), του οποίου το σχήμα είναι ίδιο με το σχήμα του αποτελέσματος της *ερώτησης-SQL* και μέσω της συνόδου χρήστη ορίζεται ένας δρομέας, ο οποίος ανακτά τα αποτελέσματα της *ερώτησης-SQL*.

ii) ο δρομέας που ορίστηκε στο βήμα (i) χρησιμοποιείται για να ανακτηθούν οι πλειάδες του αποτελέσματος της *ερώτησης-SQL*, οι οποίες εν συνεχεία εισάγονται στον πίνακα *P1*, μέσω της συνόδου συστήματος. Όταν εισαχθούν όλες οι πλειάδες του αποτελέσματος στον πίνακα *P1*, η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται, με αποτέλεσμα την περικοπή του ημερολογίου της συνόδου συστήματος.

iii) οι προτάσεις *REFORMAT* και *NORMALISE* της επεκταμένης ερώτησης εκτελούνται μέσω της συνόδου συστήματος, δηλαδή καλούνται οι κατάλληλες διαδικασίες της βιβλιοθήκης της ΣΑΧΕ. Κάθε κλήση, επεξεργάζεται έναν πίνακα εισόδου (η πρώτη κλήση επεξεργάζεται τον πίνακα *P1*) και παράγει έναν πίνακα εξόδου. Όταν μία κλήση διαδικασίας βιβλιοθήκης της ΣΑΧΕ ολοκληρώνεται, ο πίνακας εισόδου που αυτή επεξεργάστηκε καταστρέφεται, μέσω της συνόδου συστήματος και η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται.

iv) οι πλειάδες που περιέχονται στον τελευταίο προσωρινό πίνακα ανακτώνται μέσω της συνόδου συστήματος και παρουσιάζονται στον χρήστη. Όταν ανακτηθούν όλες οι πλειάδες, ο τελευταίος προσωρινός πίνακας καταστρέφεται, μέσω της συνόδου συστήματος και η δοσοληψία της συνόδου συστήματος επικυρώνεται.

Ο αλγόριθμος που περιγράφηκε ανωτέρω δεν εισάγει αυτόματα σημεία επικύρωσης για τη σύνοδο χρήστη, καθώς η δημιουργία και η καταστροφή των προσωρινών πινάκων γίνεται μέσω της συνόδου συστήματος. Το ημερολόγιο της συνόδου χρήστη δεν αυξάνεται σε μέγεθος, δεδομένου ότι καμία τροποποίηση δεν γίνεται στη βάση δεδομένων μέσω της συνόδου αυτής. Η αύξηση του μεγέθους του ημερολογίου της συνόδου συστήματος είναι προσωρινή (καθώς αυτό περικόπτεται σε κάθε σημείο επικύρωσης της συνόδου συστήματος) και σχετικά μικρή (το μέγιστο μέγεθός του είναι αυτό που απαιτείται για να καταχωριστούν οι αλλαγές που επιφέρει στη βάση δεδομένων μία διαδικασία βιβλιοθήκης της ΣΑΧΕ). Κατά το βήμα (i) το ΣΔΒΔ θα τοποθετήσει τα κατάλληλα κλειδώματα για τις πλειάδες που μετέχουν στον υπολογισμό του τελικού

αποτελέσματος, εξασφαλίζοντας ότι δεν θα τροποποιηθούν μέσω άλλων συνόδων, ενόσω η τρέχουσα δοσοληψία της συνόδου χρήστη είναι ενεργός. Τέλος, δοθέντος ότι οι πίνακες των χρηστών προσπελαύνονται μόνο μέσω της συνόδου χρήστη και οι προσωρινοί πίνακες προσπελαύνονται μόνο μέσω της συνόδου συστήματος, ο αλγόριθμος είναι απαλλαγμένος από αδιέξοδα.

3. *η εντολή SELECT περιλαμβάνει δύο επεκταμένες ερωτήσεις, των οποίων τα αποτελέσματα συνδυάζονται μέσω μιας πράξης UNION, UNION ALL, EXCEPT, PUNION ή PEXCEPT της ΣΑΧΕ.* Στην περίπτωση αυτή η διαδικασία αποτίμησης της ερώτησης έχει ως ακολούθως:

   i) εκτελούνται τα βήματα (i) έως (iii) της περίπτωσης (2) για να υπολογιστεί το αποτέλεσμα των δύο επεκταμένων ερωτήσεων. Το αποτέλεσμα της κάθε επεκταμένης ερώτησης αποθηκεύεται σε έναν προσωρινό πίνακα, ο οποίος δημιουργείται μέσω της συνόδου συστήματος, όπως περιγράφεται στο βήμα (2) (οι πίνακες που περιέχουν τα αποτελέσματα της πρώτης και της δεύτερης επεκταμένης ερώτησης θα συμβολίζονται με *T1* και *T2*, αντίστοιχα).

   ii) αν τα αποτελέσματα των επεκταμένων ερωτήσεων συνδυάζονται μέσω της πράξης *UNION* ή της πράξης *UNION ALL* της ΣΑΧΕ, ορίζεται, μέσω της συνόδου συστήματος, ένας δρομέας, ο οποίος ανακτά την ένωση των πλειάδων των πινάκων *T1* και *T2*, με ενδεχόμενη απομάκρυνση των διπλοτύπων πλειάδων (οι διπλότυπες πλειάδες απομακρύνονται αν η πράξη συνδυασμού είναι η *UNION*) και ταξινόμηση (αν η εντολή *SELECT* που έδωσε ο χρήστης περιείχε την πρόταση *ORDER BY*). Οι ανακτώμενες πλειάδες παρουσιάζονται στον χρήστη και ο αλγόριθμος τερματίζει με την καταστροφή των πινάκων *T1* και *T2*, μέσω της συνόδου συστήματος και την επικύρωση της δοσοληψίας της συνόδου συστήματος.

   iii) αν τα αποτελέσματα των επεκταμένων ερωτήσεων συνδυάζονται μέσω μιας από τις πράξεις *EXCEPT, PUNION* και *PEXCEPT* της ΣΑΧΕ, τότε καλείται η κατάλληλη διαδικασία βιβλιοθήκης της ΣΑΧΕ, η οποία επεξεργάζεται τους πίνακες *T1* και *T2* και αποθηκεύει το αποτέλεσμα σε έναν νέο προσωρινό πίνακα, τον οποίο συμβολίζουμε με *T3*. Η διαδικασία βιβλιοθήκης αλληλεπιδρά με το σχεσιακό ΣΔΒΔ αποκλειστικά μέσω της συνόδου συστήματος. Στη συνέχεια, οι πίνακες *T1* και *T2* καταστρέφονται, μέσω της συνόδου συστήματος και η δοσοληψία της συνόδου συστήματος επικυρώνεται. Τέλος, οι πλειάδες του πίνακα *T3* ανακτώνται (ενδεχομένως ταξινομημένες, αν η εντολή *SELECT* που έδωσε ο χρήστης περιείχε

την πρόταση *ORDER BY*) και παρουσιάζονται στον χρήστη, ο πίνακας *T3* καταστρέφεται, μέσω της συνόδου συστήματος και η δοσοληψία της συνόδου συστήματος επικυρώνεται.

Τα σχόλια που παρατίθενται στην περίπτωση (2) για την απουσία αδιεξόδων, τη μη εισαγωγή σημείων αυτόματης επικύρωσης για τη σύνοδο χρήστη, την αύξηση του μεγέθους του ημερολογίου των δύο συνόδων και την τοποθέτηση κλειδωμάτων, ισχύουν και γι' αυτή την περίπτωση.

5.1.2. Εκτέλεση της εντολής INSERT.

Για την εισαγωγή δεδομένων σε πίνακες διακρίνονται οι ακόλουθες περιπτώσεις (στα επόμενα ο πίνακας στον οποίο θα εισαχθούν τα δεδομένα θα συμβολίζεται με *Π*. Το σχήμα του πίνακα *Π* θα συμβολίζεται με $Π_1$, ..., $Π_ν$, $Π_{χε}$, όπου $Π_{χε}$ είναι η στήλη κανονικοποίησης του πίνακα (αν ο πίνακας δεν είναι κανονικοποιημένος, τότε η στήλη $Π_{χε}$ δεν θεωρείται μέρος του σχήματος του πίνακα). Οι στήλες του πίνακα $Π_1$ έως $Π_ν$ χωρίζονται σε δύο ξένα μεταξύ τους σύνολα, τα $Π_{κλειδί}$ και $Π_{-κλειδί}$, το πρώτο από τα οποία περιέχει τις στήλες που ανήκουν στο πρωτεύον κλειδί, ενώ το δεύτερο αυτές που δεν μετέχουν. Αν δεν έχει οριστεί πρωτεύον κλειδί πάνω στον πίνακα *Π*, το σύνολο $Π_{κλειδί}$ θα είναι κενό, ενώ το σύνολο $Π_{-κλειδί}$ θα περιλαμβάνει όλες τις στήλες $Π_1$, ..., $Π_ν$):

1. *ο πίνακας Π δεν είναι κανονικοποιημένος και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω της πρότασης VALUES ή μέσω ερώτησης που δεν περιέχει τις προτάσεις REFORMAT AS και NORMALISE*. Σ' αυτή την περίπτωση, ο χρονολογικός φλοιός ανασυνθέτει την εντολή *INSERT*, στη μορφή που δόθηκε από τον χρήστη και την προωθεί στο σχεσιακό ΣΔΒΔ, μέσω της συνόδου χρήστη. Το σχεσιακό ΣΔΒΔ θα εκτελέσει την εντολή, τοποθετώντας τα απαραίτητα κλειδώματα και καταχωρίζοντας στο ημερολόγιο της συνόδου χρήστη τα δεδομένα που είναι απαραίτητα για την αναίρεση των αλλαγών που επέφερε η εκτέλεση της εντολής, αν κάτι τέτοιο ζητηθεί ή είναι απαραίτητο, κατά τη διαδικασία επανόρθωσης.
2. *ο πίνακας Π δεν είναι κανονικοποιημένος και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω ερώτησης που περιέχει την πρόταση REFORMAT AS ή/και την πρόταση NORMALISE*. Σ' αυτή την περίπτωση αρχικά αποτιμάται η ερώτηση, ακολουθώντας τα βήματα (i) έως (iii) της περίπτωσης (2) του αλγορίθμου που περιγράφεται για την εντολή *SELECT* και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο συμβολίζουμε με *P1*. Ακολούθως, σχηματίζεται η εντολή

INSERT INTO *Π* SELECT * FROM *P1*

η οποία προωθείται στο σχεσιακό ΣΔΒΔ προς εκτέλεση, μέσω της συνόδου χρήστη. Ο αλγόριθμος τερματίζει με την καταστροφή του προσωρινού πίνακα *P1*, μέσω της συνόδου συστήματος και την επικύρωση της δοσοληψίας της συνόδου συστήματος.

Ένα σημείο που χρήζει προσοχής στην πιο πάνω διαδικασία, είναι ότι η σύνοδος χρήστη προσπελαύνει για ανάγνωση τον προσωρινό πίνακα *P1*, ο οποίος στη συνέχεια θα καταστραφεί μέσω της συνόδου συστήματος. Για να μην προκύψει αδιέξοδο, είναι απαραίτητο η προσπέλαση αυτή να μην συνεπάγεται την τοποθέτηση κλειδωμάτων στον πίνακα *P1* από τη σύνοδο χρήστη. Αυτό μπορεί να επιτευχθεί μέσω της εντολής

SET LOCKMODE ON TABLE *P1* WHERE READLOCK = NOLOCK

η οποία προωθείται στο σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη, *πριν* την προώθηση της εντολής *INSERT*. (Η εντολή *SET LOCKMODE* είναι μία επέκταση της SQL89 που υλοποιείται από το Ingres ([Ingres90]). Σε άλλα σχεσιακά ΣΔΒΔ, το συντακτικό είναι διαφορετικό.)

Ο αλγόριθμος εισαγωγής που περιγράφηκε γι' αυτή την περίπτωση, είναι απαλλαγμένος από αδιέξοδα, μια και κατά την αποτίμηση της ερώτησης οι πίνακες των χρηστών προσπελαύνονται μέσω της συνόδου χρήστη και οι προσωρινοί πίνακες μέσω της συνόδου συστήματος, ενώ κατά την εισαγωγή των πλειάδων στον πίνακα *Π*, όπου η σύνοδος χρήστη προσπελαύνει έναν προσωρινό πίνακα, η προσπέλαση σ' αυτόν γίνεται χωρίς να ελέγχεται η ύπαρξη κλειδωμάτων και χωρίς να τίθενται κλειδώματα. Επίσης, ο αλγόριθμος εισαγωγής δεν εισάγει σημεία αυτόματης επικύρωσης για τη σύνοδο χρήστη και αυξάνει το μέγεθος του ημερολογίου της μόνο στο βαθμό που είναι απαραίτητο για την αναίρεση των αλλαγών που επέφερε η εκτέλεση της εντολής, αν κάτι τέτοιο ζητηθεί ή είναι απαραίτητο, κατά τη διαδικασία επανόρθωσης. Το μέγεθος του ημερολογίου της συνόδου συστήματος αυξάνει μόνο προσωρινά (κατά τη διάρκεια αποτίμησης της ερώτησης) και η αύξηση είναι σχετικά μικρή (το μέγιστο μέγεθός του είναι αυτό που απαιτείται για να καταχωριστούν οι αλλαγές που επιφέρει στη βάση δεδομένων μία διαδικασία βιβλιοθήκης της ΣΑΧΕ). Τέλος, κατά τη διαδικασία αποτίμησης της ερώτησης τίθενται τα απαραίτητα κλειδώματα στα δεδομένα που μετέχουν στον υπολογισμό των πλειάδων που θα εισαχθούν στον πίνακα *Π* και η εντολή *INSERT* που προωθείται στο σχεσιακό ΣΔΒΔ, θέτει αποκλειστικά κλειδώματα στις εισαχθείσες πλειάδες, εξασφαλίζοντας ότι αυτές δεν είναι προσπελάσιμες από άλλες δοσοληψίες, ενόσω η δοσοληψία της συνόδου χρήστη είναι ενεργός.

3. *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί κλειδί πάνω σ' αυτόν και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω της πρότασης VALUES*. Σ' αυτή την περίπτωση, ο χρονολογικός φλοιός δημιουργεί μία *πλειάδα εισαγωγής*, την οποία θα συμβολίζουμε με *πε*. Το σχήμα της πλειάδας *πε* είναι ίδιο με αυτό του πίνακα *Π* και οι τιμές των στηλών της προκύπτουν από τις τιμές που παρατίθεται στην πρόταση *VALUES* (αν κάποια στήλη της πλειάδας εισαγωγής δεν λαμβάνει τιμή, τότε παίρνει τιμή *NULL*). Κατόπιν, ορίζεται μέσω της συνόδου χρήστη ένας δρομέας (τον οποίο συμβολίζουμε με *δρομέας_εισαγωγής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* για τις οποίες οι τιμές όλων των στηλών του συνόλου $Π_{-κλειδί}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $π.Π_{χε}$ *merges* $πε.Π_{χε}$ είναι αληθές. Για κάθε μία από τις επιλεγόμενες πλειάδες (την οποία θα συμβολίζουμε με *τρέχουσα_πλειάδα*), προσκομίζεται στη μνήμη του χρονολογικού φλοιού η στήλη $τρέχουσα\_πλειάδα.Π_{χε}$, η τιμή της στήλης $πε.Π_{χε}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης $merge(πε.Π_{χε}, τρέχουσα\_πλειάδα.Π_{χε})$ και η πλειάδα *τρέχουσα_πλειάδα* διαγράφεται από τον πίνακα *Π*, μέσω της συνόδου χρήστη. Όταν προσκομισθούν όλες οι πλειάδες του πίνακα *Π* που πληρούν τα κριτήρια επιλογής του δρομέα *δρομέας_εισαγωγής*, τότε η πλειάδα *πε* προστίθεται στον πίνακα *Π*, μέσω της συνόδου χρήστη.

Ο αλγόριθμος εισαγωγής δεδομένων για την περίπτωση αυτή αλληλεπιδρά με το σχεσιακό ΣΔΒΔ αποκλειστικά μέσω της συνόδου χρήστη, κατά συνέπεια είναι απαλλαγμένος από αδιέξοδα. Κατά την εκτέλεση του αλγορίθμου, το σχεσιακό ΣΔΒΔ θέτει τα απαραίτητα αποκλειστικά κλειδώματα, τόσο στις διαγραφόμενες πλειάδες, όσο και στην εισαγόμενη πλειάδα, οπότε οι πλειάδες αυτές δεν είναι προσπελάσιμες από άλλες συνόδους ενόσω η τρέχουσα δοσοληψία της συνόδου χρήστη είναι ενεργός. Τέλος, οι αλλαγές που επιφέρει ο αλγόριθμος στη βάση δεδομένων είναι ελάχιστες, στην περίπτωση που ο δρομέας *δρομέας_εισαγωγής* δεν επιλέξει καμία πλειάδα και σχεδόν ελάχιστες, αν μία τουλάχιστον πλειάδα επιλεχθεί από τον δρομέα *δρομέας_εισαγωγής*, κατά συνέπεια η αύξηση στο μέγεθος του ημερολογίου της συνόδου χρήστη να είναι χαμηλή.

Οι ελάχιστες αλλαγές στη βάση δεδομένων, όταν ο δρομέας *δρομέας_εισαγωγής* έχει επιλέξει τουλάχιστον μία πλειάδα, συνίστανται στη διαγραφή όλων των πλειάδων που μπορούν να συμπτυχθούν με την πλειάδα εισαγωγής, πλην μίας, της οποίας ο χρόνος εγκυρότητας θα τεθεί ίσος με την ένωση του χρόνου εγκυρότητας της πλειάδας εισαγωγής και των πλειάδων που επιλέγονται από τον δρομέα *δρομέας_εισαγωγής*. Η προσέγγιση αυτή δεν μπορεί να υιοθετηθεί, διότι:

i) η τελική τιμή του χρόνου εγκυρότητας που θα χρησιμοποιηθεί, δεν είναι γνωστή παρά μόνο αφού έχουν προσκομισθεί στη μνήμη όλες οι πλειάδες που επιλέγονται από τον δρομέα *δρομέας_εισαγωγής*. Έτσι, η μόνη λογική προσέγγιση είναι να διαγράψουμε όλες τις πλειάδες που επιλέγονται από τον δρομέα *δρομέας_εισαγωγής*, πλην της τελευταίας, της οποίας θα ενημερώσουμε τον χρόνο εγκυρότητας.

ii) οι δρομείς δεν παρέχουν καμία ένδειξη για το αν η τρέχουσα πλειάδα είναι η τελευταία που επιλέγεται ή όχι. Ο μόνος τρόπος να αποκτήσουμε αυτή την πληροφορία είναι να ζητήσουμε την προσκόμιση της επόμενης πλειάδας, μέσω του δρομέα και να εξετάσουμε αν η αίτηση αυτή διεκπεραιώνεται επιτυχώς ή δεν μπορεί να ικανοποιηθεί, λόγω εξάντλησης των δεδομένων. Ζητώντας όμως την επόμενη πλειάδα, χάνουμε τη δυνατότητα να αναφερθούμε στην τρέχουσα (διότι η αιτηθείσα πλειάδα θα είναι πλέον η τρέχουσα), με συνέπεια να μην μπορούμε να χρησιμοποιήσουμε τη σύνταξη *WHERE CURRENT OF δρομέας_εισαγωγής* στις εντολές *DELETE* και *UPDATE*, προκειμένου να προβούμε στην κατάλληλη ενέργεια.

Όμως, σε σχεσιακά ΣΔΒΔ που υποστηρίζουν την έννοια του *προσδιοριστή εγγραφής* (record identifier), όπως η Oracle ([Oracle90]), είναι δυνατό να εφαρμοστεί μία παραλλαγή του αλγορίθμου, η οποία περιορίζει τις αλλαγές στην κατάσταση της βάσης δεδομένων, με ελάχιστη επιβάρυνση στην απόδοση του αλγορίθμου. Σύμφωνα με την παραλλαγή αυτή, η πρώτη πλειάδα που θα επιλεχθεί από τον δρομέα *δρομέας_εισαγωγής* δεν διαγράφεται, αλλά ο προσδιοριστής εγγραφής που της αντιστοιχεί φυλάσσεται σε μία μεταβλητή (έστω *προσδιοριστής_εγγραφής*) και η πλειάδα κλειδώνεται με αποκλειστικό κλείδωμα (οι υπόλοιπες πλειάδες που θα επιλεχθούν διαγράφονται, σύμφωνα με τον ανωτέρω αλγόριθμο). Όταν εξεταστούν όλες οι πλειάδες που προσκομίζονται μέσω του δρομέα *δρομέας_εισαγωγής*, τότε η στήλη $\Pi_{\chi\varepsilon}$ της πρώτης πλειάδας που επιλέχθηκε, τίθεται ίση με τη στήλη $\Pi_{\chi\varepsilon}$ της πλειάδας εισαγωγής, μέσω μιας εντολής

UPDATE *Π* SET $\Pi_{\chi\varepsilon}$ = :πλειάδα_εισαγωγής.$\Pi_{\chi\varepsilon}$

WHERE προσδιοριστής_εγγραφής = :προσδιοριστής_εγγραφής

η οποία προωθείται στο σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη. Δεδομένου ότι ο προσδιοριστής εγγραφής απεικονίζει τη θέση της πλειάδας στον δίσκο, η εκτέλεση της εντολής αυτή δεν απαιτεί τη σάρωση του πίνακα, αλλά εντοπίζει άμεσα την πλειάδα και την ενημερώνει.

Σημειώνουμε, τέλος, ότι αν ο δρομέας *δρομέας_εισαγωγής* δεν επιλέξει καμία πλειάδα, η *πλειάδα_εισαγωγής* προστίθεται κανονικά στον πίνακα *Π*, μέσω της συνόδου χρήστη.

4. *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί κλειδί πάνω σ' αυτόν και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω της πρότασης VALUES.* Σ' αυτή την περίπτωση, ο χρονολογικός φλοιός δημιουργεί μία *πλειάδα εισαγωγής*, όπως στην περίπτωση (3), την οποία θα συμβολίζουμε με *πε*. Κατόπιν, δημιουργείται ένα *σημείο επαναφοράς* (savepoint) για τη σύνοδο χρήστη, το οποίο θα συμβολίζουμε με *ΣΕ* και, μέσω της ίδιας συνόδου, ορίζεται ένας δρομέας (τον οποίο συμβολίζουμε με *δρομέας_εισαγωγής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* για τις οποίες οι τιμές όλων των στηλών του συνόλου $Π_{κλειδί}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $π.Π_{χε}$ *merges* $πε.Π_{χε}$ είναι αληθές. Για κάθε μία από τις επιλεγόμενες πλειάδες (την οποία θα συμβολίζουμε με *τρέχουσα_πλειάδα*), προσκομίζονται στη μνήμη του χρονολογικού φλοιού οι στήλες του συνόλου $Π_{¬κλειδί}$ και η στήλη $Π_{χε}$ και ακολουθούνται τα εξής βήματα:

   i) αν το κατηγόρημα $τρέχουσα\_πλειάδα.Π_{χε}$ *cp* $πε.Π_{χε}$ αληθεύει, τότε η εντολή *INSERT* παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Η εντολή

   ROLLBACK TO *ΣΕ*

   προωθείται στο σχεσιακό ΣΔΒΔ, μέσω της συνόδου χρήστη, με αποτέλεσμα την αναίρεση των αλλαγών που έχει επιφέρει η σύνοδος χρήστη στη βάση δεδομένων από τη στιγμή δημιουργίας του σημείου επαναφοράς *ΣΕ*, καθώς και την άρση των κλειδωμάτων που η έχει θέσει η σύνοδος χρήστη κατά το ίδιο χρονικό διάστημα. Τέλος, η διαδικασία εκτέλεσης της εντολής τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

   ii) αν όλες οι στήλες της τρέχουσας πλειάδας που ανήκουν στο σύνολο $Π_{¬κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *πε*, τότε η τιμή της στήλης $πε.Π_{χε}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης *merge*($πε.Π_{χε}$, $τρέχουσα\_πλειάδα.Π_{χε}$) και η πλειάδα *τρέχουσα_πλειάδα* διαγράφεται από τον πίνακα *Π*, μέσω της συνόδου χρήστη.

   iii) σε όλες τις άλλες περιπτώσεις, δηλαδή αν το κατηγόρημα $τρέχουσα\_πλειάδα.Π_{χε}$ *cp* $πε.Π_{χε}$ είναι ψευδές και οποιαδήποτε από τις στήλες της τρέχουσας πλειάδας, που ανήκουν στο σύνολο $Π_{¬κλειδί}$ έχει τιμή διαφορετική από την αντίστοιχη στήλη της πλειάδας *πε*, η *τρέχουσα_πλειάδα* αγνοείται.

Όταν προσκομισθούν όλες οι πλειάδες του πίνακα *Π* που πληρούν τα κριτήρια επιλογής του δρομέα *δρομέας_εισαγωγής*, τότε η πλειάδα *πε* προστίθεται στον πίνακα *Π*, μέσω της συνόδου χρήστη.

Τα σχόλια που παρατίθενται για την περίπτωση (3) σχετικά με την έλλειψη αδιεξόδων, την αύξηση μεγέθους του ημερολογίου της συνόδου χρήστη καθώς η παραλλαγή του αλγορίθμου, ισχύουν και για αυτή την περίπτωση. Το σχεσιακό ΣΔΒΔ θα τοποθετήσει αποκλειστικά κλειδώματα στις διαγραφόμενες και την εισαγόμενη πλειάδα, απαγορεύοντας την προσπέλασή τους από άλλες συνόδους, ενόσω η τρέχουσα σύνοδος χρήστη είναι ενεργός. Το σχεσιακό ΣΔΒΔ θα τοποθετήσει όμως και διαμοιραζόμενα κλειδώματα στις πλειάδες που τυχόν θα επιλεχθούν από τον δρομέα *δρομέα_εισαγωγής*, αλλά δεν μπορούν να συμπτυχθούν με την εισαγόμενη πλειάδα (οι οποίες δεν είναι απαραίτητο να κλειδωθούν), απαγορεύοντας έτσι την ενημέρωσή της από άλλες συνόδους, ενόσω η τρέχουσα σύνοδος χρήστη είναι ενεργός. Τα επιπρόσθετα αυτά κλειδώματα δεν αποτελούν όμως σημαντικό πρόβλημα για τους εξής λόγους:

i) δεδομένου ότι έχει ορισθεί πρωτεύον κλειδί στον πίνακα *Π*, ο αριθμός των πλειάδων που θα κλειδωθούν, χωρίς να είναι απαραίτητο, θα είναι κατά μέγιστον 2 (μία πλειάδα *π* για την οποία όλες οι στήλες της που ανήκουν στο σύνολο $\Pi_{κλειδί}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $π.\Pi_{χε}$ *meets* $πε.\Pi_{χε}$ και μία πλειάδα *π′* για την οποία όλες οι στήλες της που ανήκουν στο σύνολο $\Pi_{κλειδί}$ είναι ίσες με τις αντίστοιχες στήλες της πλειάδας *πε* και το κατηγόρημα $π'.\Pi_{χε}$ *met* $πε.\Pi_{χε}$ αληθεύει).

ii) οι πλειάδες *π* και *π′*, αν υπάρχουν, θα βρίσκονται κατά πάσα πιθανότητα στην ίδια σελίδα δίσκου με μία πλειάδα που πρέπει να κλειδωθεί, μια και για τις σχέσεις στις οποίες έχουν ορισθεί κλειδιά, χρησιμοποιείται συνήθως κάποιο σχήμα αποθήκευσης που διαχωρίζει τις πλειάδες ανάλογα με την τιμή του κλειδιού τους (π.χ. B-δένδρο ή ISAM, πάνω στις στήλες που μετέχουν στο κλειδί). Δεδομένου ότι τα σχεσιακά ΣΔΒΔ τοποθετούν συνήθως κλειδώματα σε επίπεδο σελίδας δίσκου, οι πλειάδες *π* και *π′* θα κλειδωνόταν ούτως ή άλλως με αποκλειστικό κλείδωμα.

5. *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω ερώτησης*. Σ' αυτή την περίπτωση αρχικά αποτιμάται η ερώτηση, ακολουθώντας τα βήματα (i) έως (iii) της περίπτωσης (2) του αλγορίθμου που περιγράφεται για την εντολή *SELECT* και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο συμβολίζουμε με *P1*. Ακολούθως,

ορίζεται ένας δρομέας μέσω της συνόδου χρήστη, ο οποίος επιλέγει τις πλειάδες του πίνακα *Π*, οι οποίες μπορούν να συμπτυχθούν με οποιαδήποτε πλειάδα του πίνακα *P1*. Οι επιλεγόμενες πλειάδες διαγράφονται από τον πίνακα *Π*, μέσω της συνόδου χρήστη και εισάγονται στον πίνακα *P1*, μέσω της συνόδου συστήματος. Δεδομένου ότι αυτή η διαδικασία απαιτεί προσπέλαση για ανάγνωση στον πίνακα *P1* μέσω της συνόδου χρήστη και ότι ο πίνακας αυτός θα καταστραφεί στη συνέχεια μέσω της συνόδου συστήματος, είναι σημαντικό η προσπέλαση αυτή να μην οδηγήσει στην τοποθέτηση κλειδωμάτων από τη σύνοδο χρήστη πάνω στον πίνακα *P1*. Αυτό μπορεί να επιτευχθεί μέσω της εντολής

SET LOCKMODE ON TABLE *P1* WHERE READLOCK = NOLOCK

η οποία προωθείται στο σχεσιακό ΣΔΒΔ *πριν* τον ορισμό του δρομέα. Τέλος, μέσω της συνόδου συστήματος, εκτελείται η πράξη *σύμπτυξη* πάνω στη στήλη $\Pi_{\chi\varepsilon}$ του πίνακα *P1*, τα αποτελέσματα της πράξης *σύμπτυξη* ανακτώνται μέσω της συνόδου συστήματος και εισάγονται στον πίνακα *Π*, μέσω της συνόδου χρήστη. Ο αλγόριθμος ολοκληρώνεται με την καταστροφή του προσωρινού πίνακα *P1*, μέσω της συνόδου συστήματος και την επικύρωση της τρέχουσας δοσοληψίας της συνόδου συστήματος.

Ο αλγόριθμος εισαγωγής που περιγράφηκε γι' αυτή την περίπτωση, είναι απαλλαγμένος από αδιέξοδα, μια και κατά την αποτίμηση της ερώτησης οι πίνακες των χρηστών προσπελαύνονται μέσω της συνόδου χρήστη και οι προσωρινοί πίνακες μέσω της συνόδου χρήστη, ενώ στα επόμενα βήματα, όπου η σύνοδος χρήστη προσπελαύνει τον προσωρινό *P1*, η προσπέλαση σ' αυτόν γίνεται χωρίς να ελέγχεται η ύπαρξη κλειδωμάτων και χωρίς να τίθενται κλειδώματα. Επίσης, ο αλγόριθμος εισαγωγής δεν εισάγει σημεία αυτόματης επικύρωσης για τη σύνοδο χρήστη και αυξάνει το μέγεθος του ημερολογίου της μόνο στο βαθμό που είναι απαραίτητο για την αναίρεση των αλλαγών που επέφερε η εκτέλεση της εντολής, αν κάτι τέτοιο ζητηθεί ή είναι απαραίτητο, κατά τη διαδικασία επανόρθωσης. Το μέγεθος του ημερολογίου της συνόδου συστήματος αυξάνει μόνο προσωρινά (κατά την κλήση διαδικασιών βιβλιοθήκης της ΣΑΧΕ και κατά την εισαγωγή πλειάδων στον πίνακα *P1*) και η αύξηση είναι σχετικά μικρή. Τέλος, κατά τη διαδικασία αποτίμησης της ερώτησης τίθενται τα απαραίτητα κλειδώματα στα δεδομένα που μετέχουν στον υπολογισμό των πλειάδων που θα εισαχθούν στον πίνακα *Π* και στα επόμενα βήματα τίθενται αποκλειστικά κλειδώματα στις πλειάδες που διαγράφονται από τον πίνακα *Π*, ή εισάγονται σ' αυτόν, εξασφαλίζοντας ότι αυτές δεν

είναι προσπελάσιμες από άλλες δοσοληψίες, ενόσω η δοσοληψία της συνόδου χρήστη είναι ενεργός.

6. *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί πρωτεύον κλειδί πάνω σ' αυτόν και τα προς εισαγωγή δεδομένα προσδιορίζονται μέσω ερώτησης*. Σ' αυτή την περίπτωση, ο χρονολογικός φλοιός δημιουργεί ένα *σημείο επαναφοράς* για τη σύνοδο χρήστη, το οποίο συμβολίζουμε με *ΣΕ*. Στη συνέχεια αποτιμάται η ερώτηση, ακολουθώντας τα βήματα (i) έως (iii) της περίπτωσης (2) του αλγορίθμου που περιγράφεται για την εντολή *SELECT* και το αποτέλεσμά της αποθηκεύεται σε έναν προσωρινό πίνακα, τον οποίο συμβολίζουμε με *P1*. Η εντολή

SET LOCKMODE ON TABLE *P1* WHERE READLOCK = NOLOCK

προωθείται προς το σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη, έτσι ώστε οι προσπελάσεις για ανάγνωση στον πίνακα *P1* μέσω της αυτής συνόδου να μην εξετάζουν την ύπαρξη κλειδωμάτων και να μην θέτουν κλειδώματα. Ο χρονολογικός φλοιός ορίζει, μέσω της συνόδου χρήστη, έναν δρομέα ο οποίος επιλέγει τις πλειάδες *(π, ρ)* του αποτελέσματος της σύνδεσης των πινάκων *Π* και *P1*, για τις οποίες όλες οι στήλες της πλειάδας *π* που ανήκουν στο σύνολο $\Pi_{κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *ρ* και το κατηγόρημα $\pi.\Pi_{χε}$ *merges* $\rho.\Pi_{χε}$ αληθεύει. Για κάθε επιλεγόμενη πλειάδα του αποτελέσματος της σύνδεσης, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της πλειάδας *π*, οι στήλες της πλειάδας *ρ* που ανήκουν στο σύνολο $\Pi_{\neg κλειδί}$ και η στήλη $\rho.\Pi_{χε}$ και ακολουθούνται τα εξής βήματα:

i) αν το κατηγόρημα $\pi.\Pi_{χε}$ *cp* $\rho.\Pi_{χε}$ αληθεύει, τότε η εντολή *INSERT* παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Η εντολή

   ROLLBACK TO *ΣΕ*

   προωθείται στο σχεσιακό ΣΔΒΔ, μέσω της συνόδου χρήστη, με αποτέλεσμα την αναίρεση όλων των αλλαγών που έχει επιφέρει η σύνοδος αυτή στη βάση δεδομένων από τη στιγμή που ορίστηκε το σημείο επαναφοράς *ΣΕ*, καθώς και την άρση των κλειδωμάτων που έχει θέσει η σύνοδος χρήστη κατά το ίδιο χρονικό διάστημα. Τέλος, ο πίνακας *P1* καταστρέφεται, μέσω της συνόδου συστήματος, η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται και ο αλγόριθμος τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους.

ii) αν όλες οι στήλες της πλειάδας *π* που ανήκουν στο σύνολο $\Pi_{\neg κλειδί}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας *ρ*, τότε η πλειάδα *π* διαγράφεται από τον

πίνακα *Π*, μέσω της συνόδου χρήστη και εισάγεται στον πίνακα *P1*, μέσω της συνόδου συστήματος.

iii) σε όλες τις άλλες περιπτώσεις, δηλαδή αν το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *cp* $\rho.\Pi_{\chi\varepsilon}$ είναι ψευδές και οποιαδήποτε στήλη της πλειάδας π που ανήκει στο σύνολο $\Pi_{\text{κλειδί}}$ έχει τιμή διαφορετική από την αντίστοιχη στήλη της πλειάδας ρ, η πλειάδα (π, ρ) αγνοείται.

Ακολούθως, μέσω της συνόδου συστήματος ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ελέγχου*), ο οποίος επιλέγει τις πλειάδες ρ του πίνακα *P1*, ταξινομημένες βάσει των στηλών που περιέχονται στο σύνολο $\Pi_{\text{κλειδί}}$ και της στήλης $\Pi_{\chi\varepsilon}$, με αυτή τη σειρά. Κατόπιν ακολουθούνται τα εξής βήματα:

i) για την πρώτη πλειάδα που ανακτάται μέσω του δρομέα *δρομέας_ελέγχου*, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της και αποθηκεύονται στη δομή *πλειάδα_ελέγχου*.

ii) για την επόμενη πλειάδα που ανακτάται μέσω του δρομέα *δρομέας_ελέγχου*, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της, αποθηκεύονται στη δομή *νέα_πλειάδα* και διενεργούνται οι ακόλουθοι έλεγχοι:

   a) αν όλες οι στήλες της δομής *νέα_πλειάδα* που περιέχονται στο σύνολο $\Pi_{\text{κλειδί}}$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της δομής *πλειάδα_ελέγχου* και το κατηγόρημα $\textit{πλειάδα\_ελέγχου}.\Pi_{\chi\varepsilon}$ *cp* $\textit{νέα\_πλειάδα}.\Pi_{\chi\varepsilon}$ αληθεύει, τότε η λειτουργία εισαγωγής παραβιάζει τη μοναδικότητα του πρωτεύοντος κλειδιού. Η εντολή

      ROLLBACK TO *ΣΕ*

      προωθείται στο σχεσιακό ΣΔΒΔ, μέσω της συνόδου χρήστη, ο πίνακας *P1* καταστρέφεται μέσω της συνόδου συστήματος, η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται και ο αλγόριθμος τερματίζει επιστρέφοντας έναν κωδικό λάθους.

   b) αν όλες οι στήλες της δομής *νέα_πλειάδα*, εκτός της στήλης $\textit{νέα\_πλειάδα}.\Pi_{\chi\varepsilon}$, έχουν τιμές ίσες με τις αντίστοιχες στήλες της δομής *πλειάδα_ελέγχου* και το κατηγόρημα $\textit{πλειάδα\_ελέγχου}.\Pi_{\chi\varepsilon}$ *adjacent* $\textit{νέα\_πλειάδα}.\Pi_{\chi\varepsilon}$ αληθεύει, τότε η στήλη $\textit{πλειάδα\_ελέγχου}.\Pi_{\chi\varepsilon}$ αντικαθίσταται από το αποτέλεσμα της συνάρτησης *merge*$(\textit{πλειάδα\_ελέγχου}.\Pi_{\chi\varepsilon}, \textit{νέα\_πλειάδα}.\Pi_{\chi\varepsilon})$ και το βήμα (ii) επαναλαμβάνεται.

   c) η πλειάδα *πλειάδα_ελέγχου* εισάγεται στον πίνακα *Π*, μέσω της συνόδου χρήστη, οι τιμές των στηλών της δομής *πλειάδα_ελέγχου* αντικαθίστανται από

τις αντίστοιχες τιμές της δομής *νέα_πλειάδα* και το βήμα (ii) επαναλαμβάνεται έως ότου εξετασθούν όλες οι πλειάδες του πίνακα *P1*.

Ο αλγόριθμος τερματίζει με την καταστροφή του προσωρινού πίνακα *P1*, μέσω της συνόδου συστήματος και την επικύρωση της τρέχουσας δοσοληψίας της συνόδου συστήματος.

Τα σχόλια που παρατίθενται για την περίπτωση (5) σχετικά με την έλλειψη αδιεξόδων, τη μη εισαγωγή σημείων αυτόματης επικύρωσης και την αύξηση μεγέθους του ημερολογίου της συνόδου χρήστη και της συνόδου συστήματος, ισχύουν και γι' αυτή την περίπτωση. Σε ό,τι αφορά την τοποθέτηση κλειδωμάτων, το σχεσιακό ΣΔΒΔ θα τοποθετήσει, διαμοιραζόμενα κλειδώματα στα δεδομένα που μετέχουν στον υπολογισμό των πλειάδων που θα εισαχθούν στον πίνακα *Π* και αποκλειστικά κλειδώματα στις πλειάδες του πίνακα *Π* που διαγράφονται, καθώς και στις εισαχθείσες πλειάδες. Για κάθε πλειάδα *ρ* του πίνακα *P1*, ενδέχεται να κλειδωθούν με διαμοιραζόμενο κλείδωμα δύο πλειάδες του πίνακα *Π* (υπό τις συνθήκες που περιγράφονται στην περίπτωση (4)), χωρίς όμως τα κλειδώματα αυτά να αποτελεί σημαντικό πρόβλημα (οι λόγοι είναι οι ίδιοι με την περίπτωση (4)).

Μερικά σχεσιακά ΣΔΒΔ (π.χ. το Ingres) επιτρέπουν την εκτέλεση εντολών ελέγχου κλειδωμάτων, όπως η εντολή *SET LOCKMODE*, μόνο όταν αυτές βρίσκονται στην αρχή κάποιας δοσοληψίας. Για να αποφευχθεί, σ' αυτές τις περιπτώσεις, η τοποθέτηση κλειδωμάτων στους προσωρινούς πίνακες από τη σύνοδο χρήστη, το σχήμα θέσης κλειδωμάτων τροποποιείται ως ακολούθως:

1. κατά τη δημιουργία της συνόδου χρήστη, ορίζεται ότι οι προσπελάσεις για ανάγνωση, μέσω αυτής της συνόδου, δεν θα θέτουν κλειδώματα και δεν θα ελέγχουν για ύπαρξη κλειδωμάτων. Αυτό επιτυγχάνεται μέσω των εντολών

```
SET LOCKMODE SESSION WHERE READLOCK = NOLOCK
COMMIT
```

οι οποίες προωθούνται στο σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη, αμέσως μετά την εγκαθίδρυσή της.

2. παρέχεται στον χρήστη του χρονολογικού φλοιού η δυνατότητα εκτέλεσης της εντολής *SET LOCKMODE*, στη μορφή που αυτή καθορίζει το σχήμα κλειδώματος *συγκεκριμένου πίνακα*, έτσι ώστε να είναι ο χρήστης σε θέση να καθορίζει ότι η προσπέλαση σε συγκεκριμένους πίνακες χρηστών θα γίνεται θέτοντας κλειδώματα και ελέγχοντας για την ύπαρξη κλειδωμάτων. Ο χρήστης δεν θα έχει τη δυνατότητα να εκτελέσει την

εντολής *SET LOCKMODE*, στη μορφή που αυτή καθορίζει τη συνολική συμπεριφορά της συνόδου, σε ό,τι αφορά τα κλειδώματα.

Η τροποποίηση αυτή δεν οδηγεί σε περιορισμό των δυνατοτήτων κλειδώματος που παρέχονται στον χρήστη, αλλά σε αλλαγή της εξ ορισμού συμπεριφοράς του ΣΔΒΔ, σε ό,τι αφορά τα κλειδώματα. Σημειώνουμε ότι το νέο σχήμα κλειδωμάτων αποτελεί το εξ ορισμού σχήμα κλειδωμάτων σε εμπορικά σχεσιακά ΣΔΒΔ (π.χ. Oracle).

5.1.3. Εκτέλεση της εντολής DELETE.

Όταν η εντολή *DELETE* δεν περιέχει την πρόταση *PORTION*, τότε προωθείται στο σχεσιακό ΣΔΒΔ, μέσω της συνόδου χρήστη. Το σχεσιακό ΣΔΒΔ θα φροντίσει για την τοποθέτηση κλειδωμάτων και την καταχώριση στο ημερολόγιο των δεδομένων που είναι απαραίτητα για την αναίρεση των αλλαγών που επιφέρει η εντολή. Αν η εντολή *DELETE* περιέχει την πρόταση *PORTION*, τότε ελέγχεται αν είναι έγκυρη, δηλαδή αν ο πίνακας *Π* από τον οποίο διαγράφονται τα δεδομένα είναι κανονικοποιημένος (το σχήμα του πίνακα *Π* θα συμβολίζεται με Π_1, ..., Π_v, $\Pi_{\chi\varepsilon}$, όπου $\Pi_{\chi\varepsilon}$ είναι η στήλη κανονικοποίησης του πίνακα) και η στήλη που αναφέρεται στην πρόταση *PORTION* είναι στήλη $\Pi_{\chi\varepsilon}$. Σε περίπτωση αποτυχίας του ελέγχου, ο αλγόριθμος τερματίζει επιστρέφοντας έναν κατάλληλο κωδικό σφάλματος. Αν ο έλεγχος είναι επιτυχής, τότε μέσω της συνόδου χρήστη, ορίζεται ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_διαγραφής*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π*, οι οποίες ικανοποιούν τη συνθήκη της πρότασης *WHERE* της εντολής *DELETE* (αν αυτή η πρόταση παρατίθεται) και για τις οποίες το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *cp έκφραση* αληθεύει (*έκφραση* είναι το δεξί μέλος της εκχώρησης *στήλη* = *έκφραση* της πρότασης *PORTION*). Για κάθε επιλεγόμενη πλειάδα προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλα τα πεδία της, καθώς και η τιμή της έκφρασης *έκφραση* και διενεργούνται οι ακόλουθοι έλεγχοι:

1. αν το κατηγόρημα *έκφραση supinterv* $\pi.\Pi_{\chi\varepsilon}$ αληθεύει, τότε η πλειάδα *π* διαγράφεται από τον πίνακα *Π*, μέσω της συνόδου χρήστη.
2. αν τα χρονικά σημεία που περιέχονται στη στήλη $\pi.\Pi_{\chi\varepsilon}$ και δεν περιέχονται στην έκφραση *έκφραση* είναι συνεχόμενα, και άρα μπορούν να αναπαρασταθούν με ένα μόνο χρονικό διάστημα έστω δ_1, τότε η τιμή της στήλης $\pi.\Pi_{\chi\varepsilon}$ της πλειάδας *π* τίθεται ίση με δ_1, χρησιμοποιώντας την εντολή

UPDATE *Π* SET $\Pi_{\chi\varepsilon} = \delta_1$ WHERE CURRENT OF *δρομέας_διαγραφής*

η οποία προωθείται στο σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη.

3. αν τα χρονικά σημεία που περιέχονται στη στήλη $π.Π_{χε}$ και δεν περιέχονται στην έκφραση *έκφραση* δεν είναι συνεχόμενα, με αποτέλεσμα να απαιτούνται *δύο* χρονικά διαστήματα για την αναπαράστασή τους, έστω $δ_1$ και $δ_2$, τότε τιμή της στήλης $Π_{χε}$ της πλειάδας $π$ τίθεται ίση με $δ_1$, μέσω της συνόδου χρήστη όπως στην περίπτωση (2) και η ίδια σύνοδος χρησιμοποιείται για να εισαχθεί στον πίνακα $Π$ μία νέα πλειάδα, της οποίας οι στήλες $Π_1$, ..., $Π_v$ έχουν τιμές ίσες με τις αντίστοιχες στήλες της πλειάδας $π$, ενώ η στήλη $Π_{χε}$ έχει τιμή ίση με $δ_2$.

Στον ανωτέρω αλγόριθμο, όλη η αλληλεπίδραση μεταξύ του χρονολογικού φλοιού και του σχεσιακού ΣΔΒΔ γίνεται μέσω της συνόδου χρήστη, κατά συνέπεια ο αλγόριθμος είναι απαλλαγμένος από αδιέξοδα. Δεδομένου ότι δεν εκτελούνται εντολές της γλώσσας ορισμού δεδομένων, δεν εισάγονται σημεία αυτόματης επικύρωσης για τη σύνοδο χρήστη. Τέλος, οι αλλαγές στην κατάσταση της βάσης δεδομένων που επιφέρει ο αλγόριθμος είναι οι ελάχιστες δυνατές, με αποτέλεσμα να αυξάνεται το μέγεθος του ημερολογίου της συνόδου χρήστη μόνο στο βαθμό που είναι απόλυτα απαραίτητο, ενώ το σχεσιακό ΣΔΒΔ θα τοποθετήσει αποκλειστικά κλειδώματα στις πλειάδες που επηρεάζονται από τον αλγόριθμο διαγραφής (διαγράφονται, ενημερώνονται ή εισάγονται), εξασφαλίζοντας ότι δεν θα είναι προσπελάσιμες από άλλες συνόδους, ενόσω η τρέχουσα δοσοληψία της συνόδου χρήστη είναι ενεργός. (Αν η πρόταση *WHERE* της εντολής *DELETE* περιέχει υποερωτήσεις που αναφέρουν και άλλους πίνακες, τότε το σχεσιακό ΣΔΒΔ θα τοποθετήσει τα απαραίτητα διαμοιραζόμενα κλειδώματα στους πίνακες αυτούς.)

5.1.4. Εκτέλεση της εντολής UPDATE.

Αν η εντολή *UPDATE* αφορά έναν μη κανονικοποιημένο πίνακα, τότε απλά πιστοποιείται ότι δεν περιέχει την πρόταση *PORTION* και, αν ο έλεγχος είναι επιτυχής, προωθείται στο σχεσιακό ΣΔΒΔ μέσω της συνόδου χρήστη. Το ΣΔΒΔ θα φροντίσει για την εκτέλεσή της, την τοποθέτηση κλειδωμάτων και την καταχώριση των απαραίτητων δεδομένων στο ημερολόγιο της συνόδου χρήστη. Αν ο πίνακας που υπόκειται σε ενημέρωση είναι κανονικοποιημένος, τότε διακρίνονται οι ακόλουθες περιπτώσεις (στα επόμενα ο πίνακας που υπόκειται σε ενημέρωση θα συμβολίζεται με $Π$. Το σχήμα του πίνακα $Π$ θα συμβολίζεται με $Π_1$, ..., $Π_v$, $Π_{χε}$, όπου $Π_{χε}$ είναι η στήλη κανονικοποίησης του πίνακα. Οι στήλες του πίνακα $Π_1$ έως $Π_v$ χωρίζονται σε δύο ξένα μεταξύ τους σύνολα, τα $Π_{κλειδί}$ και $Π_{\neg κλειδί}$, το πρώτο από τα οποία περιέχει τις στήλες που ανήκουν στο πρωτεύον κλειδί, ενώ το

δεύτερο αυτές που δεν μετέχουν. Αν δεν έχει οριστεί πρωτεύον κλειδί πάνω στον πίνακα *Π*, το σύνολο $Π_{κλειδί}$ θα είναι κενό, ενώ το σύνολο $Π_{-κλειδί}$ θα περιλαμβάνει όλες τις στήλες $Π_1$, ..., $Π_v$):

1. *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί κλειδί πάνω σ' αυτόν και δεν χρησιμοποιείται η πρόταση PORTION*. Σ' αυτή την περίπτωση, δημιουργείται μέσω της συνόδου συστήματος ένας πίνακας *P1*, του οποίου το σχήμα είναι ίδιο με το σχήμα του πίνακα *Π* και ορίζεται μέσω της συνόδου χρήστη ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ενημέρωσης*), ο οποίος επιλέγει τις πλειάδες του πίνακα *Π* που ικανοποιούν τη συνθήκη της πρότασης *WHERE* της εντολής *UPDATE*. Για κάθε επιλεγόμενη πλειάδα, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλες οι στήλες της. Αν κάποια στήλη αναφέρεται στο αριστερό μέλος μιας εκχώρησης *στήλη = έκφραση* της πρότασης *SET* της εντολής *UPDATE*, τότε αντί της πραγματικής τιμής της προσκομίζεται η τιμή της έκφρασης *έκφραση* της ίδιας εκχώρησης. Η επιλεχθείσα πλειάδα διαγράφεται από τον πίνακα *Π* μέσω της συνόδου χρήστη και εισάγεται στον πίνακα *P1*, μέσω της συνόδου συστήματος. Όταν τα δεδομένα που μπορούν να προσκομισθούν μέσω του δρομέα *δρομέας_ενημέρωσης* εξαντληθούν, η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται και οι πλειάδες που περιέχονται στον πίνακα *P1* εισάγονται στον πίνακα *Π*, ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (5) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

Ο αλγόριθμος αυτός είναι απαλλαγμένος από αδιέξοδα, μια και κατά τη δημιουργία του πίνακα *P1* οι πίνακες των χρηστών προσπελάζονται μέσω της συνόδου χρήστη και οι προσωρινοί πίνακες μέσω της συνόδου συστήματος, ενώ στη συνέχεια χρησιμοποιείται ένας απαλλαγμένος από αδιέξοδα αλγόριθμος για την εισαγωγή των πλειάδων του πίνακα *P1* στον πίνακα *Π*. Η αύξηση στο μέγεθος του ημερολογίου της συνόδου χρήστη είναι μικρή, καθώς κάθε ενημέρωση πλειάδας απεικονίζεται σε μία διαγραφή και μία εισαγωγή πλειάδας, μέσω της συνόδου χρήστη (στην πραγματικότητα, μία ενημέρωση πλειάδας μπορεί να προκαλέσει περισσότερες από μία διαγραφές, αν η ενημερωμένη πλειάδα μπορεί να συμπτυχθεί με κάποια πλειάδα του πίνακα *Π* που δεν ενημερώνεται). Το μέγεθος του ημερολογίου της συνόδου συστήματος αυξάνει προσωρινά μόνο (κατά την εισαγωγή των πλειάδων στον πίνακα *P1* και κατά την εκτέλεση του αλγορίθμου εισαγωγής). Τέλος, δεν εισάγονται σημεία αυτόματης επικύρωσης για τη σύνοδο χρήστη και το σχεσιακό ΣΔΒΔ θέτει τα απαραίτητα αποκλειστικά κλειδώματα στις πλειάδες του

πίνακα *Π* που επηρεάζονται από την ενημέρωση (οι πλειάδες που διαγράφονται καθώς και αυτές που εισάγονται) εξασφαλίζοντας έτσι ότι οι πλειάδες αυτές δεν θα είναι προσπελάσιμες από άλλες συνόδους, ενόσω η τρέχουσα σύνοδος χρήστη είναι ενεργός. (Αν η πρόταση *WHERE* της εντολής *UPDATE* περιέχει υποερωτήσεις που αναφέρουν και άλλους πίνακες, τότε το σχεσιακό ΣΔΒΔ θα τοποθετήσει τα απαραίτητα διαμοιραζόμενα κλειδώματα στους πίνακες αυτούς.)

2. *ο πίνακας Π είναι κανονικοποιημένος, δεν έχει ορισθεί κλειδί πάνω σ' αυτόν και η πρόταση PORTION χρησιμοποιείται.* Σ' αυτή την περίπτωση αρχικά ελέγχεται ότι στην πρόταση *PORTION* υπάρχει μόνο μία εκχώρηση της μορφής *στήλη = έκφραση* και ότι η *στήλη* που αναφέρεται στην εκχώρηση αυτή είναι η στήλη $\Pi_{\chi\varepsilon}$. Αν οποιοσδήποτε από τους δύο ελέγχους αποτύχει, τότε η διαδικασία εκτέλεσης της εντολής *UPDATE* τερματίζει, επιστρέφοντας έναν κατάλληλο κωδικό λάθους. Στη συνέχεια, δημιουργείται μέσω της συνόδου συστήματος ένας προσωρινός πίνακας *P1*, του οποίου το σχήμα είναι ίδιο με το σχήμα του πίνακα *Π* και ορίζεται, μέσω της συνόδου χρήστη, ένας δρομέας (τον οποίο θα συμβολίζουμε με *δρομέας_ενημέρωσης*), ο οποίος επιλέγει τις πλειάδες *π* του πίνακα *Π* που ικανοποιούν τη συνθήκη της πρότασης *WHERE* της εντολής *UPDATE* και για τις οποίες το κατηγόρημα $\pi.\Pi_{\chi\varepsilon}$ *cp έκφραση* αληθεύει (*έκφραση* είναι το δεξί μέλος της μοναδικής εκχώρησης που περιέχεται στην πρόταση *PORTION*). Για κάθε επιλεγόμενη πλειάδα, προσκομίζονται στη μνήμη του χρονολογικού φλοιού όλα τα πεδία της, οι τιμές όλων των εκφράσεων που εμφανίζονται ως δεξιά μέλη στις εκχωρήσεις της πρότασης *SET*, καθώς και η τιμή της έκφρασης που εμφανίζεται ως δεξί μέλος στην εκχώρηση της πρότασης *PORTION* και ακολουθούνται τα κάτωθι βήματα:

   i) διενεργούνται οι έλεγχοι (1) έως (3) που περιγράφονται στον αλγόριθμο διαγραφής, προκειμένου να απομακρυνθεί από τον πίνακα *Π* η πληροφορία που αφορά τον χρόνο εγκυρότητας που προσδιορίζεται από την πρόταση *PORTION*.

   ii) η σύνοδος συστήματος χρησιμοποιείται για να εισαχθεί στον πίνακα *P1* μία πλειάδα, οι τιμές των στηλών της οποίας καθορίζονται σύμφωνα με τα ακόλουθα κριτήρια:

      a) αν το όνομα της στήλης εμφανίζεται στο αριστερό μέλος μιας εκχώρησης *στήλη = έκφραση* της πρότασης *SET*, τότε η τιμή της συγκεκριμένης στήλης στη νέα πλειάδα θα είναι ίση με την τιμή της έκφρασης *έκφραση*, του δεξιού δηλαδή μέλους της εκχώρησης.

b) για τις υπόλοιπες στήλες, χρησιμοποιείται η τιμή της αντίστοιχης στήλης στην πλειάδα *π*, εκτός από τη στήλη $\Pi_{\chi\varepsilon}$, της οποίας η τιμή τίθεται ίση με την τιμή του δεξιού μέλους της εκχώρησης *στήλη = έκφραση* της πρότασης *PORTION*.

Όταν τα δεδομένα που μπορούν να προσκομισθούν μέσω του δρομέα *δρομέας_ενημέρωσης* εξαντληθούν, τότε η τρέχουσα δοσοληψία της συνόδου συστήματος επικυρώνεται και οι πλειάδες που περιέχονται στον πίνακα *P1* εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (5) της διαδικασίας εκτέλεσης της εντολής *INSERT* (το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπεται).

Τα σχόλια που παρατίθενται για την περίπτωση (1), σχετικά με την απουσία αδιεξόδων, τη μη εισαγωγή σημείων αυτόματης επικύρωσης για τη σύνοδο χρήστη, την αύξηση του μεγέθους του ημερολογίου της συνόδου συστήματος και την τοποθέτηση κλειδωμάτων ισχύουν και γι' αυτή την περίπτωση. Σε ό,τι αφορά την αύξηση του μεγέθους του ημερολογίου της συνόδου χρήστη, αυτή μένει σε χαμηλά επίπεδα, δεδομένου ότι μία ενημέρωση τμήματος πλειάδας απεικονίζεται είτε σε μία διαγραφή και μία εισαγωγή πλειάδας, είτε σε μία ενημέρωση και μία εισαγωγή πλειάδας, είτε σε μία ενημέρωση και δύο εισαγωγές πλειάδας, ανάλογα με το τμήμα της πλειάδας που ενημερώνεται.

3. *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί κλειδί πάνω σ' αυτόν και η πρόταση PORTION δεν χρησιμοποιείται*. Στην περίπτωση αυτή, ορίζεται ένα σημείο επαναφοράς για τη σύνοδο χρήστη (το οποίο συμβολίζουμε με *ΣΕ*) και εφαρμόζεται ο αλγόριθμος που περιγράφηκε για την περίπτωση (1), τροποποιημένος ώστε οι πλειάδες του πίνακα *P1* να εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (6) της διαδικασίας εκτέλεσης της εντολής *INSERT* (η δημιουργία του σημείου επαναφοράς, καθώς και το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπονται).

Τα σχόλια που παρατίθενται για την περίπτωση (1), σχετικά με την απουσία αδιεξόδων, τη μη εισαγωγή σημείων αυτόματης επικύρωσης για τη σύνοδο χρήστη, την τοποθέτηση κλειδωμάτων και την αύξηση του μεγέθους του ημερολογίου της συνόδου συστήματος και της συνόδου χρήστη ισχύουν και γι' αυτή την περίπτωση. Ειδικά για την τοποθέτηση κλειδωμάτων, πρέπει να ληφθούν υπόψη οι παρατηρήσεις που παρατίθενται για την περίπτωση (6) του αλγορίθμου εκτέλεσης της εντολής *INSERT*.

4. *ο πίνακας Π είναι κανονικοποιημένος, έχει ορισθεί κλειδί πάνω σ' αυτόν και η πρόταση PORTION χρησιμοποιείται*. Στην περίπτωση αυτή, ορίζεται ένα σημείο επαναφοράς για

τη σύνοδο χρήστη (το οποίο συμβολίζουμε με *ΣΕ*) και εφαρμόζεται ο αλγόριθμος που περιγράφηκε για την περίπτωση (2), τροποποιημένος ώστε οι πλειάδες του πίνακα *P1* να εισάγονται στον πίνακα *Π* ακολουθώντας τον αλγόριθμο που περιγράφεται για την περίπτωση (6) της διαδικασίας εκτέλεσης της εντολής *INSERT* (η δημιουργία του σημείου επαναφοράς, καθώς και το τμήμα του αλγορίθμου που δημιουργεί τον πίνακα *P1* παραλείπονται).

Τα σχόλια που παρατίθενται για την περίπτωση (2), σχετικά με την απουσία αδιεξόδων, τη μη εισαγωγή σημείων αυτόματης επικύρωσης για τη σύνοδο χρήστη, την τοποθέτηση κλειδωμάτων και την αύξηση του μεγέθους του ημερολογίου της συνόδου συστήματος και της συνόδου χρήστη ισχύουν και γι' αυτή την περίπτωση. Ειδικά για την τοποθέτηση κλειδωμάτων, πρέπει να ληφθούν υπόψη οι παρατηρήσεις που παρατίθενται για την περίπτωση (6) του αλγορίθμου εκτέλεσης της εντολής *INSERT*.

5.2. Απόδοση των αλγορίθμων υποστήριξης δοσοληψιών.

Στην ενότητα 5.1 παρουσιάστηκαν αλγόριθμοι, μέσω των οποίων είναι δυνατή η υποστήριξη δοσοληψιών σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ. Στις επόμενες παραγράφους συγκρίνεται η απόδοση αυτών των αλγορίθμων, με τους αλγόριθμους εκτέλεσης των εντολών της SQL-XE που παρουσιάστηκαν στο κεφάλαιο 4. Η σύγκριση γίνεται βάσει του μαθηματικού μοντέλου που περιγράφηκε στην ενότητα 4.4.2, ενώ για τις εντολές *INSERT*, *DELETE* και *UPDATE* παρατίθενται και ενδεικτικά διαγράμματα απόδοσης των αλγορίθμων που εμπεριέχουν υποστήριξη δοσοληψιών.

5.2.1. Η εντολή *SELECT*.

Για την εκτέλεση της εντολής *SELECT* διακρίνονται τρεις περιπτώσεις (όπως περιγράφεται στην παράγραφο 5.1.1):

1. *η εντολή SELECT (στη μορφή που έχει μετασχηματιστεί από τον συντακτικό αναλυτή) δεν απαιτεί την εφαρμογή πράξεων που δεν υποστηρίζονται από το σχεσιακό ΣΔΒΔ (REFORMAT, PUNION κ.τ.λ.)*. Στην περίπτωση αυτή ο αλγόριθμος εκτέλεσης της εντολής *SELECT* δεν διαφέρει από τον αλγόριθμο που περιγράφεται στην παράγραφο 4.3.7 και έτσι η υποστήριξη δοσοληψιών δεν έχει καμία επίπτωση στην απόδοση του χρονολογικού ΣΔΒΔ.
2. *η εντολή SELECT αποτελείται από μία μόνο επεκταμένη ερώτηση*. Στην περίπτωση αυτή, προκειμένου να υποστηρίζονται δοσοληψίες και ταυτόχρονοι χρήστες, οι πλειάδες που προκύπτουν από το τμήμα της ερώτησης που είναι σύμφωνο με τις προδιαγραφές του

προτύπου SQL89, προσκομίζονται πρώτα στη μνήμη του χρονολογικού φλοιού, μέσω της συνόδου χρήστη και στη συνέχεια εισάγονται σε έναν προσωρινό πίνακα, μέσω της συνόδου συστήματος. Στον αλγόριθμο που περιγράφεται στην ενότητα 4.3.7, η εισαγωγή των πλειάδων στον προσωρινό πίνακα γίνεται απ' ευθείας από το σχεσιακό ΣΔΒΔ, οπότε η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών έχει επίπτωση στην απόδοση του χρονολογικού ΣΔΒΔ. Η επιβάρυνση που προκύπτει είναι ίση με

$2 * N_{\alpha} * (\mu_{\alpha} * \chi_{byte} + \chi_{επιβ})$

όπου N_{α} είναι το πλήθος των πλειάδων που περιλαμβάνονται στο αποτέλεσμα του τμήματος της ερώτησης που είναι σύμφωνο με το πρότυπο SQL89 και μ_{α} είναι το μέγεθος της κάθε πλειάδας. Η επιβάρυνση αυτή είναι μικρή, σε σχέση με το συνολικό κόστος αποτίμησης της επεκταμένης ερώτησης, δεδομένου ότι η επεκταμένη ερώτηση περιέχει τουλάχιστον μία από τις προτάσεις *REFORMAT AS* και *NORMALISE*, οι οποίες απαιτούν ταξινόμηση του προσωρινού πίνακα και το κόστος της ταξινόμησης είναι κατά πολύ μεγαλύτερο του κόστους μετακίνησης των πλειάδων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό και αντίστροφα. Επιπροσθέτως, η επεκταμένη ερώτηση μπορεί να περιλαμβάνει συνδέσεις μεταξύ πινάκων, οι οποίες έχουν υψηλό κόστος, καθιστώντας την επερχόμενη επιβάρυνση αμελητέα.

3. *η εντολή SELECT περιλαμβάνει δύο επεκταμένες ερωτήσεις, των οποίων τα αποτελέσματα συνδυάζονται μέσω μιας πράξης UNION, UNION ALL, EXCEPT, PUNION ή PEXCEPT της ΣΑΧΕ.* Σε αναλογία με την περίπτωση (2), η επιβάρυνση που επιφέρει η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών είναι ίση με

$2 * (N_{\alpha} + N_{\beta}) * (\mu_{\alpha} * \chi_{byte} + \chi_{επιβ})$

όπου N_{α} και N_{β} είναι ο αριθμός πλειάδων που περιέχονται στο αποτέλεσμα της πρώτης και της δεύτερης ερώτησης, αντίστοιχα και μ_{α} είναι το μέγεθος της κάθε πλειάδας. Και σ' αυτή την περίπτωση, η επιβάρυνση είναι μικρή, σε σχέση με το συνολικό κόστος αποτίμησης της ερώτησης, δεδομένου ότι στη συνέχεια οι προσωρινοί πίνακες είτε θα ταξινομηθούν είτε θα λάβουν μέρος σε σύνδεση (ή και τα δύο). Οι λειτουργίες της ταξινόμησης και της σύνδεσης έχουν πολύ μεγαλύτερο κόστος από τη μεταφορά των πλειάδων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό και αντίστροφα, επομένως η προκύπτουσα επιβάρυνση μπορεί να θεωρηθεί αμελητέα.

5.2.2. Η εντολή *INSERT*.

Για τις περιπτώσεις (1), (3) και (4) που περιγράφονται στην παράγραφο 5.1.2, ο αλγόριθμος εκτέλεσης της εντολής *INSERT* δεν διαφοροποιείται από αυτόν που περιγράφεται στην ενότητα 4.3.7 (πέραν της εισαγωγής και χρήσης σημείων επαναφοράς, όπου αυτό είναι απαραίτητο), οπότε η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών δεν επιφέρει επιβάρυνση στην απόδοση του χρονολογικού ΣΔΒΔ. Στις περιπτώσεις (2), (5) και (6), ο αλγόριθμος της παραγράφου 5.1.2 παρουσιάζει τις εξής διαφορές από τον αλγόριθμο της παραγράφου 4.3.7:

1. οι πλειάδες του αποτελέσματος της ερώτησης, μεταφέρονται στη μνήμη του χρονολογικού φλοιού, μέσω της συνόδου χρήστη και στη συνέχεια την εισάγονται σε προσωρινό πίνακα, μέσω της συνόδου συστήματος. Στον αλγόριθμο της παραγράφου 4.3.7 η εισαγωγή των πλειάδων στον προσωρινό πίνακα γίνεται απ' ευθείας από το σχεσιακό ΣΔΒΔ.
2. ειδικά για τις περιπτώσεις (5) και (6), στην τελική φάση, οι πλειάδες του προσωρινού πίνακα προσκομίζονται στη μνήμη του χρονολογικού φλοιού, μέσω της συνόδου συστήματος και κατόπιν εισάγονται στον πίνακα, μέσω της συνόδου χρήστη. Στον αλγόριθμο της ενότητας 4.3.7 η εισαγωγή των πλειάδων στον προσωρινό πίνακα γίνεται απ' ευθείας από το σχεσιακό ΣΔΒΔ.

Η επιβάρυνση που προκύπτει από την υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών είναι ίση με

$4 * N_{\alpha} * (\mu_{\alpha} * \chi_{byte} + \chi_{επιβ})$

για τις περιπτώσεις (5) και (6) και ίση με

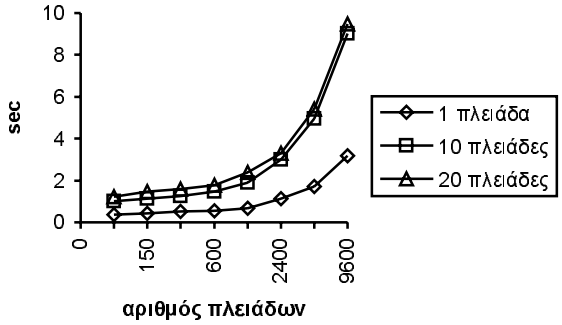
$2 * N_{\alpha} * (\mu_{\alpha} * \chi_{byte} + \chi_{επιβ})$

για την περίπτωση (2). (N_{α} είναι ο αριθμός πλειάδων που περιέχονται στο αποτέλεσμα της ερώτησης η οποία προσδιορίζει τα προς εισαγωγή δεδομένα και μ_{α} είναι το μέγεθος της κάθε πλειάδας.) Η επιβάρυνση αυτή είναι αμελητέα, δεδομένου ότι στη συνέχεια ο προσωρινός πίνακας θα ταξινομηθεί και (εκτός της περίπτωσης (2)) θα εκτελεστεί και μία σύνδεση μεταξύ του προσωρινού πίνακα και του πίνακα στον οποίο θα εισαχθούν τα δεδομένα.

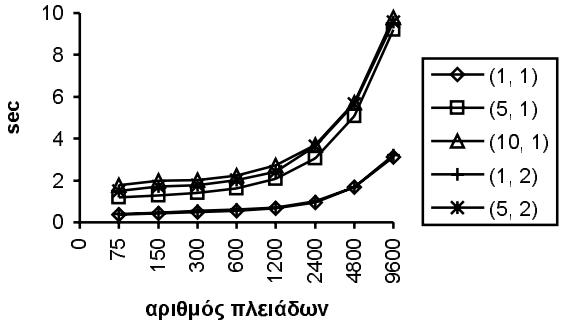
Στα σχήματα 5.1 έως 5.4 παρουσιάζονται ενδεικτικά διαγράμματα απόδοσης του αλγορίθμου εισαγωγής δεδομένων. Τα διαγράμματα 5.1 και 5.2 αφορούν εισαγωγή δεδομένων σε πίνακα χωρίς πρωτεύον κλειδί, ενώ τα δύο επόμενα αφορούν εισαγωγή δεδομένων σε πίνακα στον οποίο έχει ορισθεί πρωτεύον κλειδί. Στα διαγράμματα 5.1 και 5.3, οι εισαγόμενες πλειάδες

δεν μπορούν να συμπτυχθούν με καμία από τις πλειάδες που υπάρχουν στον πίνακα, ενώ στα διαγράμματα 5.2 και 5.4 κάθε μία από τις *ν* εισαγόμενες πλειάδες μπορεί να συμπτυχθεί με *μ* από τις πλειάδες που υπάρχουν στον πίνακα (συμβολίζεται με *(ν, μ)* στο υπόμνημα του διαγράμματος).

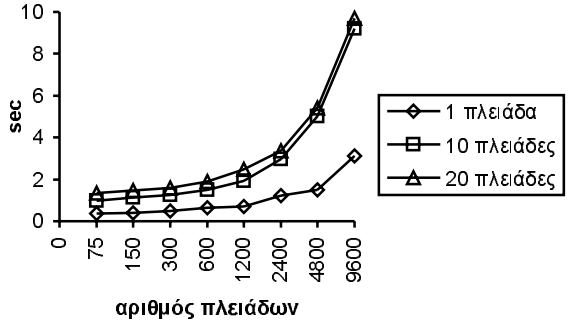
Στις περιπτώσεις όπου εισάγονται περισσότερες από μία πλειάδες (και κατά συνέπεια έχει χρησιμοποιηθεί ερώτηση για τον προσδιορισμό των προς εισαγωγή δεδομένων), μετρήθηκε ο συνολικός χρόνος εκτέλεσης της εντολής και στη συνέχεια αφαιρέθηκε ο χρόνος αποτίμησης της ερώτησης, μετρημένος σύμφωνα με τους αλγορίθμους που περιγράφονται στο κεφάλαιο 4. Η αναγωγή αυτή στον χρόνο εκτέλεσης έγινε για να είναι συγκρίσιμα τα διαγράμματα απόδοσης με αυτά της παραγράφου 4.4.3.



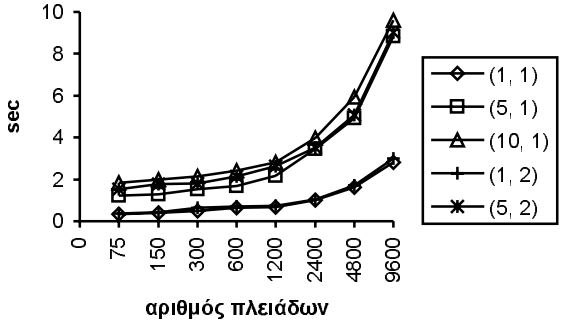
Σχήμα 5.1 - Απόδοση του αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.



Σχήμα 5.2 - Απόδοση του αλγόριθμου εισαγωγής δεδομένων σε πίνακα χωρίς κλειδί.



Σχήμα 5.3 - Απόδοση του αλγόριθμου εισαγωγής δεδομένων σε πίνακα με κλειδί.

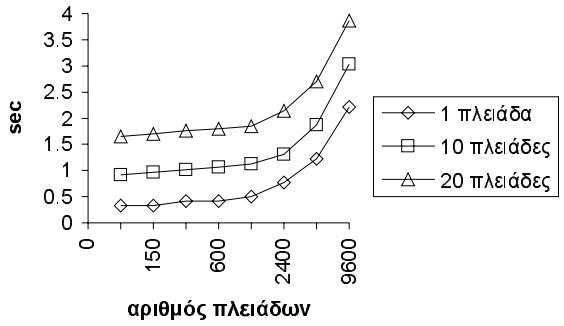


Σχήμα 5.4 - Απόδοση του αλγόριθμου εισαγωγής δεδομένων σε πίνακα με κλειδί.

Όπως προκύπτει από τη σύγκριση των ανωτέρω διαγραμμάτων με τα αντίστοιχα της παραγράφου 4.4.3, η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών δεν επιφέρει εμφανή επίπτωση στην απόδοση του αλγορίθμου εισαγωγής.

5.2.3. Η εντολή *DELETE*.

Ο αλγόριθμος εκτέλεσης της εντολής *DELETE* που περιγράφεται στην παράγραφο 5.1.3 δεν διαφοροποιείται, σε σχέση με αυτόν που περιγράφεται στην παράγραφο 4.3.7, οπότε η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών εισάγεται χωρίς να επιβαρύνεται η απόδοση του χρονολογικού ΣΔΒΔ. Στο διάγραμμα του σχήματος 5.5 παρουσιάζεται η απόδοση του αλγορίθμου διαγραφής σε ενδεικτικές περιπτώσεις. Η σύγκριση του διαγράμματος 5.5 με το αντίστοιχο διάγραμμα της παραγράφου 4.4.3 καταδεικνύει ότι η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών δεν επιφέρει εμφανή επίπτωση στην απόδοση του αλγορίθμου διαγραφής.



Σχήμα 5.5 - Απόδοση του αλγόριθμου διαγραφής δεδομένων.

5.2.4. Η εντολή *UPDATE*.

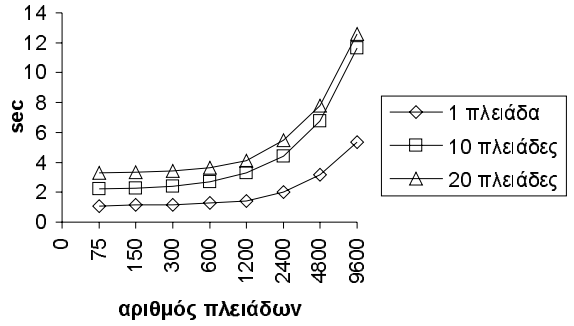
Ο αλγόριθμος εκτέλεσης της εντολής *UPDATE* που περιγράφεται στην παράγραφο 5.1.4 δεν διαφοροποιείται, σε σχέση με τον αλγόριθμο που περιγράφεται στην παράγραφο 4.3.7, παρά μόνο στο ότι κατά την τελική φάση εισαγωγής των ενημερωμένων πλειάδων στον πίνακα, αυτές προσκομίζονται στη μνήμη του χρονολογικού φλοιού, μέσω της συνόδου συστήματος και στη συνέχεια εισάγονται στον πίνακα, μέσω της συνόδου χρήστη (στον αλγόριθμο της παραγράφου 4.3.7, η εισαγωγή των πλειάδων γίνεται απ' ευθείας από το σχεσιακό ΣΔΒΔ, χωρίς τη μεσολάβηση του χρονολογικού φλοιού). Η προκύπτουσα επιβάρυνση είναι ίση με

$2 * N_{\alpha} * (\mu_{\alpha} * \chi_{byte} + \chi_{επιβ})$

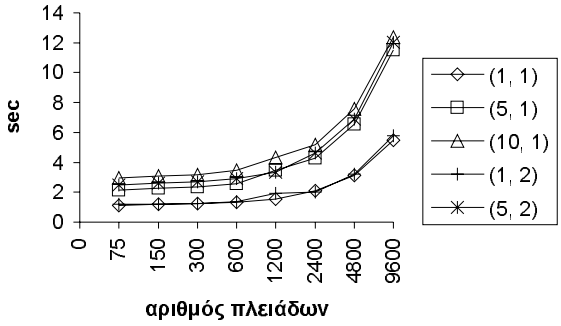
όπου N_{α} είναι το πλήθος των πλειάδων που ενημερώνονται και μ_{α} είναι το μέγεθος της κάθε πλειάδας. Η επιβάρυνση αυτή είναι αμελητέα, δεδομένου ότι η εκτέλεση της εντολής *UPDATE* περιλαμβάνει την ταξινόμηση του προσωρινού πίνακα που περιέχει τις ενημερωμένες πλειάδες και τη σύνδεσή του με τον υποκείμενο σε ενημέρωση πίνακα και το κόστος των πράξεων αυτών είναι κατά πολύ μεγαλύτερο του κόστους μετακίνησης των

πλειάδων από τον πυρήνα του σχεσιακού ΣΔΒΔ προς τον χρονολογικό φλοιό και αντίστροφα.

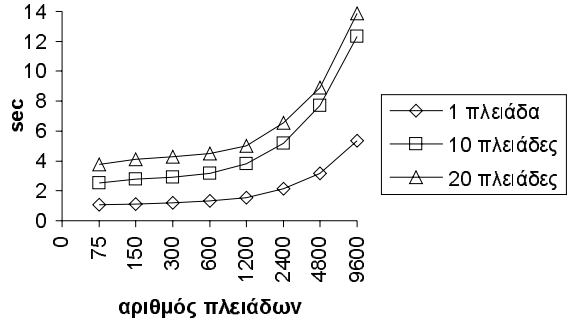
Στα σχήματα 5.6 έως 5.11, παρουσιάζονται ενδεικτικά διαγράμματα απόδοσης του αλγορίθμου ενημέρωσης. Τα διαγράμματα χωρίζονται σε δύο ομάδες, η πρώτη από τις οποίες (σχήματα 5.6 έως 5.8) αφορά ενημέρωση πινάκων στους οποίους δεν έχει ορισθεί πρωτεύον κλειδί, ενώ η δεύτερη (σχήματα 5.9 έως 5.11) αφορά ενημέρωση πινάκων στους οποίους έχει ορισθεί πρωτεύον κλειδί. Στο πρώτο διάγραμμα της κάθε ομάδας, δεν χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και οι ενημερωμένες πλειάδες δεν μπορούν να συμπτυχθούν με καμία από τις πλειάδες που δεν ενημερώνονται. Στο δεύτερο διάγραμμα της κάθε ομάδας, δεν χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και κάθε μία από τις *μ* πλειάδες που ενημερώνονται, μπορεί να συμπτυχθεί με *ν* από τις πλειάδες που δεν ενημερώνονται (συμβολίζεται με *(μ, ν)* στο υπόμνημα του διαγράμματος). Στο τελευταίο διάγραμμα της κάθε ομάδας χρησιμοποιείται η πρόταση *PORTION* στην εντολή *UPDATE* και οι ενημερωμένες πλειάδες δεν μπορούν να συμπτυχθούν με καμία από τις πλειάδες που δεν ενημερώνονται.



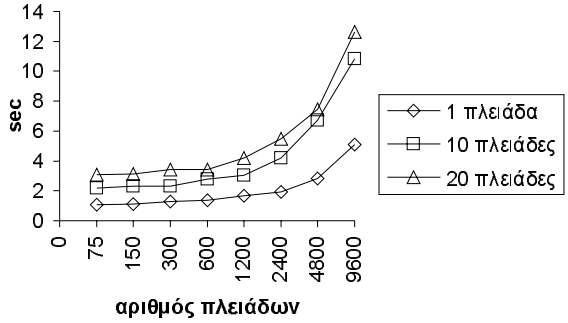
Σχήμα 5.6 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



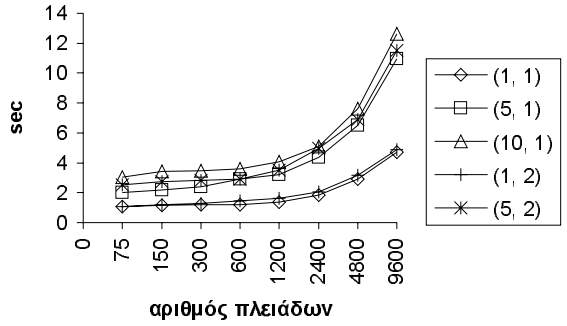
Σχήμα 5.7 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, χωρίς χρήση της πρότασης *PORTION*.



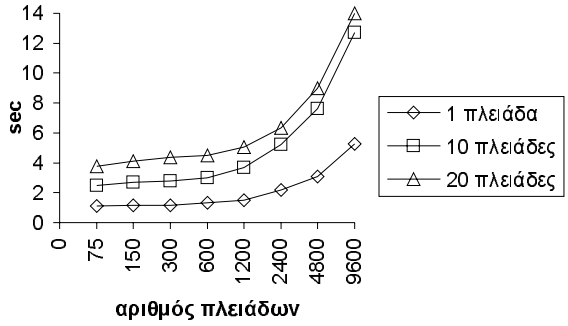
Σχήμα 5.8 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα χωρίς κλειδί, με χρήση της πρότασης *PORTION*.



Σχήμα 5.9 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



Σχήμα 5.10 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα με κλειδί, χωρίς χρήση της πρότασης *PORTION*.



Σχήμα 5.11 - Απόδοση του αλγόριθμου ενημέρωσης πίνακα με κλειδί, με χρήση της πρότασης *PORTION*.

Όπως προκύπτει από τη σύγκριση των ανωτέρω διαγραμμάτων με τα αντίστοιχα της παραγράφου 4.4.3, η υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών δεν επιφέρει εμφανή επίπτωση στην απόδοση του αλγορίθμου ενημέρωσης.

5.3. Προστασία προσωρινών πινάκων και επανόρθωση από καταρρεύσεις.

Στην παρούσα ενότητα περιγράφονται τεχνικές για την αντιμετώπιση ζητημάτων που ανακύπτουν λόγω των αλγορίθμων που παρουσιάστηκαν στην προηγούμενη ενότητα και αφορούν την προστασία των προσωρινών πινάκων και την επανόρθωση από καταρρεύσεις. Στην παράγραφο 5.3.1 περιγράφεται ένα σχήμα προστασίας των προσωρινών πινάκων από αυθαίρετες τροποποιήσεις και στην παράγραφο 5.3.2 παρουσιάζονται αλγόριθμοι για την καταστροφή των προσωρινών πινάκων που παρέμειναν στη βάση δεδομένων, λόγω καταρρεύσεων.

5.3.1. Προστασία των προσωρινών πινάκων.

Η ορθότητα των δεδομένων που περιέχονται στους προσωρινούς πίνακες είναι κεφαλαιώδους σημασίας για την επιτυχή εκτέλεση των αλγορίθμων που περιγράφηκαν στην ενότητα 5.1, κατά συνέπεια, είναι σημαντικό να αποτρέπεται η αυθαίρετη τροποποίηση των περιεχομένων των πινάκων αυτών από τους χρήστες. Επίσης, πρέπει να εμποδίζεται ακόμη και η ανάγνωση των πινάκων αυτών από τους χρήστες, δεδομένου ότι ενδέχεται να προκαλέσει την τοποθέτηση κλειδωμάτων, η οποία μπορεί με τη σειρά της να οδηγήσει σε σημαντικές καθυστερήσεις (η σύνοδος συστήματος θα πρέπει να περιμένει μέχρι να απομακρυνθούν τα κλειδώματα αυτά, πριν της επιτραπεί να καταστρέψει κάποιον προσωρινό πίνακα), ή ακόμη και σε αδιέξοδα.

Στη γενική περίπτωση, ο χρόνος ύπαρξης των προσωρινών πινάκων είναι περιορισμένος και οι χρήστες δεν γνωρίζουν τα ονόματά τους, συνεπώς η πιθανότητα να προσπελαστεί κάποιος προσωρινός πίνακας από χρήστη είναι μικρή. Είναι όμως δυνατόν, κατά την αποτίμηση μιας πολύπλοκης ερώτησης, ή σε μία περίοδο κατά την οποία ο φόρτος του συστήματος είναι αυξημένος, ένας χρήστης να λάβει γνώση του ονόματος κάποιου προσωρινού πίνακα (εξετάζοντας το λεξικό δεδομένων του ΣΔΒΔ) και να τον προσπελάσει ή να τον τροποποιήσει, μέσω μιας εντολής. Η προσπέλαση στους προσωρινούς πίνακες μπορεί να αποτραπεί με χρήση των ακόλουθων τεχνικών:

1. δημιουργείται ένας νέος χρήστης για το σχεσιακό ΣΔΒΔ, π.χ. *temporal*. Στον χρήστη αυτό παραχωρείται το δικαίωμα δημιουργίας πινάκων σε κάθε βάση που διαχειρίζεται το σχεσιακό ΣΔΒΔ. Κατά τη δημιουργία των συνόδων συστήματος, χρησιμοποιείται η πρόταση *IDENTIFIED BY* της εντολής *CONNECT* της εμφυτευμένης SQL, προκειμένου να οριστεί ότι οι σύνοδοι συστήματος θα ανήκουν στον νέο αυτό χρήστη.
2. αν ένας προσωρινός πίνακας *P1* πρέπει να προσπελαστεί μέσω μιας συνόδου χρήστη που ανήκει στον χρήστη *Χρήστης*, τότε πριν εκτελεστεί η εντολή (ή οριστεί ο δρομέας) που τον προσπελαύνει μέσω της συνόδου χρήστη, εκτελούνται οι εντολές

GRANT SELECT ON *P1* TO *Χρήστης*

COMMIT

LOCK TABLE *P1* IN EXCLUSIVE MODE

μέσω της συνόδου συστήματος, προκειμένου να παραχωρηθεί το δικαίωμα ανάγνωσης του πίνακα *P1* στον χρήστη *Χρήστης* και να κλειδωθεί ολόκληρος ο πίνακας *P1* με αποκλειστικό κλείδωμα. (Αν το σχεσιακό ΣΔΒΔ δεν διαθέτει την εντολή *LOCK TABLE*,

το ίδιο αποτέλεσμα μπορεί να επιτευχθεί, ορίζοντας ότι η διακριτότητα κλειδώματος της συνόδου συστήματος θα είναι επιπέδου πίνακα και ότι τα κλειδώματα ανάγνωσης της ίδιας συνόδου θα είναι αποκλειστικά. Στη συνέχεια, αρκεί να προσπελαστεί μία πλειάδα του πίνακα *P1* μέσω της συνόδου συστήματος, για να κλειδωθεί ολόκληρος ο πίνακας *P1* με αποκλειστικό κλείδωμα.) Όταν η σύνοδος χρήστη επιτελέσει τις απαραίτητες προσπελάσεις στον πίνακα *P1*, το δικαίωμα ανάγνωσης στον πίνακα αυτόν αφαιρείται από τον χρήστη *Χρήστης* μέσω των εντολών

REVOKE SELECT ON *P1* FROM *Χρήστης*

COMMIT

οι οποίες εκτελούνται μέσω της συνόδου συστήματος.

3. μετά από κάθε σημείο επικύρωσης της συνόδου συστήματος, όλοι οι προσωρινοί πίνακες που υπάρχουν στη βάση δεδομένων και για τους οποίους έχει παραχωρηθεί δικαίωμα ανάγνωσης, κλειδώνονται με αποκλειστικό κλείδωμα.

Η χρήση διαφορετικής ταυτότητας χρήστη για τη σύνοδο συστήματος προστατεύει τους προσωρινούς πίνακες από αυθαίρετες τροποποιήσεις, μια και, εξ ορισμού, μόνο ο χρήστης που δημιούργησε κάποιον πίνακα έχει δικαιώματα εισαγωγής, διαγραφής και ενημέρωσης δεδομένων για τον πίνακα αυτό. Το δικαίωμα ανάγνωσης στους προσωρινούς πίνακες παραχωρείται μόνο για τη χρονική περίοδο που είναι απολύτως απαραίτητο και, κατά την περίοδο αυτή, ο πίνακας είναι κλειδωμένος με αποκλειστικό κλείδωμα από τη σύνοδο συστήματος. Σύμφωνα με τα ανωτέρω, ο πίνακας μπορεί να προσπελαστεί από κάποια άλλη σύνοδο μόνο αν αυτή έχει καθορίσει ότι η προσπέλαση θα γίνεται χωρίς να εξετάζονται και να τίθενται κλειδώματα, γεγονός που εξαλείφει κάθε πιθανότητα καθυστέρησης ή/και αδιεξόδου.

5.3.2. Καταστροφή απομεινάντων προσωρινών πινάκων.

Είναι πιθανό, κατά τη διάρκεια εκτέλεσης μιας εντολής που απαιτεί τη δημιουργία προσωρινών πινάκων να καταρρεύσει ο χρονολογικός φλοιός, το σχεσιακό ΣΔΒΔ ή ο υπολογιστής στον οποίο εκτελείται κάποια από αυτές τις εφαρμογές. Δεδομένου ότι η σύνοδος συστήματος επικυρώνει τις δοσοληψίες της ενόσω η βάση δεδομένων περιέχει προσωρινούς πίνακες, το σχεσιακό ΣΔΒΔ θεωρεί τους πίνακες αυτούς μόνιμους και θα προσπαθήσει να τους προστατεύσει από τα αποτελέσματα της κατάρρευσης. Είναι έτσι απαραίτητο να χρησιμοποιηθεί κάποια μέθοδος για την καταστροφή των προσωρινών

πινάκων που έχουν παραμείνει στη βάση δεδομένων. Για την αντιμετώπιση του προβλήματος αυτού μπορεί να υιοθετηθεί μία από τις ακόλουθες προσεγγίσεις:

1. ακολουθείται μία *σύμβαση ονοματολογίας* για τους προσωρινούς πίνακες, π.χ. τα ονόματά τους αρχίζουν πάντα με τη συμβολοσειρά *tt_temp*. Κατά τη διαδικασία επανόρθωσης του σχεσιακού ΣΔΒΔ από μία κατάρρευση, ο διαχειριστής της βάσης δεδομένων εκτελεί ένα πρόγραμμα, το οποίο καταστρέφει όλους τους πίνακες της βάσης δεδομένων που ακολουθούν τη σύμβαση ονοματολογίας (τα ονόματά τους μπορούν να βρεθούν εξετάζοντας το λεξικό δεδομένων του σχεσιακού ΣΔΒΔ). Οι χρήστες θα πρέπει να είναι ενήμεροι σχετικά με τη σύμβαση ονοματολογίας, προκειμένου να μην δημιουργούν πίνακες που θα καταστραφούν από αυτή τη διαδικασία.
2. αν ο νέος χρήστης του σχεσιακού ΣΔΒΔ που δημιουργείται (ο χρήστης *temporal*) χρησιμοποιείται μόνο ως ιδιοκτήτης των συνόδων συστήματος και για κανέναν άλλο λόγο, τότε κατά τη διαδικασία επανόρθωσης του σχεσιακού ΣΔΒΔ από μία κατάρρευση, ο διαχειριστής της βάσης δεδομένων εκτελεί ένα πρόγραμμα, το οποίο καταστρέφει όλους τους πίνακες της βάσης δεδομένων που ανήκουν στον συγκεκριμένο χρήστη (τα ονόματά τους μπορούν να βρεθούν μέσω των καταλόγων συστήματος του σχεσιακού ΣΔΒΔ).
3. εισάγεται ένας νέος *κατάλογος συστήματος* (system catalogue), ο οποίος χρησιμοποιείται από τον χρονολογικό φλοιό για την αποθήκευση των ονομάτων των προσωρινών πινάκων που υπάρχουν ανά πάσα στιγμή στη βάση δεδομένων. Το όνομα κάθε προσωρινού πίνακα καταχωρείται στον νέο κατάλογο συστήματος *πριν* εκτελεστεί η αντίστοιχη εντολή *CREATE TABLE* που τον δημιουργεί και διαγράφεται μόνο αφού έχει ολοκληρωθεί η εκτέλεση της εντολής *DROP TABLE* που καταστρέφει τον πίνακα (η εισαγωγή και απομάκρυνση καταχωρίσεων από τον νέο κατάλογο συστήματος γίνεται μέσω της συνόδου συστήματος). Κατά τη διαδικασία επανόρθωσης, εκτελείται ένα πρόγραμμα που καταστρέφει τους πίνακες των οποίων τα ονόματα είναι καταχωρισμένα στον εν λόγω κατάλογο συστήματος.

5.4. Σύνοψη.

Στο κεφάλαιο αυτό παρουσιάσαμε μία μεθοδολογία για υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών, σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ, όπως αυτό που περιγράφεται στο κεφάλαιο 4. Οι τεχνικές που παρουσιάστηκαν αξιοποιούν τις δυνατότητες του σχεσιακού ΣΔΒΔ, που αφορούν την υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών.

Τα προβλήματα της αυτόματης επικύρωσης, της υπέρογκης αύξησης του μεγέθους του ημερολογίου και της άρσης των κλειδωμάτων αντιμετωπίζονται χρησιμοποιώντας μία δεύτερη σύνοδο προς το σχεσιακό ΣΔΒΔ, μέσω της οποίας δημιουργούνται και τροποποιούνται οι προσωρινοί πίνακες, χωρίς η εισαγωγή της συνόδου αυτής να επιφέρει επιβάρρυνση στην απόδοση του χρονολογικού ΣΔΒΔ. Τέλος, λαμβάνεται μέριμνα ώστε να μη δημιουργούνται αδιέξοδα, μια και διαφορετικές σύνοδοι δρούν ανταγωνιστικά για τα κλειδώματα στη βάση δεδομένων.

6. Συμπεράσματα - Μελλοντικές επεκτάσεις.

Στην παρούσα εργασία έγινε πλήρης παρουσίαση ενός ολοκληρωμένου χρονολογικού ΣΔΒΔ ιστορικού τύπου, το οποίο δίνει ιδιαίτερη έμφαση στην απόδοση. Το χρονολογικό ΣΔΒΔ που παρουσιάστηκε διαθέτει ισχυρό θεωρητικό υπόβαθρο, καθώς στηρίζεται στη σχεσιακή άλγεβρα χρόνου εγκυρότητας. Το μοντέλο της ΣΑΧΕ είναι μία συνεπής επέκταση του μοντέλου της σχεσιακής άλγεβρας, γεγονός που καθιστά εύκολη την κατανόησή του από τους χρήστες που γνωρίζουν το σχεσιακό μοντέλο.

Ως γλώσσα ορισμού και διαχείρισης δεδομένων του χρονολογικού ΣΔΒΔ χρησιμοποιείται η SQL-XE, μία συνεπής επέκταση της SQL89, τόσο από συντακτικής, όσο και από σημασιολογικής άποψης. Η SQL-XE περιλαμβάνει συντακτικές δομές για τον ορισμό πινάκων που περιέχουν δεδομένα με καθορισμένο χρόνο εγκυρότητας, καθώς και για τη διαχείριση των δεδομένων αυτών.

Τα κυριότερα νέα χαρακτηριστικά του χρονολογικού ΣΔΒΔ είναι τα ακόλουθα:

1. η υλοποίηση τμήματος της ΣΑΧΕ και συγκεκριμένα των πράξεων *EXCEPT*, *PUNION* και *SPLIT*.
2. η υλοποίηση ενός συντακτικού αναλυτή για τη γλώσσα SQL-XE. Ο συντακτικός αναλυτής ελέγχει τη συντακτική ορθότητα των εισαγομένων εντολών και διαμορφώνει κατάλληλες δομές δεδομένων, δίνοντας τη δυνατότητα στις διαδικασίες εκτέλεσης εντολών να προβούν στη σημασιολογική ανάλυση και στην εκτέλεση των εντολών.
3. ο σχεδιασμός βελτιστοποιημένων αλγορίθμων για την εκτέλεση των λειτουργιών εισαγωγής, διαγραφής και ενημέρωσης δεδομένων. Οι αλγόριθμοι αξιοποιούν τις δυνατότητες που προσφέρει η εμφυτευμένη SQL, σε αντίθεση με τους αλγεβρικούς αλγορίθμους που έχουν χρησιμοποιηθεί για τη σημασιολογική περιγραφή των πράξεων καθώς και παλαιότερες υλοποιήσεις (πρόγραμμα ORES), επιτυγχάνοντας έτσι σημαντικά καλύτερη απόδοση.
4. η προσαρμογή ενός μαθηματικού μοντέλου κόστους στη στρωματοποιημένη αρχιτεκτονική, για την εκτίμηση του κόστους εκτέλεσης των εντολών. Το προσαρμοσμένο μοντέλο λαμβάνει υπόψη τόσο το κόστος εισόδου/εξόδου, όσο και το κόστος διαδιεργασιακής επικοινωνίας μεταξύ του χρονολογικού φλοιού και του σχεσιακού ΣΔΒΔ.

5. ο σχεδιασμός τεχνικών για υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών σε ένα στρωματοποιημένο χρονολογικό ΣΔΒΔ. Οι προτεινόμενες τεχνικές αξιοποιούν τις δυνατότητες που παρέχονται από το σχεσιακό ΣΔΒΔ, ξεπερνώντας τους περιορισμούς που τίθενται στη χρήση εντολών της γλώσσας ορισμού δεδομένων μέσα σε δοσοληψίες και αντιμετωπίζοντας την υπέρμετρη αύξηση του μεγέθους του ημερολογίου. Βασίζοντας τις τεχνικές υποστήριξης ταυτοχρόνων χρηστών και δοσοληψιών στους μηχανισμούς που παρέχονται από το σχεσιακό ΣΔΒΔ, αποφεύγεται η εισαγωγή νέων κέντρων ελέγχου της ταυτόχρονης προσπέλασης και δεν καθίσταται απαραίτητη η τήρηση δεύτερου ημερολογίου, πέραν αυτού που τηρεί το σχεσιακό ΣΔΒΔ. Οι προτεινόμενες τεχνικές συμπληρώνονται με μεθόδους αποτροπής παρεμβολών, από μέρους των χρηστών, στην εκτέλεση των εντολών, καθώς και με επέκταση του μηχανισμού επανόρθωσης, ώστε να αντιμετωπίζονται ζητήματα που σχετίζονται με τους αλγόριθμους υλοποίησης των λειτουργιών του χρονολογικού ΣΔΒΔ.
6. η πλήρης υλοποίηση του χρονολογικού ΣΔΒΔ, βάσει της στρωματοποιημένης αρχιτεκτονικής που προτείνεται και με τη χρήση βελτιστοποιημένων αλγορίθμων. Η υλοποίηση του χρονολογικού ΣΔΒΔ ενσωματώνει επίσης τις τεχνικές για υποστήριξη δοσοληψιών και ταυτοχρόνων χρηστών σε ένα ΣΔΒΔ, καθώς και τις μεθόδους αποτροπής παρεμβολών στην εκτέλεση των ερωτήσεων και τις επεκτάσεις του μηχανισμού επανόρθωσης.

Όπως αναφέρθηκε, ιδιαίτερη έμφαση δόθηκε στην απόδοση του χρονολογικού ΣΔΒΔ. Για τη μελέτη της απόδοσης χρησιμοποιήθηκε το προσαρμοσμένο μοντέλο κόστους, βάσει του οποίου συγκρίθηκαν οι επιδόσεις των αλγορίθμων εκτέλεσης των εντολών εισαγωγής, διαγραφής και ενημέρωσης δεδομένων με τις αντίστοιχες επιδόσεις των αλγεβρικών αλγορίθμων. Η σύγκριση κατέδειξε ότι οι βελτιστοποιημένοι αλγόριθμοι είναι κατά πολύ ταχύτεροι των αλγεβρικών, αποφεύγοντας πλήθος συνδέσεων και ταξινομήσεων σχέσεων. Η βελτίωση της απόδοσης επιβεβαιώθηκε και πειραματικά, μετρώντας το χρόνο εκτέλεσης χαρακτηριστικών ερωτήσεων, πάνω σε πίνακες διαφορετικών μεγεθών, χρησιμοποιώντας τόσο τους βελτιστοποιημένους όσο και τους αλγεβρικούς αλγόριθμους.

Το προσαρμοσμένο μαθηματικό μοντέλο χρησιμοποιήθηκε για να εκτιμηθεί η επιβάρυνση που εισάγεται στο χρόνο εκτέλεσης από τη χρήση των τεχνικών για υποστήριξη δοσοληψιών ταυτοχρόνων χρηστών. Όπως διαπιστώθηκε, η επιβάρυνση που προκαλείται είναι αμελητέα και αυτό πιστοποιήθηκε με πειραματικές μετρήσεις, οι οποίες κατέδειξαν ότι η ενσωμάτωση

των τεχνικών υποστήριξης δοσοληψιών και ταυτοχρόνων χρηστών δεν επιφέρει εμφανή επίπτωση στην απόδοση των αλγορίθμων εισαγωγής, διαγραφής και ενημέρωσης δεδομένων. Οι μελλοντικές ερευνητικές μας προσπάθειες στα χρονολογικά ΣΔΒΔ εντοπίζονται στους ακόλουθους άξονες:

1. επέκταση του υλοποιηθέντος χρονολογικού ΣΔΒΔ ιστορικού τύπου σε αμφιχρονολογικό. Η επέκταση αυτή συνεπάγεται την εισαγωγή νέων πράξεων στη ΣΑΧΕ, τον εμπλουτισμό της SQL-XE τόσο από συντακτικής όσο και από σημασιολογικής πλευράς, καθώς και τον σχεδιασμό και την υλοποίηση των αλγορίθμων που θα διαχειρίζονται τον χρόνο δοσοληψίας.
2. υποστήριξη χρονικών διαστημάτων και χρονικών σημείων διαφορετικής διακριτότητας, με ταυτόχρονη μελέτη της δυνατότητας υποστήριξης πολλαπλών διαστάσεων χρόνου εγκυρότητας.
3. ο σχεδιασμός εμφυτευμένης SQL-XE, με ορισμό του συντακτικού για ορισμό δρομέων, των επιτρεπτών λειτουργιών και των αλγορίθμων εκτέλεσης ερωτήσεων.
4. ενσωμάτωση χρονολογικών χαρακτηριστικών σε ένα αντικειμενοστρεφές ΣΔΒΔ, με τον ορισμό των κατάλληλων κλάσεων (classes) και την ανάπτυξη των απαραίτητων μεθόδων (methods).
5. μελέτη των ιδιαιτεροτήτων που παρουσιάζουν τα κατανεμημένα χρονολογικά ΣΔΒΔ, σε σχέση με τα σχεσιακά και τα αντικειμενοστρεφή κατανεμημένα ΣΔΒΔ και σχεδιασμός εξειδικευμένων σχημάτων διαμέρισης (partitioning schemes) χρονολογικών δεδομένων, καθώς και κατανεμημένων αλγορίθμων επεξεργασίας ερωτήσεων σε χρονολογικά ΣΔΒΔ.

Βιβλιογραφία.

[Allen83] J. F. Allen, "Maintaining knowledge about temporal intervals" in *Communications of the ACM*, vol. 26, no. 11, November 1983, σελ. 832-843.

[Ariav86] G. Ariav, "A Temporally Oriented Data Model" in *ACM Transactions on Database Systems*, vol. 11, no. 4, December 1986, σελ. 499-527.

[Beech88] D. Beech and B. Mahbod, "Generalised Version Control in an Object Oriented Database" in *Proceedings of the International Conference on data Engineering*, 1988.

[Ben-Zvi82] J. Ben-Zvi, "The Time Relational Model", Ph.D. thesis, Computer Science Department, UCLA, 1982.

[Bolour83] A. Bolour, T. L. Anderson, L. J. Deleeyser and H. K. Wong, "The role of time in information processing: A Survey" in *ACM SIGMOD Record*, vol. 12, no. 3, 1983, σελ. 27-53.

[Carey88] M. J. Carey, D. J. DeWitt and S. L. Vandenburg, "A Data Model and Query Language for EXODUS" in *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, Chicago 1988, σελ. 413-423.

[Caruso88] M. Caruso and E. Sciore, "Meta-Functions and Contexts in an Object Oriented Database Language" in *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, Chicago 1988, σελ. 56-65.

[Caruso90] M. Caruso and E. Sciore, "The VISION Object Oriented Database Management System" in *Advances in Database Programming Languages*, ACM Press, New York, 1990.

[Clifford83] J. Clifford and D. S. Warren, "Formal Semantics for Time in Databases" in *ACM Transactions on Database Systems*, vol. 8, no. 2, June 1983, σελ. 214-254.

[Clifford87] J. Clifford and A. Croker, "The Historical Relational Data Model (HRDM) and Algebra Based on Lifespans" in *Proceedings of the Third*

International Conference on Data Engineering, February 1987, σελ. 528-537.

[Codd70] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks" in *Communications of the ACM*, vol. 13, no. 6, June 1970, σελ. 377-387.

[Codd85] E. F. Codd, "Is Your DBMS Really Relational?" in *Computer World*, October 1985.

[Date82] C. J. Date, "An Introduction to Database Systems", Addison-Wesley Publishing Company, 1982.

[Date85] C. J. Date, "An Introduction to Database Systems", Vol. II, Addison-Wesley Publishing Company, 1985.

[Elmarsi83] R. Elmarsi and G. Wiederhold, "GORDAS: A Formal High Level Query Language for the Entity-Relationship Model" in *Proceedings of the Conference on the Entity-Relationship Approach to Software Engineering*, 1983.

[Elmarsi90] R. Elmarsi and G. Wiederhold, "A Temporal Model and Query Language for ER Databases" in *Proceedings of the Sixth International Conference on Data Engineering*, 1990.

[ESPRIT93α] ESPRIT III Project 7224 (ORES) Deliverable C3: "Specification of Valid Time Formalism", April 1993.

[ESPRIT93β] ESPRIT III Project 7224 (ORES) Deliverable D2: "Specification of Valid Time SQL", April 1993.

[ESPRIT94] ESPRIT III Project 7224 (ORES) Deliverable D4.1, "Implementation of Valid Time SQL", April 1994.

[Gadia88] S. K. Gadia, "A Homogeneous Relational Model and Query Languages for Temporal Databases" in *ACM Transactions on Database Systems*, vol. 12, no. 4, December 1988, σελ. 418-448.

[Gadia92] S. K. Gadia, "A Seamless Generic Extension of SQL for Querying Temporal Data", Technical Report TR-92-02, Computer Science Department, Iowa State University, May 1992.

[Graefe93] Goetz Graefe, "Query Evaluation Techniques for Large Databases" in *ACM Computing Surveys*, vol. 25, no. 2, June 1993.

[Ingres91] Ingres Corporation, “Ingres SQL and ESQL Reference Manual” (for version 6.4), 1991.

[Jensen91] C. S. Jensen, L. Mark and N. Roussopoulos, “Incremental Implementation Model for Relational Databases with Transaction Time” in *IEEE Transactions on Knowledge and Data Engineering*, vol. 3, no. 4, December 1991, σελ. 461-463.

[Jensen92] C. S. Jensen, J. Clifford, S. K. Gadia, A. Segev and R. T. Snodgrass, “A glossary of temporal database concepts” in *SIGMOD Record*, vol. 21, no. 3, September 1992, σελ. 35-43.

[Jensen93] C. S. Jensen, J. Clifford, R. Elmarsi, S. K. Gardia P. Hayes and S. Jajodia, “A Consensus Glossary of Temporal Database Concepts”, Technical Report (R93-2035), Department of Mathematics and Computers.

[Jones79] S. Jones, P. Mason and R. Stamper, “LEGOL 2.0: A Relational Specification Language for Complex Rules”, in *Information Systems*, vol. 4, 1979, σελ. 293-305.

[Kδfer92] W. Kδfer and H. Shφning, “Realising a Temporal Complex-Object Data Model” in *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, San Diego 1992, σελ. 266-275.

[Kline93] N. Kline, “An Update of the Temporal Database Bibliography” in *SIGMOD Record* vol. 22, no. 4, 1993, σελ. 66-80.

[Korth86] H. F. Korth, A. Silberschatz, “Database System Concepts”, McGraw-Hill Book Company, 1986.

[Lans88] R. F. van der Lans, “The SQL standard. A Complete Reference”, Prentice Hall, 1988.

[Lorentzos88] N. A. Lorentzos, “A Formal Extension of the Relational Model for the Representation of Generic Intervals”, Ph.D. Thesis, Birkbeck College, University of London, 1988.

[Lorentzos93] N. A. Lorentzos, “The Interval Extended Relational Model and its Application to Valid Time Databases” in *Temporal Databases: Theory, Design and Implementation*, J. Clifford, R. Snodgrass et al. (editors), Benjamin Cummings, 1993, σελ. 67-91.

[McKenzie86] E. McKenzie, "Bibliography: Temporal databases" in *SIGMOD Record*, vol. 15, no. 4, 1986, σελ. 40-52.

[McKenzie87] E. McKenzie and R. Snodgrass, "Supporting Valid Time: An Historical Algebra", Technical Report (TR87-008), Department of Computer Science, North Carolina University, August1987.

[Navathe89] S. B. Navathe and R. Ahmed, "A Temporal Relational Model and a Query Language" in *Information Sciences*, no. 49, 1989, σελ. 147-175.

[Navathe93] S. B. Navathe and R. Ahmed, "Temporal Extensions to the Relational Model and SQL" in *Temporal Databases: Theory, Design and Implementation*, J. Clifford, R. Snodgrass et al. (editors), Benjamin Cummings, 1993, σελ. 92-109.

[Oracle90] ORACLE Corporation, "SQL Language Reference Manual" (for version 6.0), 1990.

[Rose91] E. Rose and A. Segev, "TODM-A Temporal Object-Oriented Data Model with Temporal Constraints" in *Proceedings of the Tenth International Conference on the Entity-Relationship Approach*, 1991.

[Rose93] E. Rose and A. Segev, "TOOSQL-A Temporal Object-Oriented Query Language" in *Proceedings of the International Conference on the Entity-Relationship Approach*, Dallas, 1993.

[Sarda90] N. L. Sarda, "Extensions to SQL for Historical Databases" in *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 2, June 1990, σελ. 220-230.

[Segev87] A. Segev and A. Shoshani, "Logical Modeling of Temporal Data", in *Proceedings of ACM SIGMOD*, San Francisco, May 1987.

[Schueler77] B. Schueler, "Update reconsidered, Architecture and Models", in *Database Management Systems*, G. M. Nijssen (editor), North Holland, 1977, σελ. 49-164.

[Sciore91] E. Sciore, "Using Annotations to Support Multiple Kinds of Versioning in an Object Oriented System", in *ACM Transactions on Database Systems*, vol. 16, no. 3, 1991.

[Sciore94] E. Sciore, "Versioning and Configuration Management in an Object Oriented Data Model" in *VLDB Journal*, vol. 3, no. 1, 1994, σελ. 77-106.

[Snodgrass86] R. Snodgrass and I. Ahn, "Temporal Databases" in *IEEE Computer*, vol. 19, no. 9, September 1986, σελ. 35-42.

[Snodgrass87] R. Snodgrass, "The Temporal Query Language TQUEL" in *ACM Transactions on Database Systems*, vol. 12, no. 2, July 1987, σελ. 247-298.

[Snodgrass94α] The TSQL2 Language Design Commitee (R. Snodgrass, I. Ahn, G. Ariav, D. Batory, J, Clifford, C. Dyreson, R. Elmarsi, F. Grandi, C. Jensen, W. Kδfer, N. Kline, K. Kulkarni, T, Leung, N. Lorentzos, J. Roddick, A. Segev, M. Soo, S. Sripada), "The TSQL2 Data Model", in *A TSQL2 Commentary*, September 1994.

[Snodgrass94β] R. T. Snodgrass, "Temporal Object-Oriented Databases: A Critical Comparison", in *In Modern Database Management Systems*, Won Kim (editor), Addison Wesley, 1995, σελ. 386-408.

[Snodgrass94γ] The TSQL2 Language Design Commitee (R. Snodgrass, I. Ahn, G. Ariav, D. Batory, J, Clifford, C. Dyreson, R. Elmarsi, F. Grandi, C. Jensen, W. Kδfer, N. Kline, K. Kulkarni, T, Leung, N. Lorentzos, J. Roddick, A. Segev, M. Soo, S. Sripada), "User Defined Time in TSQL 2", in *A TSQL2 Commentary*, September 1994.

[Snodgrass94δ] R. T. Snodgrass et al, "TSQL2 Language Specification", in *SIGMOD Record*, vol. 23, no. 1, Μάρτιος 1994, σελ. 65-86.

[Soo91] M. D. Soo, "Bibliography on Temporal Databases" in *SIGMOD Record* vol. 20, no. 1, 1991, σελ. 14-23.

[Stam88] R. Stam and R. Snodgrass, "Bibliography on Temporal Databases" in *IEEE Data Engineering*, vol. 11, no. 4, 1988, σελ. 53-61.

[Stonebreaker87] M. Stonebreaker, "The Design of the POSTGRES Storage System", in *Proceedings of the Conference on Very Large Databases*, Brighton, September 1987.

[Stonebreaker90] M. Stonebreaker, L. Rowe and M. Hirohama, "The Implementation of Postgress" in *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 1, 1990, σελ. 125-142.

[Su91] S. Y. W. Su and H. M. Chen, "A Temporal Knowledge Representation Model OSAM*T and its Query Language OQL/T", in *Proceedings of the Conference on Very Large Databases*, 1991.

[Sybase90] Sybase Inc., "Transact SQL User's Guide" (for version 4.2), 1990.

[Taha93] K. K. Al-Taha, R. T. Snodgrass and M. D. Soo, "Bibliography on Spatiotemporal Databases", in *SIGMOD Record* vol. 22, no. 1, March 1993.

[Tanenbaum92] A. S. Tanenbaum, "Modern Operating Systems", Prentice Hall Inc., 1992.

[Tansel86] A. U. Tansel, "Adding Time Dimension to Relational Model and Extending Relational Algebra", in *Information Systems*, vol. 11, no. 4, 1986.

[Tansel91] A. U. Tansel, "An Historical Query Language" in *Information Sciences*, vol. 53, 1991, σελ. 101-133.

[Wuu92] G. Wuu and U. Dayal, "A Uniform Model for Temporal Object Oriented Databases" in *Proceedings of the International Conference on Data Engineering*, 1992.

Γλωσσάρι.

anti-join	αντισύνδεση
attribute	ιδιοχαρακτηριστικό
bitemporal DBMSs	αμφιχρονολογικά ΣΔΒΔ
chronon	χρονικό κβάντο
class	κλάση
commit point	σημείο επικύρωσης
complex	σύνθετος
computation	υπολογισμός
conceptual schema	ιδεατό σχήμα
concurrent users	ταυτόχρονοι χρήστες
cursor	δρομέας
declarative query language	δηλωτική γλώσσα ερωτήσεων
difference	διαφορά
duplicate	διπλότυπος
embedded	εμφυτευμένος
entity integrity rule	κανόνας ακεραιότητας οντοτήτων
escape sequence	ακολουθία διαφυγής
expressiveness	εκφραστικότητα
external schema	εξωτερικό σχήμα
fan-in factor	παράγων διακλάδωσης εισόδου
fold	συμπτύσσω
folding	σύμπτυξη
front-end application	μετωπική εφαρμογή
granularity	διακριτότητα
hash-based aggregation	συνάθροιση με κερματισμό
hash-join	σύνδεση με κερματισμό
hashing	κερματισμός
historical DBMS	ΣΔΒΔ ιστορικού τύπου
holding point	σημείο αναμονής
implicit commit point	σημείο αυτόματης επικύρωσης

inconsistent	ασυνεπής
infix	ενδοθεματικός
integrity	ακεραιότητα
interprocess communication	διαδιεργασιακή επικοινωνία
interval	διάστημα
interval-type key	κλειδί τύπου διάστημα
join	σύνδεση
lexical token	λεκτικό σύμβολο
method	μέθοδος
merge-join	σύνδεση με συγχώνευση
merge-sort	συγχωνευτική ταξινόμηση
message passing	μεταβίβαση μηνύματος
minimality	ελαχιστότητα
module	τμήμα
nested	ένθετος
nested loops	ένθετοι βρόχοι
normalisation	κανονικοποίηση
normalise	κανονικοποιώ
object file	καταληκτικό αρχείο
partitioning scheme	σχήμα διαμέρισης
pipeline	σωλήνωση, σωληνωτή επεξεργασία
point difference	σημειακή διαφορά
point union	σημειακή ένωση
point-type key	κλειδί τύπου σημείο
pointer-based join	σύνδεση βασισμένη σε δείκτες
projection	προβολή
qualified (column name)	χαρακτηρισμένο (όνομα στήλης)
quantifier	ποσοδείκτης
query execution plan	σχέδιο εκτέλεσης ερώτησης
record identifier	προσδιοριστής εγγραφής
recovery	επανόρθωση
relation	σχέση
rollback DBMS	ΣΔΒΔ επαναφοράς

savepoint	σημείο επαναφοράς
schema	σχήμα
selection	επιλογή
semaphore	σημαφόρος
session	σύνοδος
shared memory	διαμοιραζόμενη μνήμη
snapshot DBMS	ΣΔΒΔ στιγμιοτύπου
sorting	ταξινόμηση
source code	πρωτογενής κώδικας
split	κατάτμηση
state transition table	πίνακας μετάβασης καταστάσεων
system catalogue	κατάλογος συστήματος
table	πίνακας
temporal data	χρονολογικά δεδομένα
temporal element	χρονολογικό στοιχείο
temporal engine	χρονολογικός φλοιός
temporal homogeneity	χρονολογική ομοιογένεια
time dimension	χρονική διάσταση
time interval	χρονικό διάστημα
time point	χρονικό σημείο
timestamp	χρονική ένδειξη
transaction	δοσοληψία
transaction time	χρόνος δοσοληψίας
unfold	αναπτύσσω
unfolding	ανάπτυξη
union	ένωση
union compatible	συμβατός ως προς την ένωση
unique	μοναδικός
unqualified (column name)	χωρίς χαρακτηρισμό (όνομα στήλης)
user-defined time	οριζόμενος από τον χρήστη χρόνος
valid time	χρόνος εγκυρότητας
valid time data	δεδομένα καθορισμένου χρόνου εγκυρότητας
view	όψη