

**Συγκριτική Ανάλυση Συστημάτων Διαχείρισης
Βάσεων Δεδομένων ως προς την δυνατότητα
αποθήκευσης και διαχείρισης χωρικών δεδομένων**

ΚΟΤΣΑΜΠΑΣΙΔΗΣ ΔΗΜΗΤΡΗΣ

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων Καθηγητής : Γεώργιος Ευαγγελίδης

Εξεταστές : Μάγια Σατρατζέμη

Διατμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών στα Πληροφοριακά Συστήματα

Πανεπιστήμιο Μακεδονίας

Θεσσαλονίκη

10/2008

Περίληψη

Η ραγδαία ανάπτυξη και εξάπλωση των Συστημάτων Διαχείρισης Βάσεων Δεδομένων οφείλεται κυρίως στο γεγονός ότι αποτελούν ουσιαστικά την πηγή πληροφοριών και τον τροφοδότη όλων των λογισμικών και των εφαρμογών που χειρίζονται δεδομένα. Επιπλέον τα συστήματα αυτά παρέχουν την απαραίτητη ασφάλεια, αξιοπιστία, ταχύτητα και οργάνωση που είναι απαραίτητα στοιχεία όταν πρόκειται για μεγάλο όγκο δεδομένων. Παράλληλα ολοένα και περισσότερο αυξάνεται η χρήση λογισμικών που χειρίζονται γεωγραφικά δεδομένα. Ο όγκος των δεδομένων αυτών που είναι ακόμη μεγαλύτερος σε σχέση με τα υπόλοιπα δεδομένα επιβάλλει την δυνατότητα επέκτασης των Συστημάτων Διαχείρισης Βάσεων Δεδομένων ώστε να παρέχουν την δυνατότητα αποθήκευσης και αξιοποίησης των δεδομένων αυτών. Σε αυτό το γεγονός βασίζεται και η παρούσα ανάλυση με την οποία θα προσπαθήσουμε να συγκρίνουμε τρία διαφορετικά λογισμικά και να διαπιστώσουμε πως ανταποκρίνονται στην διαχείριση των γεωμετρικών δεδομένων.

ΠΕΡΙΕΧΟΜΕΝΑ

1 Συστήματα Διαχείρισης Βάσεων Δεδομένων	5
2 Πορεία ανάλυσης	8
3 Διεθνής οργανισμός OGC(Open Geospatial Consortium)	15
4 Βασικοί τύποι χωρικών δεδομένων	18
4.1 Τύποι χωρικών δεδομένων στην βάση MySQL.....	18
4.2 Τύποι χωρικών δεδομένων στην βάση PostgreSQL.....	21
4.3 Τύποι χωρικών δεδομένων στην βάση Oracle.....	22
5 Δημιουργία και Αποθήκευση Δεδομένων	24
5.1 Δημιουργία και αποθήκευση δεδομένων στην βάση MySQL.....	24
5.2 Δημιουργία και αποθήκευση δεδομένων στην βάση PostgreSQL.....	27
5.3 Δημιουργία και αποθήκευση δεδομένων στην βάση Oracle.....	30
6 Μορφές αναπαράστασης των χωρικών δεδομένων	34
6.1 Μορφές αναπαράστασης δεδομένων στην βάση MySQL.....	34
6.2 Μορφές αναπαράστασης δεδομένων στην βάση PostgreSQL.....	35
6.3 Μορφές αναπαράστασης δεδομένων στην βάση Oracle.....	36
7 Συναρτήσεις	38
7.1 1 ^η Κατηγορία Συναρτήσεων.....	39
7.1.1 1 ^η Κατηγορία συναρτήσεων της βάσης MySQL.....	39
7.1.2 1 ^η Κατηγορία συναρτήσεων της βάσης PostgreSQL.....	49
7.1.3 1 ^η Κατηγορία συναρτήσεων της βάσης Oracle.....	53
7.2 2 ^η Κατηγορία Συναρτήσεων.....	57
7.2.1 2 ^η Κατηγορία συναρτήσεων της βάσης MySQL.....	57
7.2.2 2 ^η Κατηγορία συναρτήσεων της βάσης PostgreSQL.....	62
7.2.3 2 ^η Κατηγορία συναρτήσεων της βάσης Oracle.....	69
7.3 3 ^η Κατηγορία Συναρτήσεων.....	78
7.3.1 3 ^η Κατηγορία συναρτήσεων της βάσης PostgreSQL.....	79
7.3.2 3 ^η Κατηγορία συναρτήσεων της βάσης Oracle.....	82

8 Χρήση Δεικτών.....	85
8.1 Χρήση Δεικτών στην βάση MySQL.....	85
8.2 Χρήση Δεικτών στην βάση PostgreSQL.....	86
8.3 Χρήση Δεικτών στην βάση Oracle 85.....	87
9 Συμπεράσματα.....	89
Βιβλιογραφία.....	95

ΚΕΦΑΛΑΙΟ 1

Συστήματα Διαχείρισης Βάσεων Δεδομένων

Πριν ξεκινήσουμε την ανάλυση των Συστημάτων Διαχείρισης Βάσεων Δεδομένων που μας απασχολούν στην παρούσα μελέτη θα αναφερθούμε σε κάποια βασικά στοιχεία για να κατανοήσουμε την λειτουργία και τον σκοπό των βάσεων δεδομένων καθώς και των συστημάτων που τις διαχειρίζονται και εν συνεχεία θα περιγράψουμε το κάθε ένα από τα λογισμικά που πρόκειται να μας απασχολήσουν καθώς και τον τρόπο με τον οποίο παρέχουν υποστήριξη χωρικών δεδομένων.

Ένας σύντομος ορισμός για τις βάσεις δεδομένων είναι ότι αποτελούν ένα οργανωμένο τρόπο αποθήκευσης δεδομένων και πρόσβασης σε αυτά μέσω του κατάλληλου λογισμικού. Στην ουσία είναι ένα ολοκληρωμένο σύστημα που αποτελείται από δεδομένα, το κατάλληλο λογισμικό και το υλικό που βοηθούν στην εύρεση των κατάλληλων πληροφοριών από τους χρήστες. Το πρόγραμμα που διαχειρίζεται τις βάσεις δεδομένων λέγεται Σύστημα Διαχείρισης Βάσεων Δεδομένων.

Βασικοί στόχοι των βάσεων δεδομένων είναι :

- Η αποφυγή της πολλαπλής αποθήκευσης ίδιων δεδομένων (redundancy)
- Η δυνατότητα καταμερισμού των στοιχείων της σε πολλούς χρήστες (sharing)
- Η επιβολή κανόνων ασφαλείας (security)
- Η διατήρηση της ακεραιότητας (integrity) και της αξιοπιστίας (reliability) των δεδομένων
- Η ανεξαρτησία των δεδομένων (data independence) και των προγραμμάτων από τον φυσικό τρόπο αποθήκευσης των δεδομένων
- Η ομοιομορφία (uniformity) στον χειρισμό και την αναπαράσταση των δεδομένων.

Τα βασικά χαρακτηριστικά μιας βάσης δεδομένων αποτελούνται από τα εξής στοιχεία :

- **Πεδίο** (*Field*), είναι το μικρότερο κομμάτι δεδομένων στο οποίο μπορούμε να αναφερθούμε και περιέχει ένα μόνο χαρακτηριστικό ή ιδιότητα ενός στοιχείου της βάσης δεδομένων.
- **Εγγραφή** (*Record*), είναι ένα σύνολο από διαφορετικά πεδία που περιέχει όλες τις πληροφορίες για ένα στοιχείο της βάσης δεδομένων.
- **Αρχείο** (*File*), είναι ένα σύνολο από πολλά παρόμοια στοιχεία (εγγραφές) της βάσης δεδομένων.
- **Πρωτεύον Κλειδί** (*Primary Key*), είναι ένα πεδίο ή συνδυασμός πεδίων που χαρακτηρίζει μοναδικά μια εγγραφή.
- **Κλειδί** (*Key*), είναι ένα πεδίο που δεν έχει κατ' ανάγκη μοναδική τιμή και που μπορούμε να το χρησιμοποιήσουμε για να κάνουμε αναζήτηση σ' ένα αρχείο.
- **Ξένο Κλειδί** (*Foreign Key*), είναι ένα πεδίο που έχει το ίδιο σύνολο τιμών με το πρωτεύον κλειδί ενός άλλου αρχείου.

Τα εργαλεία χειρισμού πληροφοριών μιας βάσης δεδομένων είναι γνωστά και σαν "*Γλώσσες Εντολών*" και με τη βοήθειά τους μπορούμε να δώσουμε εντολές χειρισμού των δεδομένων. Η πιο γνωστή και ευρέως διαδεδομένη γλώσσα εντολών για τις σύγχρονες βάσεις δεδομένων είναι η *Δομημένη Γλώσσα Ερωτήσεων SQL (Structured Query Language)*, η οποία αποτελείται από τα εξής μέρη :

DDL (*Data Definition Language, Γλώσσα Ορισμού Δεδομένων*), με την οποία καθορίζουμε τις δομές και τα τμήματα μιας βάσης δεδομένων.

DML (*Data Manipulation Language, Γλώσσα Χειρισμού Δεδομένων*), με την οποία επεξεργαζόμαστε τα δεδομένα μιας βάσης δεδομένων.

DCL (*Data Control Language, Γλώσσα Ελέγχου Δεδομένων*), με την οποία εξασφαλίζουμε την ασφάλεια και την ακεραιότητα των δεδομένων μιας βάσης δεδομένων.

Οι βάσεις δεδομένων που θα μας απασχολήσουν στην μελέτη είναι σχεσιακές βάσεις.

Στις Σχεσιακές (Relational) βάσεις δεδομένων, τα δεδομένα συνδέονται μεταξύ τους με σχέσεις (relations), οι οποίες προκύπτουν από τα κοινά πεδία που υπάρχουν σε διαφορετικά αρχεία. Τα αρχεία αποκαλούνται πίνακες (tables), οι εγγραφές γραμμές (rows) και τα πεδία στήλες (columns). Η ύπαρξη μιας κοινής τιμής στα πεδία δύο αρχείων καθορίζει και μια σχέση μεταξύ των γραμμών διαφορετικών πινάκων. Οι σχεσιακές βάσεις δεδομένων έχουν το πλεονέκτημα ότι είναι λογικά κατανοητές και πολύ ευέλικτες και δεκτικές σε αλλαγές.

Τα τρία συστήματα διαχείρισης βάσεων δεδομένων που θα μελετηθούν ως προς την ικανότητα αποθήκευσης και διαχείρισης χωρικών δεδομένων είναι τα πλέον αντιπροσωπευτικά. Τα δύο εξ' αυτών είναι ανοιχτού κώδικα ελεύθερα λογισμικά από τα πιο ευρέως διαδεδομένα η MySQL και η PostgreSQL και το τρίτο είναι μια έκδοση της εταιρείας Oracle (Express Edition) που διανέμεται δωρεάν. Συγκεκριμένα οι εκδόσεις των λογισμικών που χρησιμοποιήθηκαν είναι η MySQL Server 5.0 η οποία παρέχει υποστήριξη για χωρικά δεδομένα χωρίς την ανάγκη για εγκατάσταση κάποιου πρόσθετου πακέτου, η PostgreSQL 8.2 η οποία δίνει την επιλογή στον χρήστη κατά την εγκατάσταση της να επιλέξει εάν θα εγκαταστήσει ένα πρόσθετο πακέτο για την υποστήριξη χωρικών δεδομένων που ονομάζεται PostGIS. Κάθε έκδοση της PostgreSQL περιλαμβάνει αυτό το πακέτο δίνει όμως την δυνατότητα στον χρήστη να μην το επιλέξει κατά την εγκατάσταση αλλά να το εγκαταστήσει αργότερα σαν ξεχωριστό πακέτο για να έχει την δυνατότητα να εγκαθιστά την πιο ενημερωμένη έκδοση. Το πακέτο PostGIS υποστηρίζεται από τον οργανισμό Refraction Research ο οποίος ειδικεύεται στην ανάπτυξη συστημάτων διαχείρισης βάσεων δεδομένων και ιδιαίτερα σε χωρικά δεδομένα. Το τρίτο λογισμικό που χρησιμοποιήθηκε είναι η έκδοση Oracle express 10g το οποίο διανέμεται δωρεάν από την εταιρεία Oracle και παρέχει ένα πακέτο υποστήριξης χωρικών δεδομένων που ονομάζεται Oracle Locator το οποίο έχει σαφώς λιγότερες δυνατότερες από το πακέτο Oracle Spatial που χρησιμοποιείται στην πλήρη έκδοση του λογισμικού.

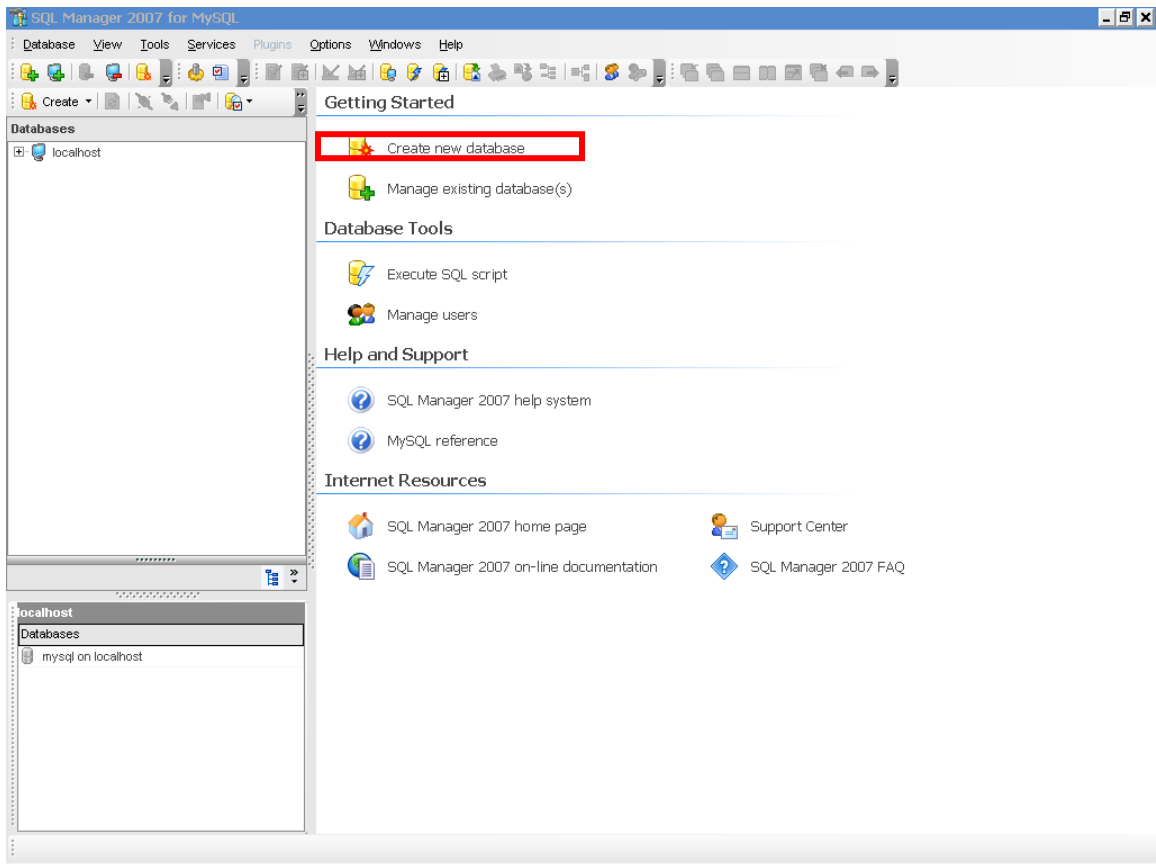
ΚΕΦΑΛΑΙΟ 2

Πορεία ανάλυσης

Η ανάλυση των συστημάτων διαχείρισης βάσεων δεδομένων θα χωριστεί σε επιμέρους κεφάλαια ώστε να μελετώνται και τα τρία λογισμικά παράλληλα και σε ένα συγκεκριμένο τομέα κάθε φορά για να γίνεται όσο το δυνατόν πιο ουσιαστική η σύγκριση και η ανάλυση των δυνατοτήτων του καθενός από αυτά. Σε κάθε κεφάλαιο θα αναπτύσσονται οι δυνατότητες του κάθε συστήματος και στο τέλος θα γίνεται μια σύγκριση μεταξύ των τριών ώστε να είναι προφανή τα πλεονεκτήματα και μειονεκτήματα του καθενός. Το πρώτο πράγμα που θα μας απασχολήσει στα Συστήματα αυτά είναι να δούμε τους βασικούς τύπους των γεωμετρικών δεδομένων που υποστηρίζουν και τις μορφές αναπαράστασης των δεδομένων αυτών σε κάθε λογισμικό, στη συνέχεια θα αναλύσουμε τον τρόπο που δημιουργούνται τα γεωμετρικά δεδομένα δηλαδή το σύνολο των συναρτήσεων που μπορούμε να χρησιμοποιήσουμε για να δημιουργήσουμε γεωμετρικά δεδομένα εσωτερικά σε κάθε σύστημα καθώς και τον τρόπο που αποθηκεύονται τα δεδομένα αυτά στην βάση. Το επόμενο βήμα μετά την παρουσίαση των διαφορετικών τύπων γεωμετρικών δεδομένων και του τρόπου δημιουργίας και αποθήκευσης τους σε κάθε σύστημα είναι να αναλύσουμε την πληθώρα των συναρτήσεων που διαθέτει το καθένα για την επεξεργασία και ανάλυση των δεδομένων. Λόγω της ύπαρξης πολλών συναρτήσεων και για την καλύτερη κατανόηση τους οι συναρτήσεις θα ομαδοποιηθούν σε κατηγορίες και κάθε κατηγορία θα αποτελέσει ξεχωριστό κεφάλαιο ανάλυσης παραθέτοντας και σχετικά γραφήματα για να γίνουν περισσότερο κατανοητές οι λειτουργίες των εκάστοτε συναρτήσεων. Στη συνέχεια θα εξετάσουμε και την ταχύτητα που επιτελούνται οι λειτουργίες σε κάθε σύστημα όπως αναζήτηση και επεξεργασία δεδομένων διότι αποτελεί πολύ κρίσιμο σημείο η συμπεριφορά των συστημάτων όταν έχουμε να κάνουμε με πληθώρα δεδομένων με αποτέλεσμα να έχει δραματικές επιπτώσεις στην απόδοση της βάσης. Στο τέλος θα γίνει μία σύνοψη των επιμέρους αυτών γραφημάτων και πινάκων για την εξαγωγή συμπερασμάτων σχετικά με την συνολική απόδοση και λειτουργικότητα καθενός από τα συστήματα αυτά. Σε όλα τα διαγράμματα θα χρησιμοποιήσουμε μια κλίμακα

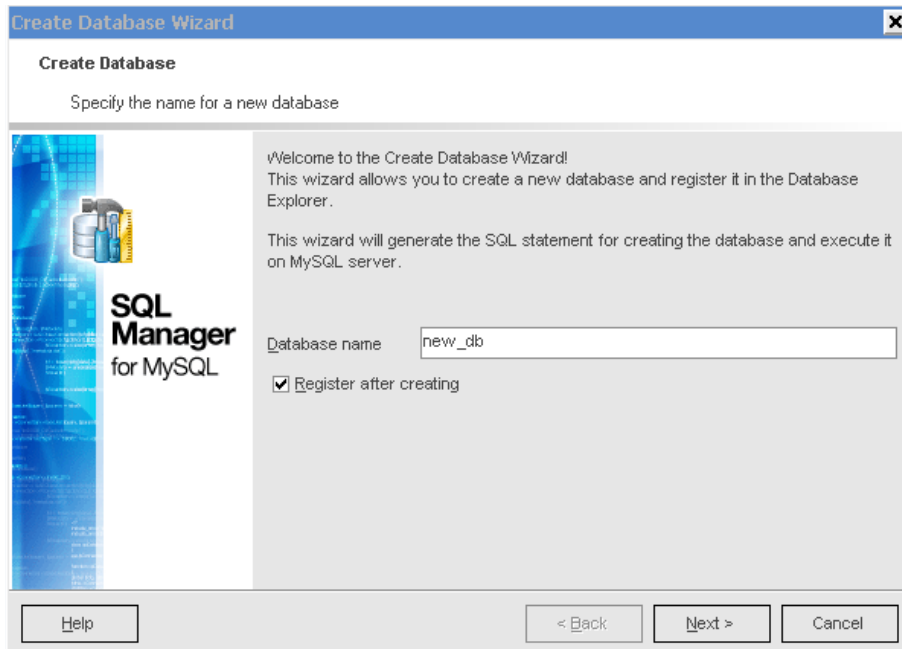
βαθμονόμησης που θα αποτελείται από τους αριθμούς ένα, δύο και τρία που αντιστοιχούν στη λειτουργικότητα του συστήματος διαχείρισης χωρικών δεδομένων σε κάθε τομέα σύγκρισης. Δηλαδή σε κάθε κεφάλαιο που θα αναλύουμε συγκεκριμένα χαρακτηριστικά των συστημάτων ο αριθμός ένα θα αντιστοιχεί σε χαμηλή απόδοση-λειτουργικότητα συστημάτων, το δύο σε μέτρια απόδοση και τέλος το τρία σε πολύ καλή για να υπάρχει ένα μέτρο σύγκρισης ώστε στο τέλος να δούμε συνολικά την απόδοση των συστημάτων βασιζόμενοι στα επιμέρους διαγράμματα.

Τα εργαλεία που θα χρησιμοποιήσουμε στην ανάλυση πέραν των τριών πακέτων λογισμικού που αναφέρθηκαν παραπάνω είναι το πακέτο λογισμικού SQL Manager της εταιρείας EMS το οποίο αποτελεί ένα γραφικό περιβάλλον διασύνδεσης (GUI) που προσφέρει στον χρήστη ένα γραφικό περιβάλλον, εύκολο στη χρήση και πιο λειτουργικό. Η εταιρεία EMS διαθέτει ξεχωριστό πακέτο λογισμικού για να καλύψει κάθε ένα από τα Συστήματα Διαχείρισης Χωρικών Δεδομένων τα οποία διατίθενται για χρήση από την εταιρεία δωρεάν αλλά για περιορισμένο χρόνο ώστε να τα αξιολογήσει ο χρήστης και να προχωρήσει στην αγορά του πακέτου. Πριν ξεκινήσουμε θα περιγράψουμε τις βασικές ρυθμίσεις που πρέπει να κάνουμε για να μπορέσουμε να χρησιμοποιήσουμε τα λογισμικά αυτά πάνω από τα Συστήματα Διαχείρισης Βάσεων Δεδομένων. Η πρώτη εικόνα που θα συναντήσουμε τρέχοντας το πακέτο αυτό φαίνεται παρακάτω. Θα χρησιμοποιήσουμε το πακέτο SQL Manager για την βάση MySQL ως παράδειγμα σχετικά με το πως χρησιμοποιείται το λογισμικό αυτό.



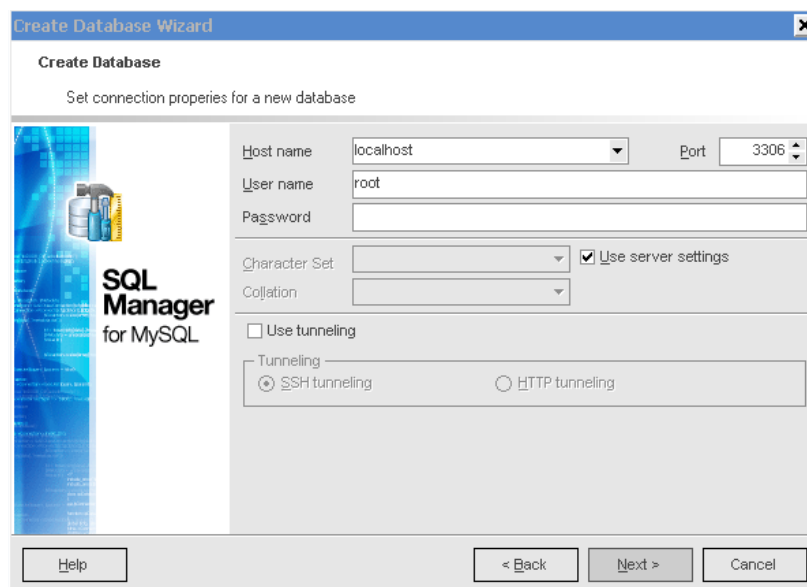
Θα αναφερθούμε στις ενέργειες που πρέπει να γίνουν από την στιγμή που θα τρέξουμε για πρώτη φορά το λογισμικό αυτό μέχρι να είμαστε σε θέση να ξεκινήσουμε να δουλεύουμε με το λογισμικό αυτό την βάση δεδομένων.

Αρχικά για να δημιουργήσουμε και μετά να συνδεθούμε με την βάση προχωράμε ως εξής. Επιλέγουμε 'Create new Database' όπως φαίνεται στην εικόνα παραπάνω με το κόκκινο πλαίσιο, στην συνέχεια στο παράθυρο που θα εμφανιστεί όπως φαίνεται στην επόμενη εικόνα δίνουμε το όνομα της νέας μας βάσης και πατάμε συνέχεια.

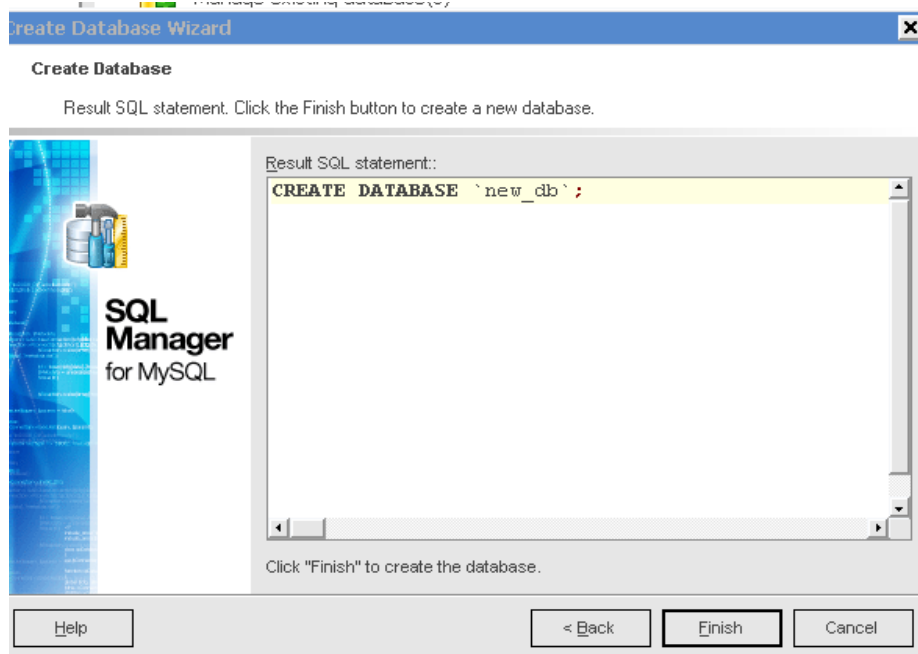


Όπως βλέπουμε είναι ήδη επιλεγμένο και το τετραγωνάκι για την καταχώρηση της νέας μας βάσης μετά την δημιουργία της, το οποίο το αφήνουμε ως έχει.

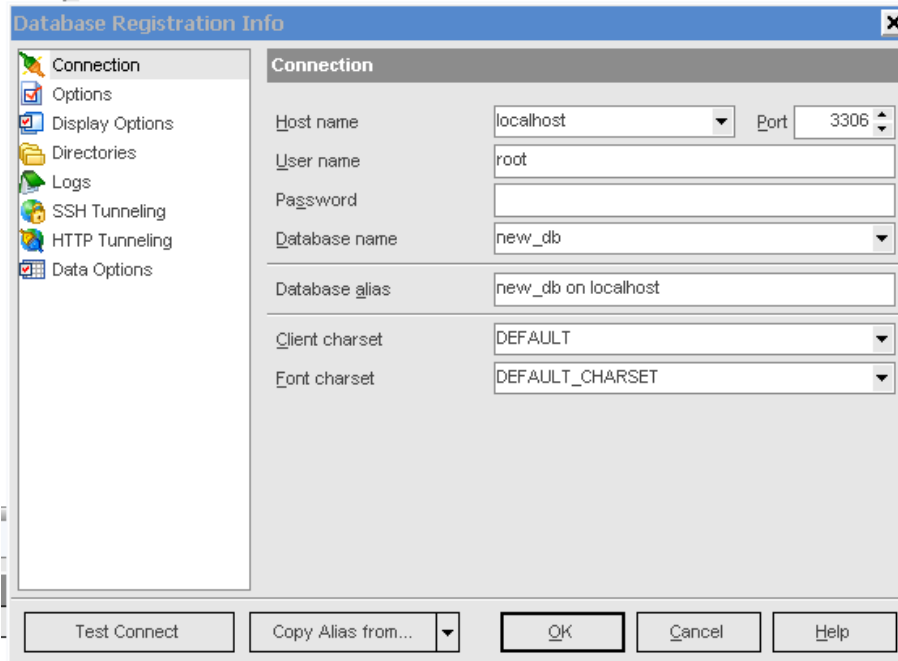
Το επόμενο παράθυρο που εμφανίζεται όπως φαίνεται στην επόμενη εικόνα μας ζητά να εισάγουμε το όνομα χρήστη και τον κωδικό του. Τις υπόλοιπες παραμέτρους τις αφήνουμε.



Μετά την εισαγωγή του επιθυμητού ονόματος χρήστη καθώς και του κωδικού πατάμε συνέχεια και στο παράθυρο που εμφανίζεται όπως φαίνεται και στην επόμενη εικόνα μας ζητά να επιβεβαιώσουμε την δημιουργία της βάσης.

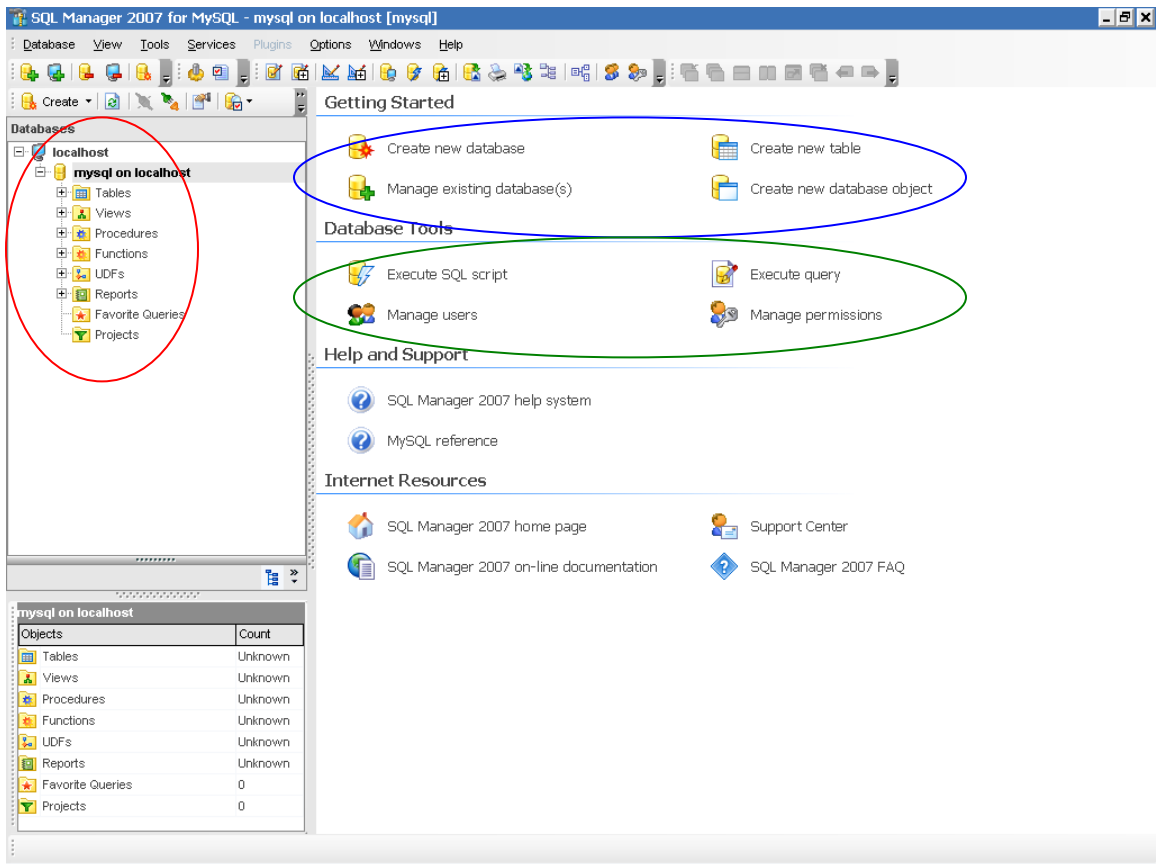


Επιλέγουμε ολοκλήρωση και δημιουργήσαμε την βάση στην οποία θα εργαστούμε. Το τελευταίο βήμα πριν αρχίσουμε να εργαζόμαστε με την βάση που δημιουργήσαμε είναι όπως φαίνεται στην παρακάτω εικόνα είναι να ελέγξουμε τα στοιχεία που δηλώσαμε μέχρι στιγμής, να ρυθμίσουμε κάποιες παράμετρους σχετικά με την σύνδεση ή με το γραφικό περιβάλλον και πλέον πατώντας 'ok' είμαστε έτοιμοι να εργαστούμε.



Πλέον περνάμε στο κεντρικό περιβάλλον εργασίας του λογισμικού και θα κάνουμε μια σύντομη αναφορά στις δυνατότητες και την ευκολία που παρέχει στο χρήστη να χειριστεί με αυτό την αντίστοιχη βάση δεδομένων.

Οι διαδικασίες που ακολουθήσαμε μέχρι στιγμής ισχύουν για όλες τις εκδόσεις του λογισμικού που όπως είπαμε κάθε έκδοση αντιστοιχεί και σε μία συγκεκριμένη βάση δεδομένων.



Στην παραπάνω εικόνα φαίνεται το περιβάλλον εργασίας του πακέτου αυτού όσον αφορά το Σύστημα Διαχείρισης Βάσεων Δεδομένων MySQL. Αριστερά μέσα στον κόκκινο κύκλο φαίνεται η βάση μας και τα χαρακτηριστικά της, όπως οι πίνακες, οι όψεις και οι συναρτήσεις. Αριστερά μέσα στον μπλε κύκλο περιλαμβάνονται οι λειτουργίες σχετικά με την δυνατότητα δημιουργίας μιας νέας βάσης, την δυνατότητα διαχείρισης μια υπάρχουσας βάσης ή την δημιουργία νέων πινάκων. Τέλος στον πράσινο κύκλο φαίνονται τα εργαλεία διαχείρισης μιας βάσης που είναι να εκτελέσουμε ένα πρόγραμμα, να δημιουργήσουμε και να τρέξουμε ένα ερώτημα με την βοήθεια της γλώσσας ερωτημάτων, να διαχειριστούμε τους χρήστες και τέλος να διαχειριστούμε και τις άδειες χρήσης.

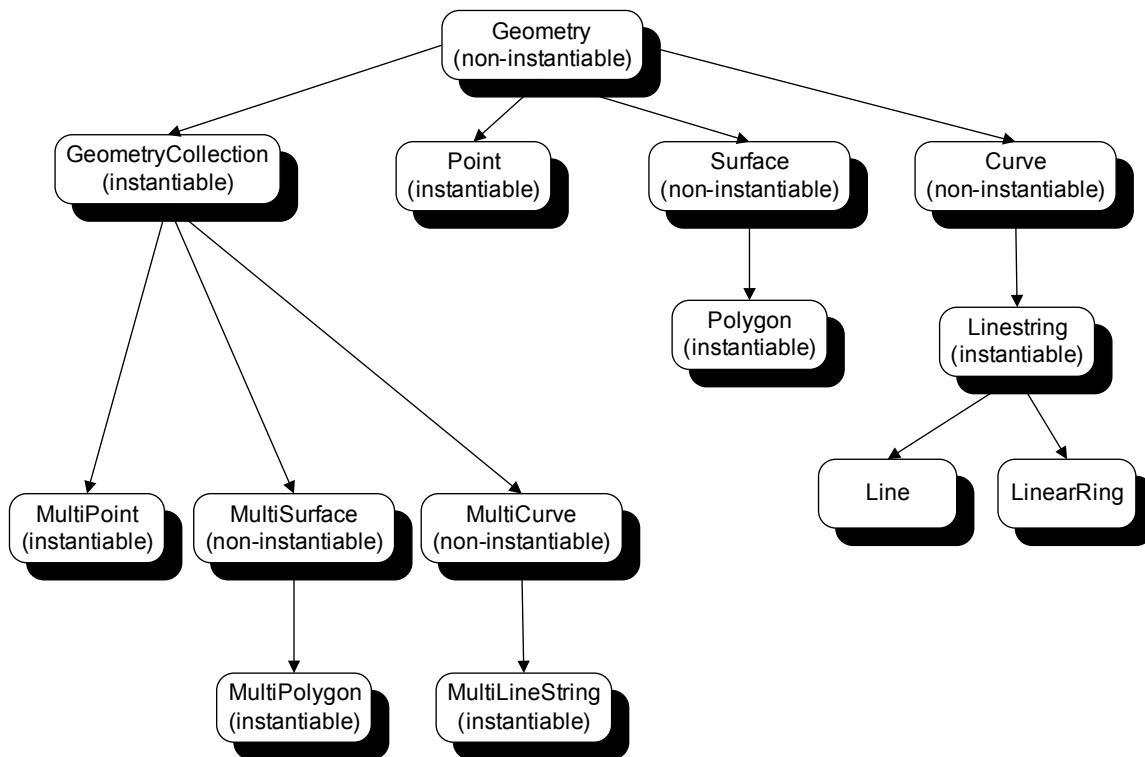
ΚΕΦΑΛΑΙΟ 3

Διεθνής οργανισμός OGC(Open Geospatial Consortium)

Πριν ξεκινήσουμε την ανάλυση των συστημάτων αυτών θα πρέπει να αναφερθούμε στον οργανισμό OGC ένα διεθνή μη-κερδοσκοπικό εθελοντικό οργανισμό προτυποποίησης ο οποίος αποτελεί το προϊόν της συνεργασίας διάφορων εταιρειών και πανεπιστημίων και ηγείται στην προσπάθεια ανάπτυξης προτύπων σχετικά με χωρικά δεδομένα και υπηρεσίες βασισμένες σε τοποθεσία ανεξάρτητες από κάποιο συγκεκριμένο λογισμικό. Σκοπός του οργανισμού είναι η ανάπτυξη θεμελιωδών λύσεων που μπορούν να χρησιμοποιηθούν σε κάθε είδους εφαρμογή που χειρίζεται χωρικά δεδομένα. Όσον αφορά τις σχεσιακές βάσεις δεδομένων ο οργανισμός αυτός έχει εκδώσει πληθώρα εγγράφων και προτυποποιήσεων με τα οποία προτείνει διάφορους θεμελιώδης τρόπους ώστε να επεκταθεί ένα σχεσιακό σύστημα διαχείρισης βάσης δεδομένων και να υποστηρίζει χωρικά δεδομένα. Ο οργανισμός αυτός ορίζει ένα σύνολο προδιαγραφών σχετικά με τους γεωμετρικούς τύπους δεδομένων, τους τρόπους αναπαράστασης αυτών, ενός συνόλου συναρτήσεων και μεθόδων και γενικότερα όλων των λειτουργιών ενός συστήματος διαχείρισης βάσεων δεδομένων σχετικά με χωρικά δεδομένα τα οποία θα μας απασχολήσουν στην παρούσα ανάλυση.

Θα κάνουμε μια σύντομη αναφορά στα βασικά στοιχεία που προτείνονται από τον οργανισμό σχετικά με τα γεωμετρικά αντικείμενα στα οποία θα αναφερθούμε ξανά αργότερα πιο αναλυτικά κατά την μελέτη των λογισμικών διότι όπως θα παρατηρήσουμε στην συνέχεια όλα τα λογισμικά σε μικρό ή μεγαλύτερο βαθμό ακολουθούν την προτυποποίηση του οργανισμού αυτού.

Θα ξεκινήσουμε από την περιγραφή της βασικής ιεραρχίας κλάσεων των γεωμετρικών δεδομένων που υποστηρίζονται από τον οργανισμό και φαίνονται στο παρακάτω διάγραμμα :



Στο διάγραμμα αυτό παρουσιάζονται οι κλάσεις που αντιστοιχούν σε γεωμετρικά αντικείμενα καθώς και ο συσχετισμός τους με άλλες κλάσεις από τις οποίες είτε κληρονομούν ιδιότητες και μεθόδους είτε κληρονομούν τις δικές τους στις υπό-κλάσεις τους. Επίσης πάνω σε κάθε κλάση αναφέρονται ποιες από αυτές μπορούν να δημιουργήσουν αντικείμενα και ποιες απλώς κληρονομούν ιδιότητες και μεθόδους. Αυτό είναι το βασικό γεωμετρικό μοντέλο που ορίζεται από τον οργανισμό (OGC). Στα γεωμετρικά αντικείμενα που δημιουργούνται σύμφωνα με το παραπάνω μοντέλο ο οργανισμός προτείνει δύο βασικές μορφές αναπαράστασης που αντιπροσωπεύουν τα γεωμετρικά αντικείμενα οι οποίες είναι:

- Αναπαράσταση κειμένου (Well-Known Text format «WKT»)
- Δυαδική αναπαράσταση (Well-Known Binary format «WBT»)

Η αναπαράσταση κειμένου (WKT) σχεδιάστηκε για να μετατρέπει τα χωρικά δεδομένα σε μορφή ASCII. Παρακάτω δίνονται σχετικά παραδείγματα από διάφορους τύπους δεδομένων.

- ✓ Point(1 1) Γεωγραφικό Σημείο
- ✓ LineString(1 2,3 3,4 5) Γραμμή που αποτελείται από δύο τμήματα
- ✓ Polygon((1 1,2 1,2 2,1 2,1 1)) Πολύγωνο μόνο με εξωτερικό δακτύλιο
- ✓ Multipoint(1 1,3 4,5 7) Σύνολο τριών σημείων
- ✓ GeometryCollection(Point(2 2), Linestring(2 3,7 8)) Σύνολο ενός σημείου και ενός ευθύγραμμου τμήματος.

Η μορφή δυαδικής αναπαράστασης (WBT) μετατρέπει τα χωρικά δεδομένα σε δυαδικές ροές δεδομένων, είναι σαφώς πιο δυσνόητη για αυτό αναφέρουμε το πιο απλό παράδειγμα γεωμετρικού αντικειμένου, ένα σημείο.

Η μετατροπή του γεωμετρικού σημείου Point(1 1) στην μορφή δυαδικής αναπαράστασης δίνει ως αποτέλεσμα τον παρακάτω αριθμό

01010000 00000000 000000F0 3F000000 000000F0 3F ο οποίος είναι σε δεκαεξαδική μορφή και αποτελεί συνολικά είκοσι-ένα byte δεδομένων που αναλύονται περαιτέρω ως εξής:

Το πρώτο byte χρησιμοποιείται για την δήλωση σειράς δεδομένων

Τα επόμενα τέσσερα byte για τον τύπο του χωρικού δεδομένου

Στη συνέχεια τα επόμενα οκτώ byte για την συντεταγμένη στον X άξονα

Και τέλος τα τελευταία οκτώ byte για την συντεταγμένη στον Y άξονα

Σύνολο $1+4+8+8=21$ byte

Συγκεκριμένα στο παραπάνω παράδειγμα η ανάλυση είναι ως εξής:

Byte Order : 01	}	Σε δεκαεξαδική μορφή
WKB type : 01000000		
X coord. : 000000000000F03F		
Y coord. : 000000000000F03F		

Είδαμε λοιπόν πως ορίζονται τα γεωμετρικά αντικείμενα και τις μεθόδους αναπαράστασης αυτών. Επίσης ο οργανισμός προτείνει και μια πληθώρα μεθόδων-συναρτήσεων γεωμετρικών αντικειμένων που έχουν σκοπό να μας δώσουν πληροφορίες για τα ίδια τα γεωμετρικά αντικείμενα να ελέγξουν την σχέση μεταξύ των αντικειμένων

ή και μεθόδους που υποστηρίζουν χωρική ανάλυση. Τις συναρτήσεις αυτές θα τις δούμε αναλυτικότερα παρακάτω αφού πρώτα τις χωρίσουμε σε κατηγορίες ώστε να τις μελετήσουμε λεπτομερώς.

Η σύντομη αναφορά στον οργανισμό (OGC) και τις βασικές αρχές που αυτός προτείνει σχετικά με την επέκταση μιας σχεσιακής βάσης δεδομένων ώστε να υποστηρίζει χωρικά δεδομένα έγινε διότι και οι τρεις βάσεις δεδομένων που αποτελούν αντικείμενο της παρούσας ανάλυσης ενσωματώνουν σε μικρότερο ή μεγαλύτερο βαθμό η κάθε μία τις βασικές αρχές της.

ΚΕΦΑΛΑΙΟ 4

Βασικοί τύποι χωρικών δεδομένων που υποστηρίζονται σε κάθε Βάση Δεδομένων

Σε αυτό το κεφάλαιο θα ασχοληθούμε αποκλειστικά με τους γεωμετρικούς τύπους δεδομένων που υποστηρίζουν τα λογισμικά.

4.1 Τύποι χωρικών δεδομένων στη βάση MySQL

Θα ξεκινήσουμε με την MySQL η οποία υποστηρίζει τους γεωμετρικούς τύπους δεδομένων που περιγράφονται στο γεωμετρικό μοντέλο που προτείνει ο διεθνής οργανισμός OGC. Θα διαχωρίσουμε τους τύπους σε δύο κατηγορίες. Στην πρώτη κατηγορία ανήκουν όσοι από αυτούς έχουν την δυνατότητα αποθήκευσης ενός δεδομένου και στην δεύτερη κατηγορία όσοι από αυτούς έχουν την δυνατότητα να αποθηκεύσουν πολλαπλά δεδομένα.

➤ Τύποι με δυνατότητα αποθήκευσης μιας τιμής

- Geometry : Ο τύπος αυτός αντιστοιχεί στην κλάση geometry που είναι η αρχική κλάση του γεωμετρικού μοντέλου της ιεραρχίας κλάσεων του OGC. Δεν μπορούμε να δημιουργήσουμε στιγμιότυπο αυτής της κλάσης απλά αποτελεί υπερ-κλάση των υπολοίπων στις οποίες κληρονομεί τις μεθόδους και ιδιότητες της. Επίσης όλες οι υπό-κλάσεις της που μπορούν να δημιουργήσουν στιγμιότυπα περιορίζονται σε αντικείμενα μηδενικών, ενός ή δύο διαστάσεων. Η κλάση αυτή έχει κάποιες βασικές ιδιότητες :

- i. Το σύστημα συντεταγμένων στο οποίο αναφέρεται το κάθε αντικείμενο(Spatial Reference Identifier, SRID), πρόκειται για ένα ακέραιο αριθμό που συνοδεύει κάθε τύπο δεδομένων για να το συσχετίζει με το Σύστημα Συντεταγμένων.
- ii. Τις συντεταγμένες του αντικειμένου. Κάθε μη-μηδενικό αντικείμενο έχει τουλάχιστον ένα ζεύγος συντεταγμένων που αντιστοιχούν στο Σύστημα που καθορίζεται από τον αριθμό SRID.
- iii. Τον χώρο που καταλαμβάνει, το όριο του και τον εξωτερικό χώρο. Κάθε αντικείμενο έχει κάποια θέση στον χώρο. Εξωτερικό χώρο εννοούμε όλο τον χώρο που δεν καταλαμβάνεται από το αντικείμενο.
- iv. Το ελάχιστο τετράγωνο του που αποτελείται από ένα αντικείμενο που σχηματίζεται από τις ελάχιστες και μέγιστες συντεταγμένες του. ((MINX MINY, MAXX MINY, MAXX MAXY, MINX MAXY, MINX MINY))
- v. Το αν το αντικείμενο είναι απλό ή όχι. Κάθε μια από τις υπό-κλάσεις που υποστηρίζει αυτή την ιδιότητα την χειρίζεται και διαφορετικά όπως θα δούμε παρακάτω
- vi. Εάν το αντικείμενο είναι κλειστό ή όχι
- vii. Εάν το αντικείμενο είναι κενό ή όχι. Ένα αντικείμενο θεωρείται κενό εάν δεν έχει κανένα σημείο.
- viii. Η διάσταση του που μπορεί να έχει τιμές -1 για κενό αντικείμενο,0 για αντικείμενο χωρίς μήκος και εμβαδό,1 για αντικείμενο με μήκος αλλά μηδενικό εμβαδό και τέλος 2 για αντικείμενο με μη-μηδενικό εμβαδό.

- Point : Αποθηκεύει γεωμετρικά σημεία. Τα αντικείμενα αυτά όπως γνωρίζουμε είναι μηδενικών διαστάσεων και αποτελούνται από ένα ζευγάρι συντεταγμένων. Η αναπαράσταση κειμένου που αντιστοιχεί σε ένα σημείο με τετμημένη και τεταγμένη ένα είναι **Point(1 1)**.
 - Linestring : Πρόκειται για τον τύπο που χρησιμοποιείται για την αποθήκευση όλων των τύπων γραμμών. Είναι μιας διάστασης και η αναπαράσταση κειμένου που αντιστοιχεί σε μία γραμμή αποτελούμενη από δύο συνεχόμενα ευθύγραμμα τμήματα **Linestring(1 1,4 5,7 8)**
 - Polygon : Περιλαμβάνει όλων των ειδών τα πολύγωνα. Πρόκειται για αντικείμενα δύο διαστάσεων. Η παρακάτω αναπαράσταση αντιστοιχεί σε ένα πολύγωνο το οποίο διαθέτει ένα εξωτερικό και ένα εσωτερικό δακτύλιο που διαχωρίζονται μεταξύ τους με κόμμα. Πάντοτε όταν πρόκειται για πολύγωνο με δύο δακτυλίους αναφερόμαστε πρώτα στον εξωτερικό. **Polygon((1 1,5 1,5 5,1 5,1 1),(2 2,3 2,3 3,2 3,2 2))**
- Τύποι με δυνατότητα αποθήκευσης πολλαπλών τιμών
- Multipoint : Πολλαπλά γεωμετρικά σημεία
 - Multilinestring : Πολλαπλές γραμμές
 - Multipolygon : Διάφορα πολύγωνα
 - Geometrycollection : Αυτός ο τύπος δεδομένων αποτελεί την μοναδική εξαίρεση διότι μπορούμε να αποθηκεύσουμε πολλαπλά δεδομένα ίδιου ή και διαφορετικών τύπων

Η μορφή της αναπαράστασης κειμένου σε αυτούς τους τύπους δεδομένων φαίνεται στα παραδείγματα που ακολουθούν.

MultiPoint(1 1,3 3,5 5) Αποτελείται από τρία διαφορετικά σημεία τα οποία διαχωρίζονται με κόμμα

MultiLinestring((1 1,4 5),(1 2,3 4)) Αποτελείται από δύο ξεχωριστές γραμμές η κάθε μία εκ των οποίων αντιστοιχεί σε ένα μόνο ευθύγραμμο τμήμα

MultiPolygon(((1 3,3 1,3 3,1 3,1 1)),((2 6,4 6,6 6,6 4,2 6))) Αυτή η κλάση παρουσιάζει κάποια ιδιαιτερότητα και για να γίνει πιο κατανοητή πήραμε την πιο απλή περίπτωση στην οποία έχουμε δύο ξεχωριστά πολύγωνα τα οποία αποτελούνται από ένα εξωτερικό δακτύλιο μόνο.

GeometryCollection(Linestring(1 1,2 2),Point(4 4)) Ο τύπος αυτός περιλαμβάνει δύο ξεχωριστά γεωμετρικά δεδομένα διαφορετικού τύπου που αποτελούνται από ένα ευθύγραμμο τμήμα και ένα σημείο

Είδαμε λοιπόν ότι η MySQL σε ότι αφορά τους τύπους δεδομένων ακολουθεί το μοντέλο του οργανισμού OGC χωρίς διαφοροποιήσεις

4.2 Τύποι χωρικών δεδομένων στην βάση PostgreSQL

Όσον αφορά την PostgreSQL είδαμε παραπάνω ότι για να μας παρέχει υποστήριξη χωρικών δεδομένων θα πρέπει να εγκαταστήσουμε το επιπλέον λογισμικό PostGIS το οποίο αναθεωρείται πολύ συχνά και παρέχει πλήρη υποστήριξη σε όλες τις προδιαγραφές που ορίζονται στο γεωμετρικό μοντέλο του (OGC). Υποστηρίζει λοιπόν όλους του τύπους δεδομένων τους οποίους αναλύσαμε παραπάνω και τους αναφέρω επιγραμματικά.

- Point
- LineString
- Polygon
- MultiPoint
- MultiLinestring
- MultiPolygon
- GeometryCollection

Επίσης υποστηρίζει και τις δύο μορφές αναπαράστασης δεδομένων, την αναπαράσταση κειμένου και την δυαδική αναπαράσταση καθώς επίσης και τις μεθόδους για μετατροπή από την μια μορφή σε άλλη. Έτσι θα αναφερθούμε μόνο σε ένα παράδειγμα λίγο πιο πολύπλοκο, το οποίο δεν είδαμε παραπάνω και αποτελείται από την κλάση MultiPolygon στην οποία αποθηκεύουμε δύο πολύγωνα που διαθέτουν δύο δακτυλίους το καθένα

MultiPolygon(((0 0 ,4 0,4 4,0 4,0 0),(1 1,2 1,2 2,1 2,1 1)),(-1 -1,-1 -2,-2 -2,-2 0, --1 -1),(0 0,1 2,2 2,2 1,0 0)))

Η PostgreSQL παρέχει πλήρη υποστήριξη στους βασικούς τύπους γεωμετρικών δεδομένων και επίσης υποστηρίζει δεδομένα τριών και τεσσάρων διαστάσεων.

4.3 Τύποι χωρικών δεδομένων στην βάση Oracle

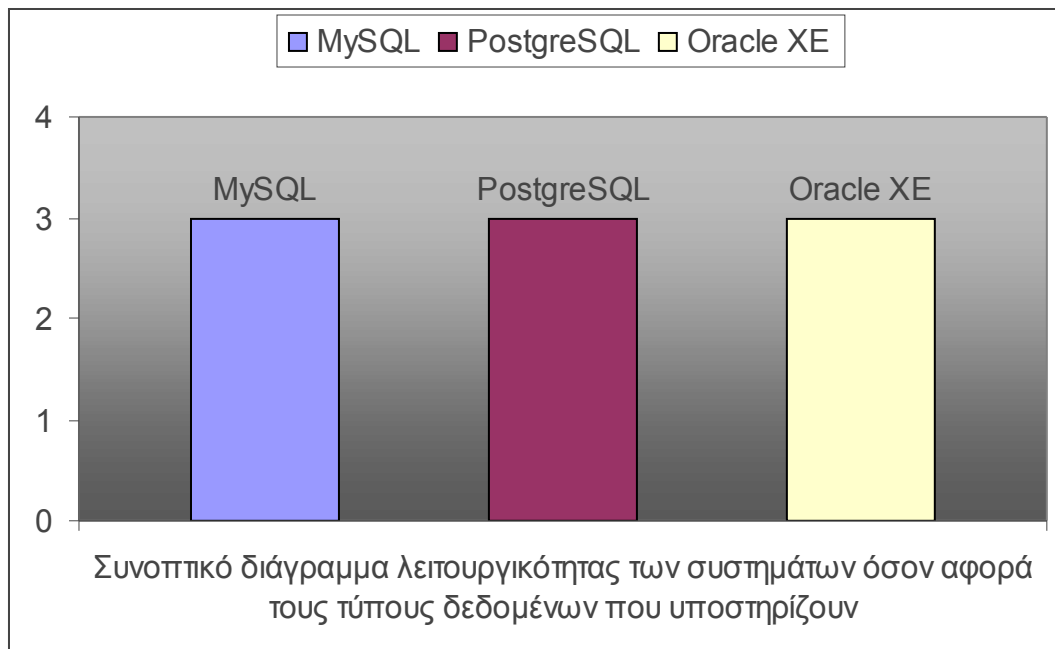
Τέλος και η Oracle υποστηρίζει τους βασικούς τύπους χωρικών δεδομένων και η αναπαράσταση κειμένου είναι η ίδια με τις υπόλοιπες διότι όλα τα λογισμικά ακολουθούν τις προδιαγραφές του διεθνούς οργανισμού προτυποποίησης (OGC).

- UNKNOWN_GEOMETRY
- POINT
- LINE or CURVE
- POLYGON
- COLLECTION
- MULTIPOINT
- MULTILINE or MULTICURVE
- MULTIPOLYGON

Θα αναφέρουμε και εδώ ένα παράδειγμα για να δούμε ότι όσον αφορά τους γεωμετρικούς τύπους και τα τρία υπό εξέταση συστήματα δεν διαφοροποιούνται σχεδόν καθόλου.

POLYGON ((5.0 1.0, 8.0 1.0, 8.0 6.0, 5.0 7.0, 5.0 1.0)) Για ένα πολύγωνο με ένα δακτύλιο όπως αναφέρθηκε και παραπάνω.

Διαπιστώνουμε ότι και τα τρία λογισμικά υποστηρίζουν τους τύπους δεδομένων που αναφέρονται από τον οργανισμό OGC καθώς επίσης και την ίδια αναπαράσταση κειμένου. Άρα λοιπόν όσον αφορά τους βασικού τύπους δεδομένων που υποστηρίζονται από τα εν λόγω συστήματα καθώς και τις μορφές αναπαράστασης των τύπων αυτών δεν υπάρχουν διαφοροποιήσεις οπότε όπως φαίνεται και στο παρακάτω διάγραμμα τα τρία συστήματα είναι στο ίδιο επίπεδο.



Όπως αναφέραμε και στην πορεία ανάλυσης τα διαγράμματα που θα χρησιμοποιούμε σε κάθε κεφάλαιο για την σύγκριση των συστημάτων σε κάθε συγκεκριμένο τομέα θα έχουν σύστημα βαθμολόγησης ένα, δύο και τρία για να είναι προφανής η απόδοση συνολικά των συστημάτων σε κάθε κεφάλαιο ώστε στο τέλος να είναι εύκολο να δούμε την συνολική εικόνα του καθενός. Εδώ επειδή και τα τρία υποστηρίζουν το σύνολο των τύπων δεδομένων που ορίζονται από τον Διεθνή Οργανισμό OGC βαθμολογήθηκαν με τρία.

ΚΕΦΑΛΑΙΟ 5

Δημιουργία και Αποθήκευση Δεδομένων

Ένα πολύ ιδιαίτερο και σημαντικό κομμάτι είναι αυτό που αφορά τον τρόπο που διαχειρίζονται οι βάσεις τα γεωμετρικά δεδομένα και κυρίως ο τρόπος που αποθηκεύονται εσωτερικά στο σύστημα. Σε αυτό το σημείο λοιπόν θα εξετάσουμε τις δυνατότητες που μας παρέχουν τα λογισμικά για την δημιουργία γεωμετρικών δεδομένων αλλά και τον τρόπο που αποθηκεύονται τα δεδομένα αυτά εσωτερικά στο κάθε σύστημα.

5.1 Δημιουργία και αποθήκευση Δεδομένων στην βάση MySQL

Ξεκινώντας και πάλι από την MySQL το πρώτο πράγμα που πρέπει να κάνουμε για να μπορέσουμε να δημιουργήσουμε και να αποθηκεύσουμε γεωμετρικά δεδομένα είναι να κατασκευάσουμε ένα κοινό πίνακα με την διαφορά ότι θα πρέπει να διαθέτει μια στήλη που να μπορεί να αποθηκεύει αυτά τα δεδομένα. Ο τύπος της στήλης που μας δίνει αυτή την δυνατότητα είναι της μορφής <geometry>.

Οπότε με την βοήθεια της γλώσσας ερωτημάτων :

- create table test (geo geometry)

Δημιουργήσαμε ένα πίνακα με όνομα 'test' και όνομα στήλης 'geo' που έχει την δυνατότητα αποθήκευσης χωρικών δεδομένων (geometry). Τώρα μπορούμε να δημιουργήσουμε και να εισάγουμε δεδομένα. Υπάρχουν αρκετές μέθοδοι να δημιουργήσουμε νέα δεδομένα και θα τις διαχωρίσουμε σε αυτές που δέχονται ως είσοδο την αναπαράσταση κειμένου και προαιρετικά την χρήση ενός συστήματος συντεταγμένων και σε αυτές που δέχονται ως είσοδο την αναπαράσταση σε δυαδική μορφή με προαιρετική και πάλι χρήση συστήματος συντεταγμένων. Από το σύνολο των μεθόδων αυτών που χρησιμοποιούν ως είσοδο την αναπαράσταση κειμένου η GeomFromText() είναι η πιο γενική και μπορούμε να την χρησιμοποιήσουμε για την

δημιουργία δεδομένων οποιουδήποτε τύπου ενώ όλες οι υπόλοιπες δημιουργούν μόνο δεδομένα του ίδιου τύπου στον οποίο αναφέρονται όπως φαίνεται στην συνέχεια.

- `GeomFromText()`
- `GeometryCollectionFromText(wkt[,srid])`
- `GeomFromText(wkt[,srid])`
- `LineStringFromText(wkt[,srid])`
- `MultiLineStringFromText(wkt[,srid])`
- `MultiPointFromText(wkt[,srid])`
- `MultiPolygonFromText(wkt[,srid])`
- `PointFromText(wkt[,srid])`
- `PolygonFromText(wkt[,srid])`

Θα δώσουμε δύο παραδείγματα δημιουργίας και αποθήκευσης δεδομένων στον πίνακα που δημιουργήσαμε, ένα με την γενική μέθοδο `GeomFromText()` και ένα με την μέθοδο που δημιουργεί αποκλειστικά δεδομένα τύπου γραμμής με την βοήθεια της γλώσσας ερωτημάτων.

```
Insert into Test Values (GeomFromText('Point(2 3)'));
```

```
/* Result : "Query OK, 1 rows affected (47 ms)" */
```

Δημιουργήσαμε και ταυτόχρονα αποθηκεύσαμε στον πίνακα `test` ένα γεωμετρικό σημείο με συντεταγμένες (2,3) χωρίς να προσδιορίσουμε κάποιο Σύστημα Συντεταγμένων. Το σύστημα μας επέστρεψε ότι το αποτέλεσμα ήταν επιτυχές, ότι το ερώτημα επηρέασε μια γραμμή του πίνακα μας και ότι ο χρόνος αντίδρασης ήταν 47ms.

```
Insert into Test Values (LineFromText('LineString(2 3,4 4)'));
```

```
/* Result : "Query OK, 1 rows affected (63 ms)" */
```


GeometryCollection(g1,g2)
LineString(pt1,pt2,...)
MultiLineString(ls1,ls2)
MultiPoint(pt1,pt2,...)
MultiPolygon(poly1,poly2)
Point(x,y)
Polygon(ls1,ls2)

Μέχρι τώρα έχουμε αναφερθεί στο πως δημιουργούμε ένα πίνακα έτοιμο να δεχθεί χωρικά δεδομένα και το σύνολο των συναρτήσεων που μπορούμε να χρησιμοποιήσουμε για να τα κατασκευάσουμε. Για να τα εισάγουμε στον πίνακα όπως είδαμε και παραπάνω με την βοήθεια της γλώσσας ερωτημάτων:

```
insert into test VALUES(GEOMFROMTEXT('POINT(1 1)'));
```

Με την παραπάνω εντολή αποθηκεύσαμε εσωτερικά στην βάση ένα σημείο με τετμημένη και τεταγμένη ένα. Η μορφή με την οποία αποθηκεύτηκε στη βάση δεν μπορεί να γίνει αντιληπτή. Για την μέθοδο GeomFromText καθώς και όλες τις μεθόδους θα αναφερθούμε αναλυτικά στην συνέχεια.

5.2 Δημιουργία και αποθήκευση Δεδομένων στην βάση MySQL

Όσον αφορά την PostgreSQL θα ξεκινήσουμε και πάλι με την διαδικασία δημιουργίας πίνακα και της κατάλληλης στήλης η οποία είναι ικανή να αποθηκεύσει χωρικά δεδομένα και στη συνέχεια θα αναφερθούμε στους τρόπους που μπορούμε να δημιουργήσουμε τα δεδομένα αυτά εσωτερικά στην βάση και να τα εισάγουμε στον πίνακα που δημιουργήσαμε.

```
CREATE TABLE geodata (  
geodata_id INTEGER,  
geodata_name VARCHAR  
);
```

Με το παραπάνω ερώτημα δημιουργούμε ένα πίνακα με όνομα geodata και προσθέτουμε δύο στήλες ικανές να δεχθούν η πρώτη ακέραιους αριθμούς και η δεύτερη αλφαριθμητικά.

```
SELECT AddGeometryColumn( 'geodata', 'geodata_obj', -1, 'GEOMETRY', 2 );
```

Με την παραπάνω εντολή προσθέσαμε στον πίνακα μας ακόμη μια στήλη με δυνατότητα αποθήκευσης χωρικών δεδομένων με όνομα geodata_obj χωρίς να ορίσουμε σύστημα συντεταγμένων (-1), τύπου Geometry που θα περιέχει δεδομένα με δύο διαστάσεις. Έτσι είμαστε σε θέση να δημιουργήσουμε τώρα τα δεδομένα και να τα εισάγουμε στον πίνακα.

Για την δημιουργία χωρικών δεδομένων το πακέτο PostGIS προσφέρει ένα σύνολο συναρτήσεων που ακολουθούν τις προδιαγραφές του διεθνούς οργανισμού OGC. Υπάρχουν γενικές συναρτήσεις που δημιουργούν δεδομένα οποιουδήποτε τύπου ανάλογα με το όρισμα που δίνουμε ως είσοδο και άλλες συναρτήσεις που δημιουργούν δεδομένα συγκεκριμένου τύπου όπως θα δούμε και στα παραδείγματα στην συνέχεια.

Αρχικά θα κάνουμε μια απλή αναφορά σε όσες από τις συναρτήσεις δέχονται ως όρισμα την αναπαράσταση κειμένου του αντικειμένου.

```
GeomFromText(text,[<srid>])
```

```
PointFromText(text,[<srid>])
```

```
LineFromText(text,[<srid>])
```

```
LinestringFromText(text,[<srid>])
```

```
SpolyFromText(text,[<srid>])
```

```
PolygonFromText(text,[<srid>])
```

```
MpointFromText(text,[<srid>])
```

MlineFromText(text,[<srid>])
MpolyFromText(text,[<srid>])
GeomCollFromText(text,[<srid>])

Αντίστοιχες με τις συναρτήσεις που δημιουργούν δεδομένα από την αναπαράσταση κειμένου υπάρχουν και για την δημιουργία δεδομένων δεχόμενες ως όρισμα την δυαδική αναπαράσταση.

GeomFromWKB(bytea,[<srid>])
GeometryFromWKB(bytea,[<srid>])
PointFromWKB(bytea,[<srid>])
LineFromWKB(bytea,[<srid>])
LinestringFromWKB(bytea,[<srid>])
PolyFromWKB(bytea,[<srid>])
PolygonFromWKB(bytea,[<srid>])
MPointFromWKB(bytea,[<srid>])
MLineFromWKB(bytea,[<srid>])
MPolyFromWKB(bytea,[<srid>])
GeomCollFromWKB(bytea,[<srid>])
BdPolyFromText(text WKT, integer SRID)
BdMPolyFromText(text WKT, integer SRID)

Έχουμε ήδη δημιουργήσει τον πίνακα geodata που είναι ικανός να δεχθεί τα δεδομένα μας, αναφερθήκαμε και στο σύνολο των συναρτήσεων που είναι ικανές να δημιουργήσουν τα δεδομένα αυτά οπότε θα δώσουμε και δύο παραδείγματα χρησιμοποιώντας δύο από τις συναρτήσεις που δημιουργούν δεδομένα με όρισμα την αναπαράσταση κειμένου για να γίνει πιο κατανοητή η διαδικασία.

```
INSERT INTO geodata (geodata_id, geodata_obj, geodata_name)
VALUES (1,GeomFromText('LINESTRING(1 1,2 2)',-1),'LineString');
```

Με την βοήθεια και πάλι της γλώσσας ερωτημάτων δημιουργούμε και εισάγουμε στον πίνακα geodata με την γενική συνάρτηση GeomFromText ένα ευθύγραμμο τμήμα μεταξύ των σημείων (1 1) και (2 2) δίνοντας ως ακέραιο τον αριθμό ένα και ονομάζοντας το νέο δεδομένο LineString.

```
INSERT INTO geodata (geodata_id, geodata_obj, geodata_name)
VALUES (2,LineFromText('LINESTRING(1 1,4 4)',-1),'LineString');
```

```
/* Result : "Query OK, 1 rows affected (47 ms)" */
```

Σε αυτό το παράδειγμα χρησιμοποιήσαμε την συνάρτηση που δημιουργεί γραμμές με όρισμα όπως είπαμε την αναπαράσταση σε μορφή κειμένου και προαιρετική εισαγωγή του συστήματος συντεταγμένων. Στο νέο δεδομένο δώσαμε τον αριθμό δύο και το ονομάσαμε και αυτό LineString.

5.3 Δημιουργία και αποθήκευση Δεδομένων στην βάση Oracle

Τέλος θα αναφερθούμε στην Oracle που παρουσιάζει αρκετές ιδιαιτερότητες σε σχέση με αυτά που είδαμε μέχρι στιγμής στα άλλα δύο λογισμικά. Η Oracle διαφέρει αρκετά από τον τρόπο που διαχειρίζονται τα δεδομένα τα άλλα λογισμικά. Καταρχήν για την αποθήκευση χωρικών δεδομένων θα πρέπει να δημιουργηθεί ένα πίνακας που να έχει μια στήλη του τύπου (SDO_GEOMETRY). Στην ουσία ο τύπος (SDO_GEOMETRY) αποτελεί ένα αντικείμενο το οποίο έχει συγκεκριμένες ιδιότητες.

1. SDO_GTYPE NUMBER
2. SDO_SRID NUMBER
3. SDO_POINT SDO_POINT_TYPE
4. SDO_ELEM_INFO SDO_ELEM_INFO_ARRAY
5. SDO_ORDINATES SDO_ORDINATE_ARRAY

Η πρώτη ιδιότητα αντιστοιχεί στον γεωμετρικό τύπο του αντικειμένου αποτελείται από ένα τετραψήφιο αριθμό εκ των οποίων το πρώτο ψηφίο δηλώνει τον αριθμό των διαστάσεων, το δεύτερο ψηφίο καθορίζει τις διαστάσεις του γραμμικού συστήματος συντεταγμένων όταν πρόκειται για τρισδιάστατους γεωμετρικούς τύπους και τα δύο τελευταία ψηφία δηλώνουν τον γεωμετρικό τύπο του αντικειμένου. Η δεύτερη ιδιότητα χρησιμοποιείται για να δηλώσει ένα σύστημα συντεταγμένων με το οποίο δηλώνεται το γεωμετρικό αντικείμενο. Αν η τιμή δεν οριστεί δεν υπάρχει κανένα σύστημα συντεταγμένων. Η Τρίτη ιδιότητα χρησιμοποιείται μόνο για σημειακά δεδομένα και σχετίζεται άμεσα με τις επόμενες δύο ιδιότητες. Αν η ιδιότητες 4,5 δεν έχουν οριστεί και η ιδιότητα 3 έχει τιμές σημαίνει ότι αυτές οι τιμές αποτελούν τις συντεταγμένες ενός σημείου διαφορετικά αν οι ιδιότητες 4,5 οριστούν αγνοείται η ιδιότητα 3.

Αρχικά δημιουργούμε και πάλι ένα πίνακα στον οποίο θα αποθηκεύσουμε τα χωρικά δεδομένα μας με δύο στήλες η πρώτη τύπου number για την αρίθμηση των δεδομένων και η δεύτερη τύπου SDO_GEOMETRY για την αποθήκευση των χωρικών δεδομένων μας.

```
Create table geodata (  
geodata_id number,  
geodata_obj SDO_GEOMETRY);
```

Στην συνέχεια ενημερώνουμε την όψη user_sdo_geom_metadata που αποθηκεύει πληροφορίες σχετικά με τις διαστάσεις των δεδομένων μας

```
insert into user_sdo_geom_metadata(  
table_name, column_name, diminfo, srid)  
values('geodata', 'geodata_obj', sdo_dim_array(  
sdo_dim_element('X', 0, 20, 0.1),  
sdo_dim_element('Y', 0, 20, 0.1)  
,NULL);
```

Και τέλος δημιουργούμε τον δείκτη

```
create index geodata_idx  
on geodata(geodata_obj)  
indextype is mdsys.spatial_index;
```

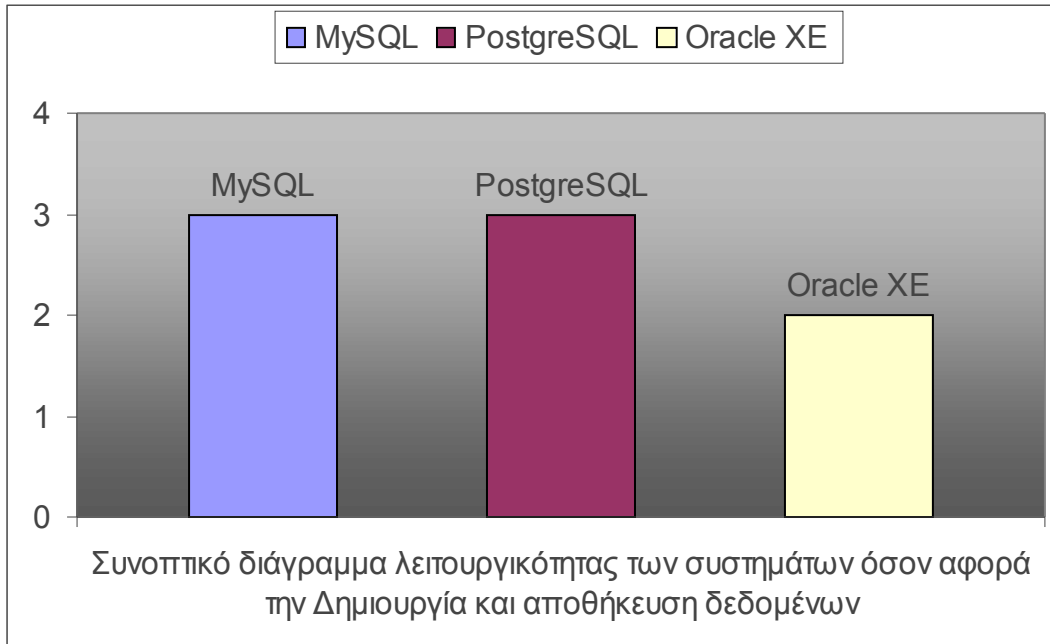
Τώρα μπορούμε να δημιουργήσουμε και να αποθηκεύσουμε τα δεδομένα μας
Οι συναρτήσεις για να δημιουργήσουμε δεδομένα με την Oracle αναφέρονται σε
παραδείγματα στη συνέχεια.

```
insert into geodata values(  
7,sdo_geometry(2003,null,null,sdo_elem_info_array(1,1003,1),  
sdo_ordinate_array(1,1, 5,1, 5,5, 1,5, 1,1)));
```

```
Insert into geodata values(  
8,sdo_geometry('POLYGON ((1.0 1.0, 5.0 1.0, 5.0 5.0, 1.0 5.0, 1.0 1.0))',null));
```

```
insert into geodata values(  
9,sdo_geometry('0x00: 504f4c59474f4e202828312e3020312e  
0x10: 302c20352e3020312e302c20352e3020  
0x20: 352e302c20312e3020352e302c20312e  
,null));  
POLYGON ((1.0 1.0, 5.0 1.0, 5.0 5.0, 1.0 5.0, 1.0 1.0))  
0x00: 504f4c59474f4e202828312e3020312e  
0x10: 302c20352e3020312e302c20352e3020  
0x20: 352e302c20312e3020352e302c20312e  
0x30: 3020312e302929
```

Είδαμε λοιπόν ότι τα δύο συστήματα (MySQL, PostgreSQL) υποστηρίζουν πληθώρα συναρτήσεων εύκολων στη χρήση για δημιουργία χωρικών δεδομένων σε αντίθεση με την Oracle που παρουσιάζει και κάποιες δυσκολίες στην χρήση της. Οπότε είναι προφανές ότι οι δύο πρώτες προσφέρουν περισσότερα στον χρήστη σε αυτόν το τομέα.



ΚΕΦΑΛΑΙΟ 6

Μορφές αναπαράστασης των χωρικών δεδομένων που υποστηρίζονται σε κάθε βάση

Σε αυτό το κεφάλαιο θα περιγράψουμε τις μορφές αναπαράστασης των χωρικών δεδομένων που υποστηρίζουν τα συστήματα διαχείρισης βάσεων δεδομένων. Σύμφωνα με τον διεθνή οργανισμό προτυποποίησης, όπως περιγράψαμε και σε προηγούμενο κεφάλαιο προτείνονται δύο βασικοί τρόποι αναπαράστασης :

- Well-Known Text (WKT) format
- Well-Known Binary (WBT) format

Είδαμε ότι με τον τρόπο αναπαράστασης σε μορφή κειμένου (WKT) ανταλλάσσονται δεδομένα σε μορφή ASCII, ενώ με την μορφή δυαδικής αναπαράστασης επιτυγχάνουμε ανταλλαγή δεδομένων με την βοήθεια δυαδικών ροών δεδομένων.

Για την μετατροπή χωρικών δεδομένων στις παραπάνω μορφές υπάρχουν οι αντίστοιχες μέθοδοι που παραθέτονται στη συνέχεια.

- ✓ AsText() Για μετατροπή σε μορφή κειμένου
- ✓ AsBinary() Για μετατροπή σε δυαδική μορφή

6.1 Μορφές αναπαράστασης δεδομένων στην βάση MySQL

Η MySQL υποστηρίζει και τις δύο μορφές αναπαράστασης καθώς και τις μεθόδους για την μετατροπή των δεδομένων από μια μορφή σε άλλη, ο τρόπος όμως που αποθηκεύει τα δεδομένα εσωτερικά στην βάση είναι διαφορετικός και από τις δύο μορφές. Με την μορφή που αποθηκεύονται εσωτερικά στη MySQL δεν μπορούμε να κατανοήσουμε τα δεδομένα οπότε χρησιμοποιούμε τις παραπάνω μεθόδους ώστε να τα μετατρέψουμε στην μορφή που θέλουμε. Ακολουθεί ένα παράδειγμα για την κάθε μία μορφή αναπαράστασης

δεδομένων. Οι μέθοδοι αυτοί και κυρίως η μετατροπή σε μορφή αναπαράστασης κειμένου είναι πάρα πολύ σημαντικές. Διότι ο χρήστης έχει άμεση εικόνα στα δεδομένα τα οποία χειρίζεται, διαφορετικά η χρήση της θα ήταν πολύ δυσκολότερη, για αυτό τον λόγο αποτελούν και ξεχωριστό κεφάλαιο.

```
select ASTEXT(geo) from test;
```

```
/* Result : "1 rows fetched (93 ms)" */
```

Αποτέλεσμα : POINT(1 1)

Το παραπάνω ερώτημα αναζητά τα γεωμετρικά δεδομένα που βρίσκονται στον πίνακα test και συγκεκριμένα στην στήλη geo και μας τα επιστρέφει σε μορφή κειμένου.

```
Select Asbinary(geo) from test;
```

```
/* Result : «1 rows fetched (141 ms)» */
```

```
0x00:  01 01 00 00 00 00 00 00 00 00 00 00 F0 3F 00 00 00
0x10:  00 00 00 F0 3F
```

Ενώ με την παραπάνω εντολή μετατρέπουμε και πάλι τα δεδομένα της στήλης geo του πίνακα test σε μορφή δυαδικής αναπαράστασης αυτή την φορά.

6.2 Μορφές αναπαράστασης δεδομένων στην βάση PostgreSQL

Την ίδια λειτουργικότητα σε αυτό τον τομέα εμφανίζει και η βάση PostgreSQL. Αποθηκεύει τα δεδομένα εσωτερικά με μία μορφή και υποστηρίζει την μετατροπή των γεωμετρικών δεδομένων στους δύο τύπους που ορίζονται από τον οργανισμό (OGC) κατά τον ίδιο τρόπο χρησιμοποιώντας τις εντολές AsText() και AsBinary(). Για την δημιουργία γεωμετρικών δεδομένων υπάρχουν πάρα πολλές συναρτήσεις που ακολουθούν τα πρότυπα του οργανισμού (OGC) ή αποτελούν παραλλαγές τους που δέχονται παραμέτρους σε μορφή κειμένου ή σε δυαδική μορφή. Ας δούμε και πάλι δύο παραδείγματα, ένα για κάθε μορφή να κατανοήσουμε την λειτουργία τους.

```
select astext(geodata_obj) from geodata;
/* Result : "2 rows fetched (78 ms)" */
```

Αποτέλεσμα : LINESTRING(1 1,2 2)
LINESTRING(1 1,4 4)

Με το ερώτημα αυτό ζητήσαμε να μας επιστρέψει σε μορφή κειμένου τα δεδομένα του πίνακα geodata.

```
select asbinary(geodata_obj) from geodata;
/* Result : "2 rows fetched (63 ms)" */
```

Αποτέλεσμα :

```
0x00: 01 02 00 00 00 02 00 00 00 00 00 00 00 00 00 F0
0x10: 3F 00 00 00 00 00 00 00 F0 3F 00 00 00 00 00 00
0x20: 40 00 00 00 00 00 00 00 40

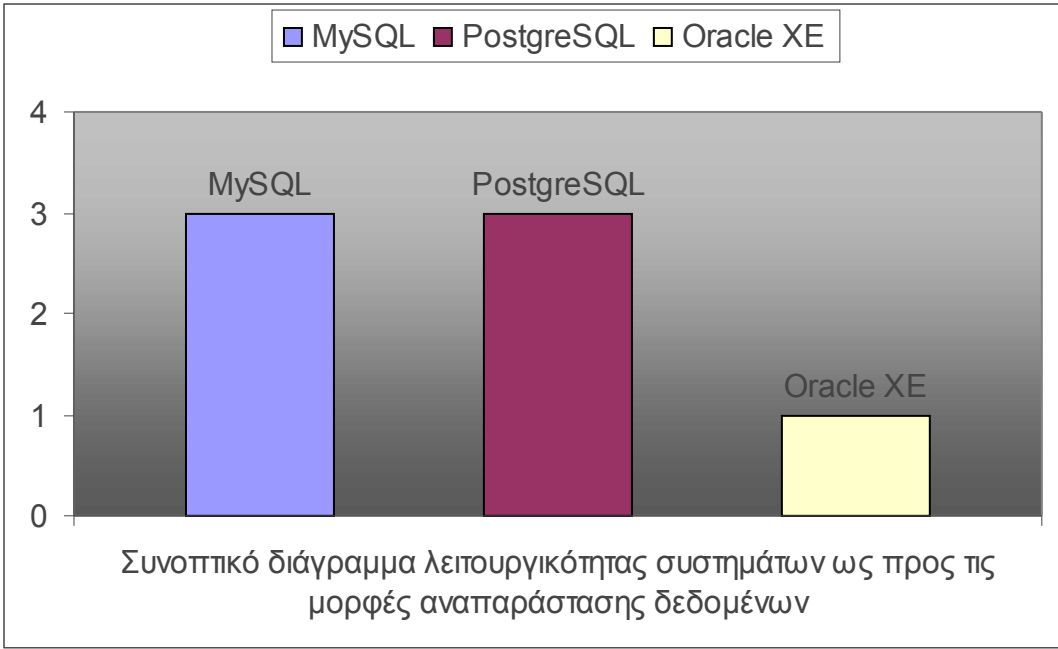
0x00: 01 02 00 00 00 02 00 00 00 00 00 00 00 00 00 F0
0x10: 3F 00 00 00 00 00 00 00 F0 3F 00 00 00 00 00 10
0x20: 40 00 00 00 00 00 00 00 10 40
```

Σε αυτή την περίπτωση μας επέστρεψε τα ίδια δεδομένα σε μορφή δυαδικής αναπαράστασης αυτή τη φορά.

6.3 Μορφές αναπαράστασης δεδομένων στην βάση Oracle

Όσον αφορά την Oracle είδαμε νωρίτερα ότι η μορφή που είναι αποθηκευμένα τα γεωμετρικά δεδομένα στο σύστημα δεν είναι κατανοητή. Είδαμε επίσης ότι δίνει την δυνατότητα αναπαράστασης σε μορφή κειμένου και σε δυαδική μορφή. Επίσης υποστηρίζει τις μεθόδους για μετατροπή από την μορφή που είναι αποθηκευμένα τα δεδομένα εσωτερικά σε μορφή κειμένου και σε δυαδική μορφή. Στην πράξη όμως όταν εκτελούμε το ερώτημα για την μετατροπή το σύστημα δεν επιστρέφει τίποτε χωρίς να υπάρχει κάποιο σφάλμα. Υπάρχει αδυναμία εκτέλεσης των μεθόδων στην πράξη. Αυτές οι μέθοδοι υποστηρίζονται μόνο στην πλήρη έκδοση του λογισμικού. Αυτό το γεγονός αποτελεί πολύ μεγάλο μειονέκτημα διότι είναι πολύ δύσκολο να χειριστεί κανείς δεδομένα αν δεν έχει άμεση εικόνα.

Οπότε αν δούμε συνολικά τα συστήματα η Oracle υστερεί πάρα πολύ σε σχέση με τα άλλα δύο συστήματα γιατί γίνεται πολύ δύσχρηστη.



ΚΕΦΑΛΑΙΟ 7

Συναρτήσεις

Είδαμε λοιπόν μέχρι στιγμής πως με την βοήθεια των λογισμικών μπορούμε να δημιουργήσουμε γεωμετρικά δεδομένα να τα αποθηκεύσουμε σε πίνακες και να τα μετατρέπουμε στην μορφή που θέλουμε. Το πιο σημαντικό πράγμα όμως πέραν των δυνατοτήτων που είδαμε είναι η ικανότητα ανάλυσης και επεξεργασίας των δεδομένων αυτών. Αυτό επιτυγχάνεται με την βοήθεια πληθώρας συναρτήσεων που μας προσφέρουν τα λογισμικά. Οι συναρτήσεις αυτές λόγω του μεγάλου αριθμού τους και του διαφορετικού τρόπου με τον οποίο επεξεργάζονται τα δεδομένα θα χωρισθούν σε διάφορες κατηγορίες για την καλύτερη κατανόηση και αξιολόγηση μεταξύ των τριών διαφορετικών συστημάτων βάσεων δεδομένων. Αρχικά θα γίνει μια γενική αναφορά στις κατηγορίες και στους τύπους των συναρτήσεων που ανήκουν σε κάθε μια κατηγορία και εν συνεχεία θα γίνει ξεχωριστή ανάλυση και σύγκριση μεταξύ των τριών συστημάτων για κάθε κατηγορία συναρτήσεων.

- **1^η κατηγορία** : Αποτελείται από συναρτήσεις που δέχονται σαν είσοδο ένα γεωμετρικό αντικείμενο και μας δίνουν πληροφορίες για τις ιδιότητες του
- **2^η κατηγορία** : Πρόκειται για συναρτήσεις που ελέγχουν την σχέση μεταξύ των γεωμετρικών αντικειμένων
- **3^η κατηγορία** : Σε αυτή την κατηγορία περιγράφονται οι συναρτήσεις που υποστηρίζουν χωρική ανάλυση

7.1 1^η Κατηγορία Συναρτήσεων

7.1.1 1^η Κατηγορία συναρτήσεων της βάσης MySQL

Ξεκινάμε πάλι με την βάση MySQL. Σε αυτή την κατηγορία όπως προαναφέραμε περιέχονται οι συναρτήσεις που μας δίνουν πληροφορίες σχετικά με τις ιδιότητες ενός γεωμετρικού αντικειμένου το οποίο δέχονται ως είσοδο. Για να γίνουν πιο κατανοητές Θα διαχωρίσουμε τις συναρτήσεις ακόμη περισσότερο σε αυτές που ισχύουν γενικά για πολλούς τύπους γεωμετρικών αντικειμένων και σε αυτές που ισχύουν αποκλειστικά σε ένα τύπο δεδομένων.

Για να γίνουν περισσότερο κατανοητές οι συναρτήσεις θα χρησιμοποιήσουμε πίνακα με συγκεκριμένα δεδομένα που θα είναι αντίστοιχα των συναρτήσεων που θα μελετάμε. Δημιουργούμε τον πίνακα με όνομα geom και όνομα στήλης g που περιέχει ένα δεδομένο τύπου LineString και ένα τύπου Polygon.

- **Dimension()** : Αυτή η συνάρτηση μας δίνει πληροφορίες για την διάσταση ενός γεωμετρικού αντικειμένου. Συγκεκριμένα επιστρέφει τιμή -1 όταν πρόκειται για κενή στήλη, 0 για αντικείμενο με μηδενικό μήκος και εμβαδόν, 1 για αντικείμενο με μηδενικό εμβαδόν και μη-μηδενικό μήκος και τέλος 2 για αντικείμενο με μη-μηδενικό εμβαδόν

```
select Dimension(g) from geom;
```

```
/* Result : "2 rows fetched (94 ms)" */
```

```
Αποτέλεσμα : 1
```

```
                2
```

Στο παραπάνω παράδειγμα βλέπουμε ότι η συνάρτηση μας επέστρεψε την τιμή 1 για την γραμμή διότι είναι δεδομένο με μη-μηδενικό μήκος και μηδενικό εμβαδόν, και την τιμή 2 για το πολύγωνο διότι έχει επιπλέον και μη-μηδενικό εμβαδόν.

- **Envelope()** : Επιστρέφει το ελάχιστο τετράγωνο που περικλείει το γεωμετρικό αντικείμενο

```

SELECT AsText(Envelope(g))
from geom;
/* Result : "2 rows fetched (109 ms)" */
Αποτέλεσμα : POLYGON((1 1,4 1,4 4,1 4,1 1))
                POLYGON((2 2,6 2,6 6,2 6,2 2))

```

Σε αυτό το παράδειγμα βλέπουμε ότι η συνάρτηση επέστρεψε δύο πολύγωνα σε μορφή αναπαράστασης κειμένου τα οποία δημιουργούνται με βάση τον τύπο :
 Polygon((MinX MinY, MaxX MinY,MaxX MaxY,MinX MaxY,MinX MinY)) για κάθε ένα από τα δεδομένα που περιέχονται στον πίνακα geom.

- **GeometryType()** : Επιστρέφει τον τύπο του γεωμετρικού αντικειμένου που δόθηκε σαν είσοδος

```

SELECT GeometryType(g) from geom;
/* Result : "2 rows fetched (109 ms)" */
Αποτέλεσμα : LINESTRING
                POLYGON

```

Εδώ η συνάρτηση μας δίνει τον τύπο καθενός από τα δεδομένα που είναι αποθηκευμένα στον πίνακα μας.

- **SRID()** : Επιστρέφει ένα ακέραιο που υποδεικνύει το σύστημα συντεταγμένων ως προς το οποίο αναφέρεται το γεωμετρικό αντικείμενο που δόθηκε ως είσοδος

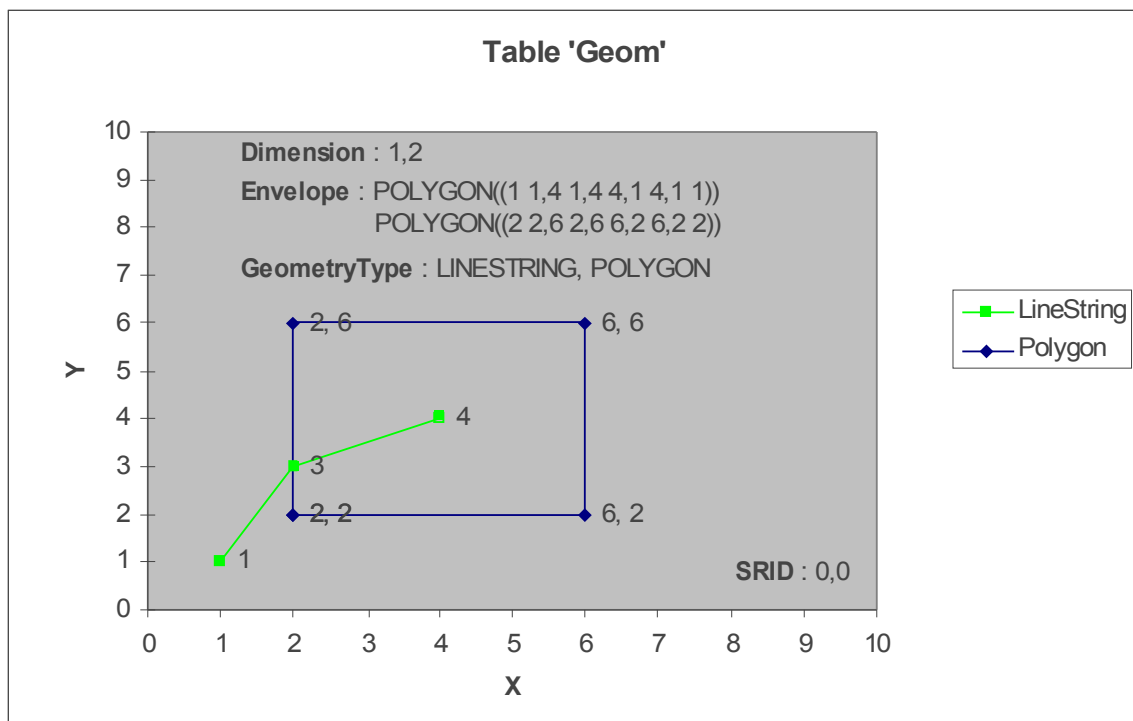
```

SELECT srid(g)
from geom;
/* Result : "2 rows fetched (78 ms)" */
Αποτέλεσμα : 0
                0

```

Τέλος στο παράδειγμα αυτό η συνάρτηση μας επιστρέφει τον ακέραιο αριθμό που αντιστοιχεί στον σύστημα συντεταγμένων που ανήκουν τα δεδομένα μας. Επειδή στον συγκεκριμένο πίνακα κατά την εισαγωγή των δεδομένων δεν δηλώθηκε σύστημα συντεταγμένων μας επέστρεψε μηδέν.

Για να γίνουν περισσότερο κατανοητές οι συναρτήσεις παραθέτουμε στην συνέχεια ένα διάγραμμα που απεικονίζει τα δεδομένα που περιέχει ο πίνακας geom, καθώς και τις συναρτήσεις που περιγράψαμε παραπάνω με το αποτέλεσμα που μας έδωσαν για τα δεδομένα του πίνακα μας.



Εδώ θα πρέπει να αναφέρουμε ότι οι παραπάνω συναρτήσεις είναι ένα υποσύνολο των γενικών συναρτήσεων που ορίζονται από τον διεθνή οργανισμό (OGC) και υποστηρίζονται από την MySQL.

Ειδικές Μέθοδοι των κλάσεων

Τώρα θα αναφερθούμε στις μεθόδους που ισχύουν σε κάθε μια από τις κατηγορίες γεωμετρικών δεδομένων.

Point

- X() Επιστρέφει την τετμημένη ενός σημείου
- Y() Επιστρέφει την τεταγμένη ενός σημείου

```
select y(p) from points;
```

```
/* Result : "3 rows fetched (110 ms)" */
```

```
Αποτέλεσμα : 1  
                2  
                5
```

Για δεδομένα τύπου σημείου βλέπουμε ότι μπορούμε να χρησιμοποιήσουμε τις δύο παραπάνω μεθόδους που μας επιστρέφουν είτε την τετμημένη είτε την τεταγμένη του εκάστοτε σημείου. Όπως φαίνεται και στο παράδειγμα, ζητάμε την τεταγμένη των σημείων που είναι αποθηκευμένα στην στήλη p του πίνακα points.

LineString - MultiLineString

Για την κατηγορία αυτή θα χρησιμοποιήσουμε ένα πίνακα με όνομα line στον οποίο έχουμε αποθηκεύσει ένα γεωμετρικό δεδομένο τύπου LineString και όνομα στήλης l ώστε να δείξουμε και γραφικά τα αποτελέσματα των ερωτημάτων SQL. Εισάγουμε στον πίνακα την γραμμή LineString(1 1,2 2,4 4). Για κάθε μέθοδο θα δίνουμε και αντίστοιχο παράδειγμα πάντα με βάση τον πίνακα line. Επίσης πάντοτε όταν ζητάμε να επιστραφεί κάποιο γεωμετρικό δεδομένο χρησιμοποιούμε την μέθοδο AsText για να μας τα επιστρέφει σε μορφή αναπαράστασης κειμένου.

- StartPoint() Επιστρέφει το πρώτο σημείο του γεωμετρικού αντικειμένου
SELECT AsText(StartPoint(l))

from line;

/* Result : «1 rows fetched (78 ms)” */

Αποτέλεσμα : POINT(1 1)

- EndPoint() Επιστρέφει το τελευταίο σημείο ενός γεωμετρικού αντικειμένου αυτού του τύπου

SELECT AsText(EndPoint(l))

from line;

/* Result : “1 rows fetched (63 ms)” */

Αποτέλεσμα : POINT(4 4)

- Glength() Επιστρέφει το συνολικό μήκος μιας γραμμής

SELECT glength(l)

from line;

/* Result : “1 rows fetched (94 ms)” */

Αποτέλεσμα: 4.2426406871193

- NumPoints() Επιστρέφει τον συνολικό αριθμό των σημείων του της γραμμής

SELECT NumPoints(l)

from line;

/* Result : “1 rows fetched (78 ms)” */

Αποτέλεσμα : 3

- PointN() Επιστρέφει το νιοστό σημείο του αντικειμένου ξεκινώντας την αρίθμηση από το ένα.

SELECT ASTEXT(PointN(l,2))

from line;

/* Result : “1 rows fetched (78 ms)” */

Αποτέλεσμα : POINT(2 2)

Στην παράδειγμα αυτό ζητάμε μέσα στην παρένθεση το δεύτερο σημείο της στήλης I.

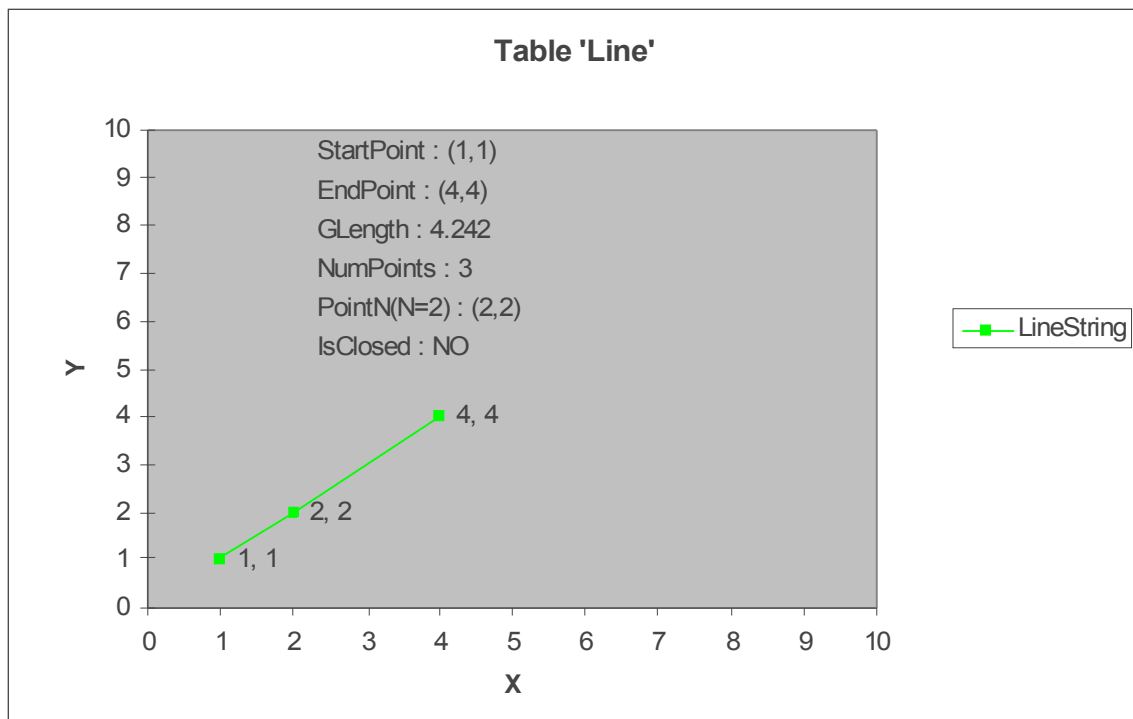
- IsClosed() Επιστρέφει την τιμή 1 εάν όλα τα αντικείμενα αυτού του τύπου που περιέχονται στην στήλη του πίνακα είναι κλειστά, δηλαδή το αρχικό τους σημείο συμπίπτει με το τελικό

```
SELECT isclosed(I)
```

```
from line;
```

```
/* Result : "1 rows fetched (94 ms)" */
```

Αποτέλεσμα : 0



Στον παραπάνω πίνακα εμφανίζονται όλες οι συναρτήσεις και τα αποτελέσματα των ερωτημάτων σε κάθε παράδειγμα καθώς και τα δεδομένα του πίνακα line που στην συγκεκριμένη περίπτωση αποτελούνταν μόνο από μια γραμμή.

Polygon - MultiPolygon

Για τις συναρτήσεις αυτές θα δημιουργήσουμε ένα νέο πίνακα με όνομα Polygons και γεωμετρική στήλη με όνομα p που θα περιέχει ένα δεδομένο Polygon(1 1,5 1,5 5,1 5,1 1),(2 2,4 2,4 4,2 4,2 2). Το πολύγωνο αυτό αποτελείται από ένα εξωτερικό και ένα εσωτερικό δακτύλιο. Κάθε συνάρτηση συνοδεύεται και από αντίστοιχο παράδειγμα πάντα σε σχέση με τα δεδομένα του πίνακα polygons.

- Area()

Επιστρέφει το εμβαδό του πολυγώνου

```
SELECT Area(p)
```

```
from polygons;
```

```
/* Result : “1 rows fetched (94 ms)” */
```

Αποτέλεσμα : 12

Βλέπουμε λοιπόν ότι το αποτέλεσμα της συνάρτησης μας δίνει τον αριθμό δώδεκα που όπως φαίνεται και στο συγκεντρωτικό διάγραμμα παρακάτω αποτελεί την διαφορά του εμβαδού του εξωτερικού δακτυλίου από τον εσωτερικό δακτύλιο

- ExteriorRing()

Επιστρέφει τον εξωτερικό δακτύλιο του πολυγώνου που δέχεται ως είσοδο. Ο τύπος του αντικειμένου που επιστρέφει η συνάρτηση αυτή είναι μια κλειστή γραμμή που ταυτίζεται με τον εξωτερικό δακτύλιο του πολυγώνου μας.

```
SELECT astext(ExteriorRing(p))
```

```
from polygons;
```

```
/* Result : “1 rows fetched (78 ms)” */
```

Αποτέλεσμα : LINESTRING(1 1,5 1,5 5,1 5,1 1)

- InteriorRingN()

Επιστρέφει τον νιοστό εσωτερικό δακτύλιο του πολυγώνου σε μορφή LineString διότι σε ένα πολύγωνο μπορεί να έχουμε περισσότερους από ένα εσωτερικούς δακτυλίους. Η αρίθμηση των δακτυλίων ξεκινά από τον αριθμό ένα

```
SELECT astext(InteriorRingN(p,1))
```

```
from polygons;
```

```
/* Result : “1 rows fetched (109 ms)” */
```

```
Αποτέλεσμα : LINESTRING(2 2,4 2,4 4,2 4,2 2)
```

Σε αυτό το παράδειγμα επειδή το πολύγωνο μας διαθέτει μόνο ένα εσωτερικό δακτύλιο δώσαμε ως όρισμα τον αριθμό ένα για να μας επιστρέψει τον δακτύλιο που όπως και προηγουμένως έχει την μορφή γραμμής.

- NumInteriorRings()

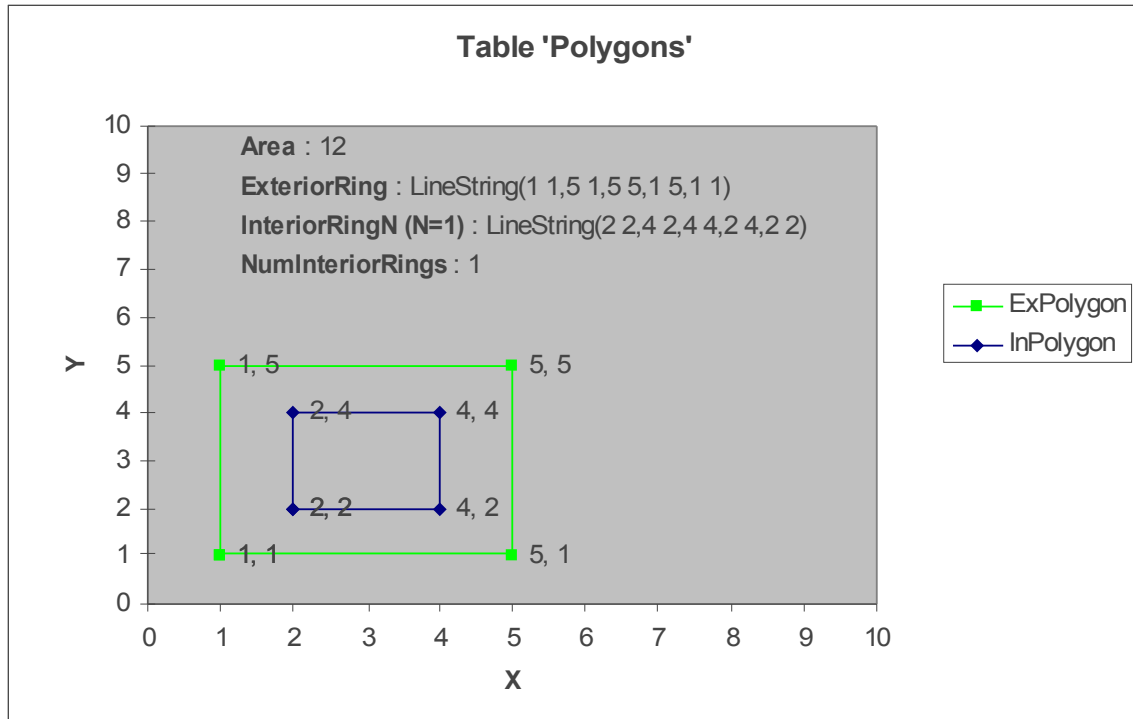
Επιστρέφει τον συνολικό αριθμό των εσωτερικών δακτυλίων του πολυγώνου

```
SELECT NumInteriorRings(p)
```

```
from polygons;
```

```
/* Result : “1 rows fetched (78 ms)” */
```

```
Αποτέλεσμα : 1
```



Συνοπτικά φαίνονται στο παραπάνω διάγραμμα το πολύγωνο που ήταν αποθηκευμένο στον πίνακα Polygons και οι συναρτήσεις που μελετήσαμε με τα αποτελέσματα που μας έδωσε η κάθε μια για τα δεδομένα του πίνακα μας.

GeometryCollection

Δημιουργούμε ένα πίνακα με όνομα GeometryCollections με όνομα γεωμετρικής στήλης gc και εισάγουμε ένα δεδομένο αυτού του τύπου που περιέχει ένα σημείο και μία γραμμή
 GeometryCollection(Point(2 3),LineString(1 1,2 2,3 3))

- GeometryN(gc,N)

Επιστρέφει το νιοστό αντικείμενο που είναι αποθηκευμένο στην στήλη gc.

SELECT astext(GeometryN(gc,1))

from geometrycollections;

/* Result : "1 rows fetched (79 ms)" */

Αποτέλεσμα : Point(2 3)

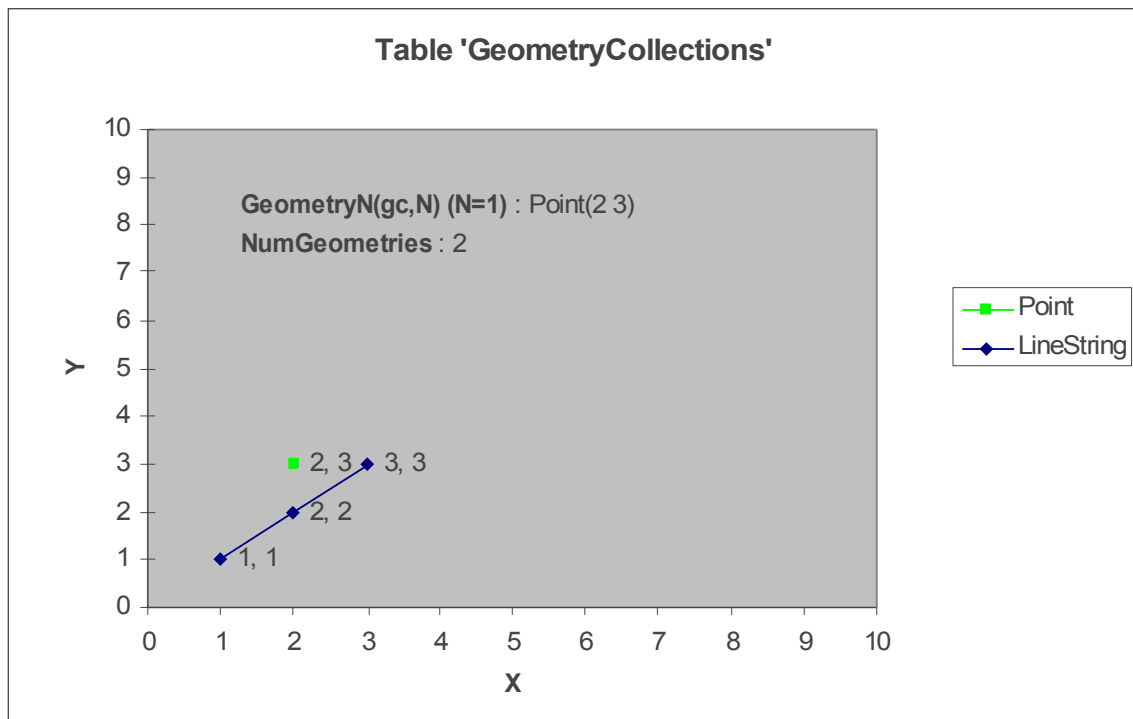
Στο παραπάνω παράδειγμα ζητάμε να μας επιστρέψει η συνάρτηση τον πρώτο γεωμετρικό τύπο του αντικειμένου που βρίσκεται στον πίνακα `geometrycollections` και για να γίνει κατανοητός τον μετατρέπουμε ταυτόχρονα σε μορφή κειμένου.

- `NumGeometries()` Επιστρέφει τον αριθμό των επί μέρους αντικειμένων που περιέχονται στον γεωμετρικό τύπο.

```
SELECT NumGeometries(gc)  
from geometrycollections;  
/* Result : "1 rows fetched (78 ms)" */
```

Αποτέλεσμα : 2

Όπως προείπαμε το αντικείμενο που βρίσκεται στον πίνακα μας αποτελείται από δύο δεδομένα για αυτό και η συνάρτηση επιστρέφει τον αριθμό δύο.



Στο διάγραμμα απεικονίζονται τα δεδομένα που περιέχονται στον πίνακα μας καθώς και οι συναρτήσεις και τα αντίστοιχα αποτελέσματα που είδαμε στα παραδείγματα για κάθε μία από αυτές.

7.1.2 1^η Κατηγορία συναρτήσεων της βάσης PostgreSQL

Συνεχίζουμε με τις συναρτήσεις της PostgreSQL που μας δίνουν πληροφορίες για τα γεωμετρικά δεδομένα. Όσες από τις συναρτήσεις υποστηρίζονται και από την MySQL και της μελετήσαμε παραπάνω θα τις αναφέρουμε επιγραμματικά και θα δοθούν παραδείγματα μόνο σε αυτές που θα συναντήσουμε για πρώτη φορά ώστε να σχηματίσουμε μια εικόνα από όλες τις συναρτήσεις.

- SRID(geometry)
- Dimension(geometry)
- Envelope(geometry)
- GeometryType(geometry)

Δημιουργούμε ένα νέο πίνακα Geomdata με όνομα στήλης που δέχεται χωρικά δεδομένα geomdata_obj η οποία περιέχει ένα δεδομένο τύπου LineString και ένα τύπου Polygon

- Boundary(geometry)

Η συνάρτηση αυτή όταν πρόκειται για LineStrings ή MultiLineStrings μας επιστρέφει τα διαδοχικά σημεία των γραμμών και όταν πρόκειται για δεδομένα τύπου Polygon ή MultiPolygon μας επιστρέφει μία γραμμή που αποτελείται από τμήματα που ισοδυναμούν με τις πλευρές του πολυγώνου

```
SELECT astext(boundary(geomdata_obj)) from geomdata;
```

```
/* Result : "2 rows fetched (719 ms)" */
```

```
Αποτέλεσμα : MULTIPOINT(1 2,2 4)
```

LINestring(1 1,5 1,5 5,1 5,1 1)

Να παρατηρήσουμε σε αυτό το σημείο ότι οποτεδήποτε περιμένουμε να μας επιστρέψει μια συνάρτηση δεδομένα, πάντα χρησιμοποιούμε την συνάρτηση astext για να γίνονται κατανοητά

- IsEmpty(geometry)

Η συνάρτηση IsEmpty επιστρέφει True ή False εάν σε ένα πίνακα υπάρχουν εγγραφές οι οποίες δεν περιέχουν δεδομένα.

```
SELECT IsEmpty(geomdata_obj) from geomdata;
```

```
/* Result : «2 rows fetched (31 ms)» */
```

Αποτέλεσμα : False

False

Στον συγκεκριμένο πίνακα υπάρχουν δύο μόνο εγγραφές όπως είπαμε που περιέχουν έγκυρα χωρικά δεδομένα για αυτό και επιστρέφει false.

- IsSimple(geometry)

Επιστρέφει False εάν κάποιο αντικείμενο περιέχει σημεία τα οποία διασταυρώνονται μεταξύ τους αλλιώς επιστρέφει True.

```
SELECT issimple(geomdata_obj) from geomdata;
```

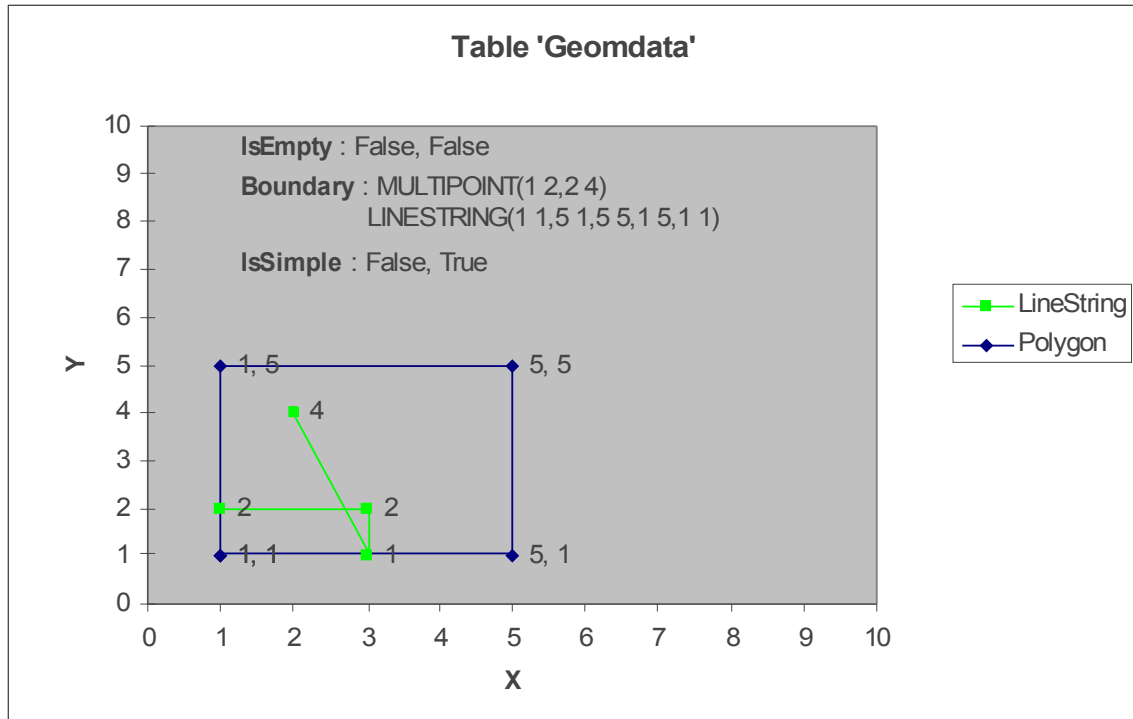
```
/* Result : “2 rows fetched (62 ms)” */
```

Αποτέλεσμα : False

True

Εδώ η συνάρτηση μας επέστρεψε True στο δεύτερο αντικείμενο διότι το αρχικό και τελικό σημείο του πολυγώνου συμπίπτουν.

Παρακάτω φαίνονται συνοπτικά στο διάγραμμα οι συναρτήσεις και τα αποτελέσματα που μας επέστρεψαν.



Μέχρι στιγμής είδαμε τις γενικές συναρτήσεις, τώρα θα αναφερθούμε στις συναρτήσεις που ισχύουν για κάθε ένα τύπο δεδομένων ξεχωριστά.

LineString - MultiLineString

Επιγραμματικά αναφέρουμε όσες από τις συναρτήσεις συναντήσαμε στην MySQL και δόθηκε αντίστοιχο παράδειγμα

- Length(geometry)
Αυτή είναι η αντίστοιχη συνάρτηση της Glength που συναντήσαμε στην MySQL
- IsClosed(geometry)
- NumPoints(geometry)
- EndPoint(geometry)
- StartPoint(geometry)

- PointN(geometry,integer)

- IsRing(geometry)

Η συνάρτηση αυτή αφορά αποκλειστικά γραμμές και δεν ισχύει για πολλαπλές γραμμές

```
SELECT isRing(geomdata_obj)
```

```
from geomdata
```

```
where geomdata_id=1;
```

```
/* Result : "1 rows fetched (63 ms)" */
```

Αποτέλεσμα : False

Επειδή η συνάρτηση ισχύει μόνο για γραμμές και ο πίνακας μας περιέχει και ένα δεδομένο τύπου πολύγωνο, αναζητάμε στον πίνακα μόνο αυτό με αύξων αριθμό ένα που αντιστοιχεί στην γραμμή.

Polygon - MultiPolygon

Προχωράμε στα απλά και σύνθετα πολύγωνα. Για να αποφύγουμε επαναλήψεις ίδιων συναρτήσεων απλώς αναφέρονται επιγραμματικά και δίνουμε παραδείγματα σε όσες συναντώνται για πρώτη φορά.

- Area(geometry)
- ExteriorRing(geometry)
- NumInteriorRing (geometry)
- InteriorRingN(geometry,integer)

Οι συναρτήσεις ExteriorRing και InteriorRingN αφορούν αποκλειστικά γεωμετρικά δεδομένα τύπου Polygon και όχι για πολλαπλά πολύγωνα.

- **Centroid(geometry)**

```
SELECT astext(centroid(geom_obj)) from geom;
```

```
/* Result : "2 rows fetched (31 ms)" */
```

```
Αποτέλεσμα : Point(1.5 1.5)
```

```
Point(2 2)
```

Αυτή η συνάρτηση μας επιστρέφει ένα σημείο που αντιστοιχεί στο κέντρο κάθε γεωμετρικού σχήματος

GeometryCollection

Χρησιμοποιείτε όταν πρόκειται για πολλαπλά αντικείμενα διαφορετικών τύπων μεταξύ τους.

- **NumGeometries (geometry)**

Επιστρέφει το σύνολο των γεωμετρικών αντικειμένων που περιέχει

- **GeometryN (geometry,Int)**

Επιστρέφει το νιοστό γεωμετρικό αντικείμενο που ορίζεται από τον ακέραιο που δόθηκε ως όρισμα στην συνάρτηση

7.1.3 1^η Κατηγορία συναρτήσεων της βάσης Oracle

Προχωράμε στις συναρτήσεις που υποστηρίζει η Oracle. Αρχικά πρέπει και πάλι να δημιουργήσουμε τον πίνακα και να εισάγουμε τα κατάλληλα δεδομένα που θα βοηθήσουν στην κατανόηση των συναρτήσεων αυτού του τύπου.

Δημιουργούμε ένα πίνακα geom2 με γεωμετρική στήλη geom2_obj και εισάγουμε ένα δεδομένο τύπου LineString(1 1,2 2) και ένα τύπου Polygon(1 1,3 1,3 3,1 3,1 1)

- **CoordDim**

Επιστρέφει τις διαστάσεις των γεωμετρικών αντικειμένων όπως και η **Get_Dims** που παραθέτουμε στην συνέχεια και δίνεται αντίστοιχο παράδειγμα.

- **Get_Dims**

Επιστρέφει τον αριθμό των διαστάσεων του γεωμετρικού αντικειμένου που έχει οριστεί στην παράμετρο SDO_GTYPE

```
SELECT g.geom2_obj.Get_Dims()  
FROM geom2 g;  
/* Result : "2 rows fetched (313 ms)" */
```

Αποτέλεσμα : 2

2

- **Get_Gtype**

Επιστρέφει ένα αριθμό που αντιστοιχεί στον τύπο του γεωμετρικού αντικειμένου που έχει οριστεί στην παράμετρο SDO_GTYPE

```
SELECT g.geom2_obj.Get_Gtype()  
FROM geom2 g;  
/* Result : "2 rows fetched (78 ms)" */
```

Αποτέλεσμα : 3

1

Get_LRS_Dim

```
SELECT g.geom2_obj.Get_LRS_Dim()  
FROM geom2 g;  
/* Result : "2 rows fetched (78 ms)" */
```

Αποτέλεσμα : 0

0

- **ST_IsValid**

Επιστρέφει μηδέν εάν το γεωμετρικό αντικείμενο είναι μη-έγκυρο και ένα εάν είναι έγκυρο

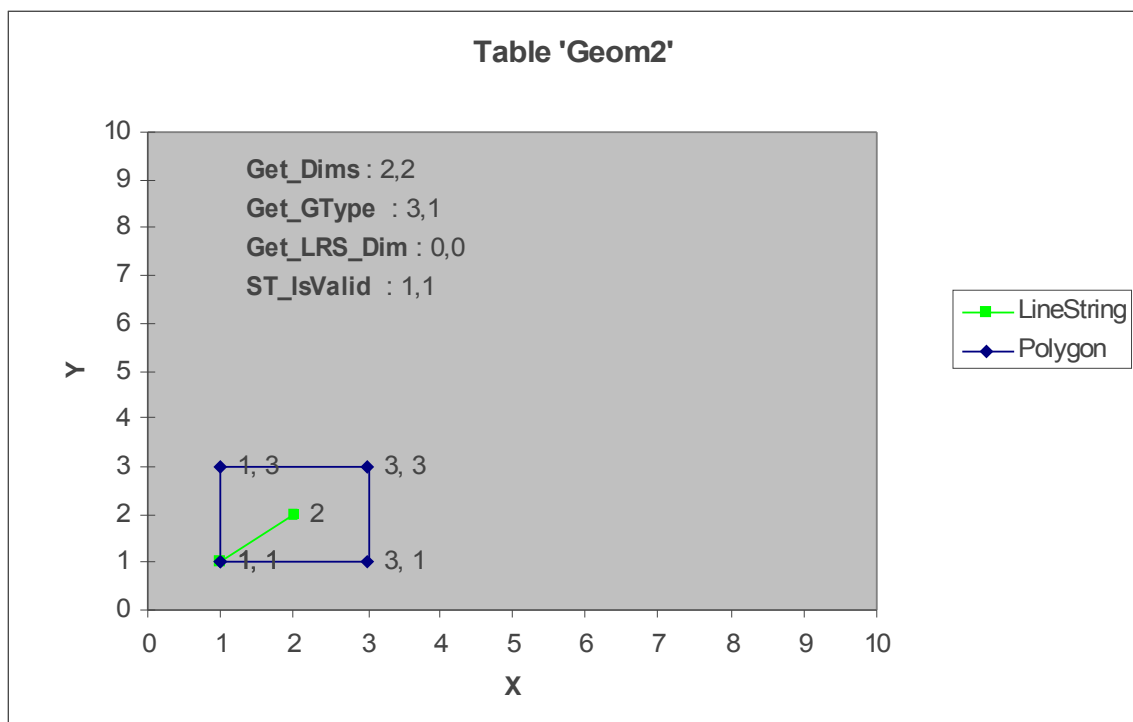
```
SELECT g.geom2_obj.ST_IsValid()
```

```
FROM geom2 g;
```

```
/* Result : "2 rows fetched (531 ms)" */
```

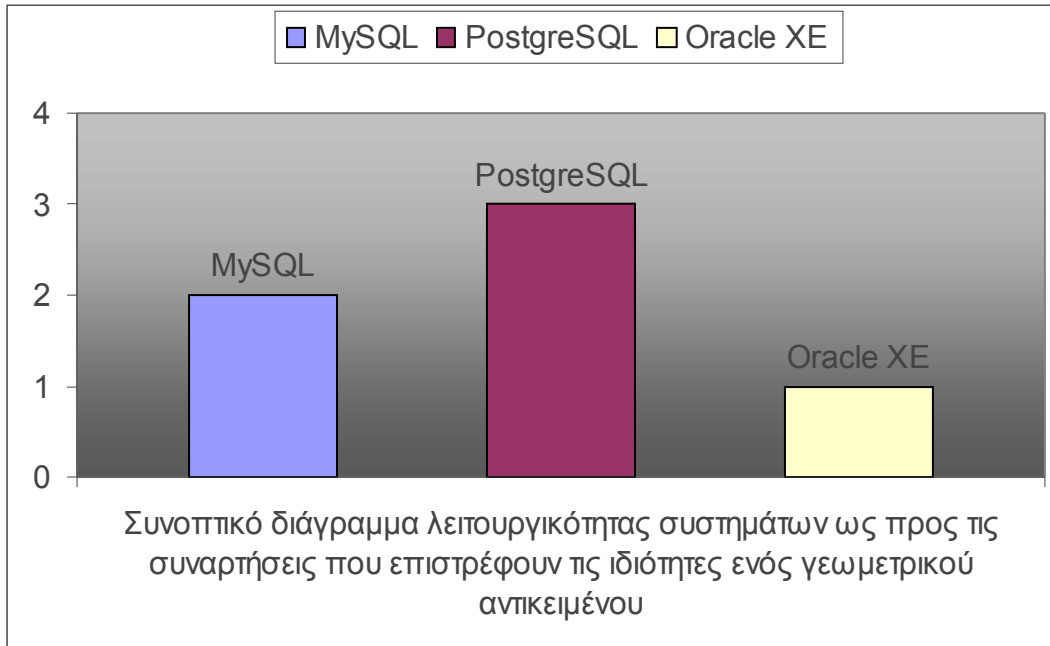
Αποτέλεσμα : 1

1



Εξετάσαμε σε αυτό το κεφάλαιο τις συναρτήσεις του κάθε ενός από τα υπό μελέτη συστήματα που μας επιστρέφουν τις ιδιότητες του αντικειμένου που δέχθηκαν ως είσοδο. Είδαμε λοιπόν ξεκινώντας από την βάση MySQL ότι υποστηρίζει αρκετές συναρτήσεις που ισχύουν είτε σε για ένα τύπο γεωμετρικών δεδομένων είτε γενικά για όλους. Η MySQL ακολουθεί και εδώ την προτυποποίηση του οργανισμού OGC και υλοποιεί μεγάλο μέρος των συναρτήσεων αυτού του τύπου. Προχωρώντας ένα βήμα παραπάνω η βάση PostgreSQL που και αυτή ακολουθεί την προτυποποίηση του οργανισμού υλοποιεί στο σύνολο τους τις συναρτήσεις που προτείνονται από τον οργανισμό. Τέλος η Oracle

είδαμε ότι έχει την δυνατότητα να εκτελέσει 4-5 πολύ βασικές σε σχέση με τα άλλα δύο λογισμικά οπότε υστερεί σημαντικά σε αυτό τον τομέα. Για να μπορέσουμε να έχουμε μια γενικότερη εικόνα θα παρουσιάσουμε ένα συγκριτικό διάγραμμα σε κάθε επί μέρους κεφάλαιο ώστε στο τέλος να είναι διακριτές με ευκολία οι διαφορές των συστημάτων.



7.2 2^η Κατηγορία Συναρτήσεων

Σε αυτό το κεφάλαιο θα μελετήσουμε τις συναρτήσεις που ελέγχουν την σχέση μεταξύ των γεωμετρικών αντικειμένων.

7.2.1 2^η Κατηγορία συναρτήσεων της βάσης MySQL

Η MySQL υποστηρίζει αρκετές συναρτήσεις που ελέγχουν σχέσεις μεταξύ δύο γεωμετρικών αντικειμένων, ο έλεγχος όμως που πραγματοποιεί σχετίζεται με τα ελάχιστα τετράγωνα των αντικειμένων και όχι με τα ίδια τα αντικείμενα. Έτσι τα αποτελέσματα των συναρτήσεων αυτών δεν μας δίνουν την ακριβή εικόνα που πρέπει να έχουμε κατά την επεξεργασία γεωμετρικών δεδομένων. Για κάθε συνάρτηση που θα μελετήσουμε θα δοθεί και αντίστοιχο παράδειγμα με συνοπτικό διάγραμμα στο τέλος για να έχουμε καλύτερη εικόνα των αποτελεσμάτων.

- **MBRContains(g1,g2)**

Επιστρέφει 1 ή 0 εάν το ελάχιστο τετράγωνο του αντικειμένου g1 περιέχει το ελάχιστο τετράγωνο του αντικειμένου g2 ή όχι.

```
SET @g1 = GeomFromText('Polygon((1 1,6 1,6 6,1 6,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 2,3 2,5 5)');
```

```
SELECT MBRCONTAINS(@g1,@g2);
```

```
/* Result : "1 rows fetched (32 ms)" */
```

Αποτέλεσμα : 1

Στο παραπάνω διάγραμμα χρησιμοποιούμε για πρώτη φορά δύο αντικείμενα που δεν είναι αποθηκευμένα σε κάποιο πίνακα, απλώς είναι αντιστοιχισμένα σε μεταβλητές και τις καλούμε στην συνάρτηση ώστε να εξετάσουμε εάν το ελάχιστο τετράγωνο του πρώτου περιέχει το ελάχιστο τετράγωνο του δεύτερου αντικειμένου.

- **MBRDisjoint(g1,g2)**

Επιστρέφει 1 εάν τα ελάχιστα τετράγωνα των δύο γεωμετρικών αντικειμένων δεν έχουν κανένα κοινό σημείο και 0 εάν έχουν κοινό σημείο η ακόμη εάν εφάπτονται

```
SET @g1 = GeomFromText('Polygon((1 1,6 1,6 6,1 6,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 2,3 2,5 5)');
```

```
SELECT MBRDISJOINT(@g1,@g2);
```

```
1 rows fetched (31 ms)
```

Αποτέλεσμα : 0

- **MBRWithin(g1,g2)**

Επιστρέφει 1 εάν το ελάχιστο τετράγωνο του πρώτου γεωμετρικού αντικειμένου περιέχεται στο ελάχιστο τετράγωνο του δεύτερου αντικειμένου διαφορετικά επιστρέφει 0

```
SET @g1 = GeomFromText('Polygon((1 1,6 1,6 6,1 6,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 2,3 2,5 5)');
```

```
SELECT MBRWithin(@g2,@g1);
```

```
/* Result : «1 rows fetched (78 ms)» */
```

Αποτέλεσμα : 1

- **MBREqual(g1,g2)**

Επιστρέφει 1 εάν τα ελάχιστα τετράγωνα των δύο αντικειμένων είναι ίδια και 0 εάν είναι διαφορετικά

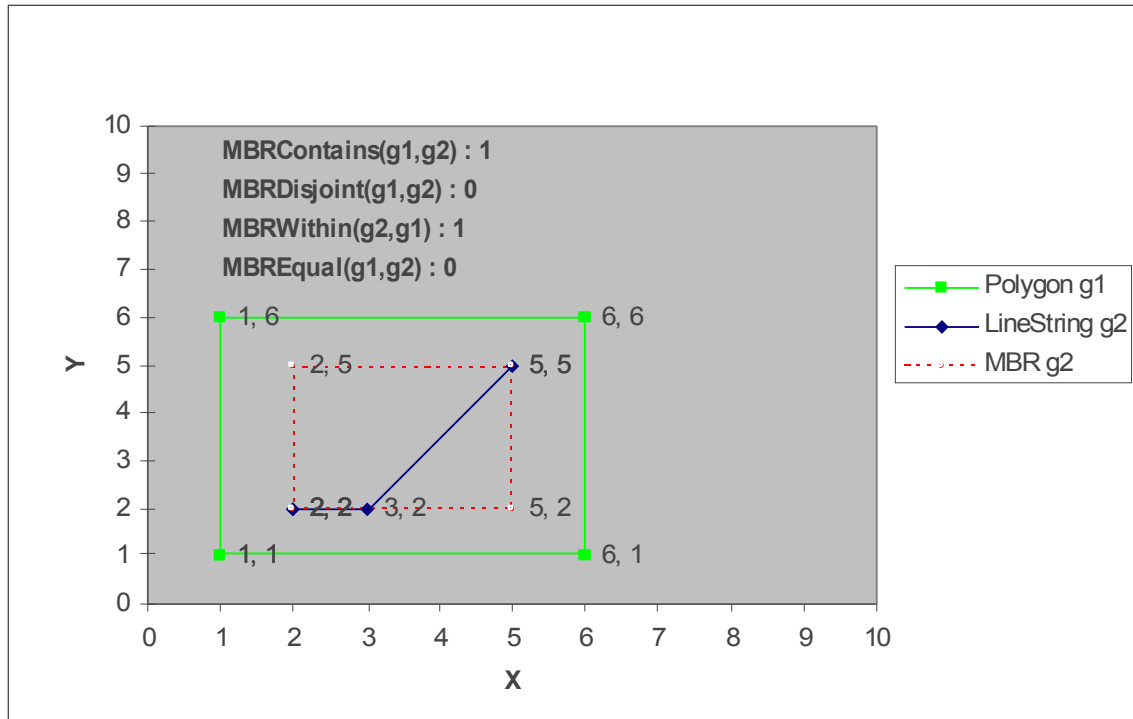
```
SET @g1 = GeomFromText('Polygon((1 1,6 1,6 6,1 6,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 2,2 3,5 5)');
```

```
SELECT MBREQUAL(@g1,@g2);
```

```
/* Result : “1 rows fetched (31 ms)” */
```

Αποτέλεσμα : 0



Βλέπουμε λοιπόν στο παραπάνω διάγραμμα τα δύο αντικείμενα μας και με κόκκινη διακεκομμένη γραμμή το ελάχιστο τετράγωνο της γραμμής. Το ελάχιστο τετράγωνο του πολυγώνου είναι το ίδιο το πολύγωνο διότι είναι παραλληλόγραμμο. Οπότε στην πρώτη συνάρτηση μας επιστρέφει ένα διότι το ελάχιστο τετράγωνο του πολυγώνου περιέχει το ελάχιστο τετράγωνο της γραμμής, στη δεύτερη συνάρτηση επιστρέφει μηδέν γιατί δεν υπάρχει κοινό σημείο, στην τρίτη μας επιστρέφει ένα διότι το τετράγωνο της γραμμής περιέχεται σε αυτό του πολυγώνου και τέλος επειδή όπως είναι προφανές δεν ταυτίζονται μας επιστρέφει μηδέν.

- **MBRIntersects(g1,g2)**

Επιστρέφει 1 εάν τα ελάχιστα τετράγωνα των αντικειμένων διασταυρώνονται και 0 εάν όχι

```
SET @g1 = GeomFromText('Polygon((1 1,4 1,4 4,1 4,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 1,3 3,5 3)');
```

```
SELECT MBRINTERSECTS(@g1,@g2);
```

```
/* Result : «1 rows fetched (62 ms)» */
```

Αποτέλεσμα : 1

- **MBROverlaps(g1,g2)**

Επιστρέφει ένα εάν τα ελάχιστα τετράγωνα των αντικειμένων επικαλύπτονται και μηδέν εάν όχι. Επικάλυψη όταν πρόκειται για γεωμετρικά αντικείμενα σημαίνει ότι δύο αντικείμενα διασταυρώνονται και αυτό έχει ως αποτέλεσμα ένα νέο γεωμετρικό αντικείμενο που να μην είναι ίσο με κάποιο από τα προηγούμενα αλλά με τις ίδιες διαστάσεις αλλιώς επιστρέφει μηδέν.

```
SET @g1 = GeomFromText('Polygon((1 1,4 1,4 4,1 4,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 1,3 3,5 3)');
```

```
SELECT MBROVERLAPS(@g1,@g2);
```

```
/* Result : «1 rows fetched (47 ms)» */
```

Αποτέλεσμα : 1

- **MBRTouches(g1,g2)**

Επιστρέφει ένα εάν τα ελάχιστα τετράγωνα εφάπτονται. Δηλαδή θα πρέπει τα εσωτερικά των αντικειμένων να μην διασταυρώνονται αλλά το όριο του ενός να διασταυρώνεται είτε με το όριο του άλλου είτε με το εσωτερικό του διαφορετικά επιστρέφει 0

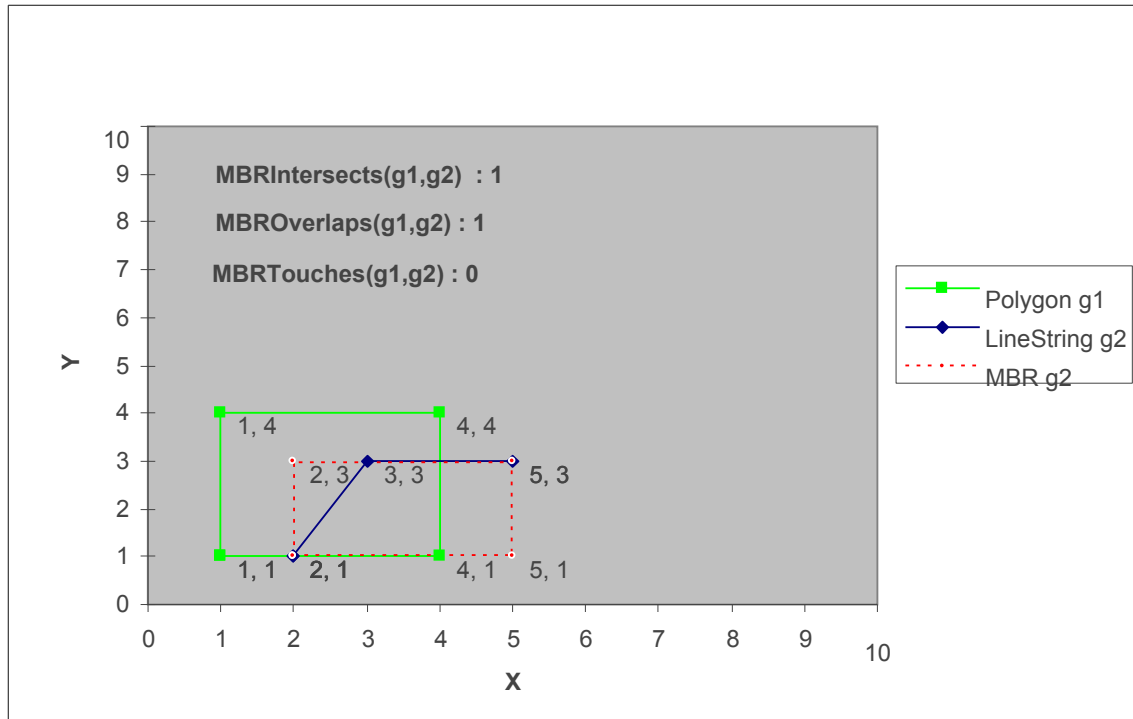
```
SET @g1 = GeomFromText('Polygon((1 1,4 1,4 4,1 4,1 1))');
```

```
SET @g2 = GeomFromText('LineString(2 1,3 3,5 3)');
```

```
SELECT MBRTOUCHES(@g1,@g2);
```

```
/* Result : «1 rows fetched (78 ms)» */
```

Αποτέλεσμα : 0



Συνοπτικά βλέπουμε ότι η πρώτη συνάρτηση επιστρέφει ένα διότι τα ελάχιστα τετράγωνα διασταυρώνονται, η δεύτερη επιστρέφει ένα διότι τα αντικείμενα επικαλύπτονται και δημιουργείται ένα νέο πολύγωνο με συντεταγμένες (2 1,4 1,4 3,2 3,2 1) που δεν ταυτίζεται με κανένα από τα αντικείμενα μας και τέλος η τρίτη συνάρτηση επιστρέφει μηδέν διότι μπορεί σε κάποιο σημείο να εφάπτονται αλλά διασταυρώνονται κιάλας.

Οι συναρτήσεις που μελετήσαμε παραπάνω αποτελούν ένα μέρος αυτών που ορίζονται από τον διεθνή οργανισμό OGC και τις υποστηρίζει η MySQL με την διαφορά ότι τις υλοποιεί σε σχέση με τα ελάχιστα τετράγωνα των αντικειμένων και όχι για τα ίδια τα αντικείμενα όπως ορίζει και ο οργανισμός με αποτέλεσμα να μην έχουμε ακριβή αποτελέσματα, κάτι που είναι πολύ σημαντικό όταν πρόκειται για γεωμετρικά δεδομένα.

7.2.2 2^η Κατηγορία συναρτήσεων της βάσης PostgreSQL

Στη συνέχεια θα δούμε τις συναρτήσεις που υποστηρίζει η PostgreSQL και ελέγχουν την σχέση μεταξύ των γεωμετρικών αντικειμένων. Τις περισσότερες από αυτές τις συναντήσαμε στη MySQL αλλά η μεγάλη διαφορά είναι πως τώρα ισχύουν για τα ίδια τα αντικείμενα και όχι για τα ελάχιστα τετράγωνα τους. Για τα παραδείγματα θα χρησιμοποιήσουμε τον πίνακα geom με στήλη για αποθήκευση των γεωμετρικών δεδομένων geom_obj. Ο πίνακας αυτός έχει ένα δεδομένο τύπου γραμμής με συντεταγμένες (1 1,2 2) και ένα πολύγωνο με συντεταγμένες (1 1,3 1,3 3,1 3,1 1).

- **Equals(g1,g2)**

Επιστρέφει ένα εάν τα δύο αντικείμενα είναι ίσα.

```
SELECT * FROM geom
WHERE equals(geom_obj,'LINESTRING(1 1,2 2)');
/* Result : "1 rows fetched (63 ms)" */
```

Αποτέλεσμα : geom_id=1 & geom_name=LineString

Στο ερώτημα αυτό αναζητούμε τα δεδομένα του πίνακα geom που είναι ίσα με την γραμμή που έχει συντεταγμένες (1 1,2 2).

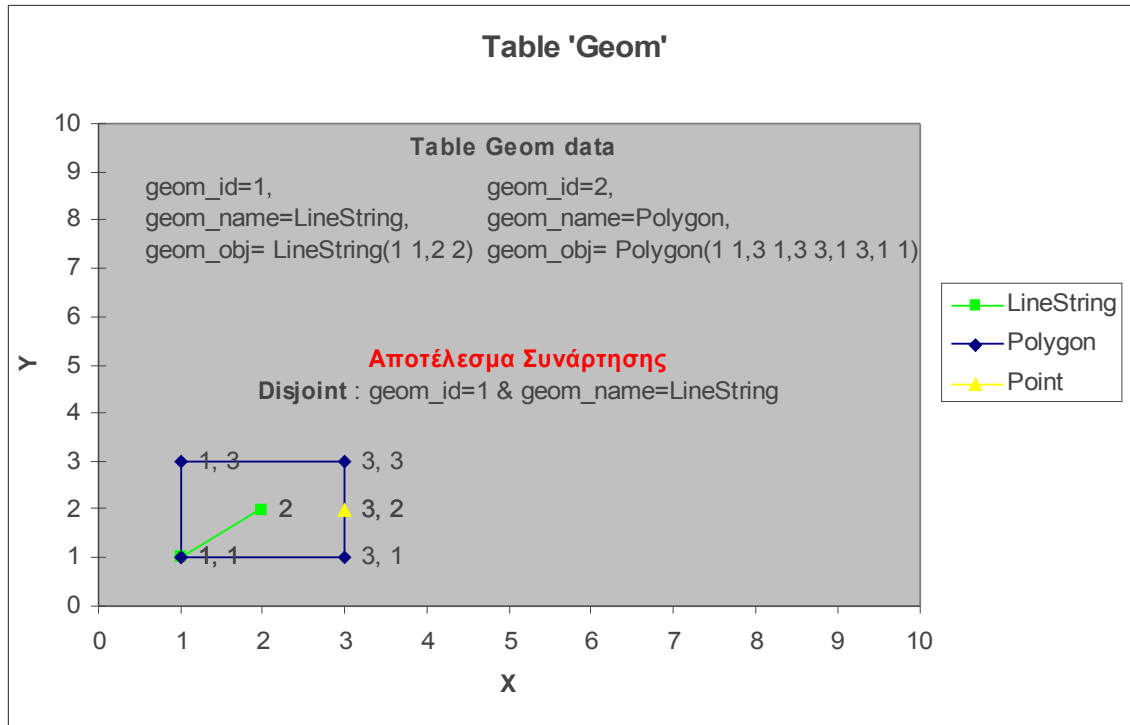
- **Disjoint(g1,g2)**

Επιστρέφει ένα εάν τα αντικείμενα δεν έχουν κοινό σημείο μεταξύ τους και μηδέν εάν έχουν.

```
SELECT * FROM geom
WHERE disjoint(geom_obj,'point(3 2)');
/* Result : "1 rows fetched (78 ms)" */
```

Αποτέλεσμα : geom_id=1 & geom_name=LineString

Στα παραδείγματα μας ελέγχουμε την σχέση όλων των αντικειμένων της στήλης geom_obj του πίνακα geom σε σχέση με κάποιο άλλο αντικείμενο. Έτσι τα αποτελέσματα των συναρτήσεων είναι τα αντικείμενα του πίνακα που ικανοποιούν την εκάστοτε συνθήκη.



- **Intersects(g1,g2)**

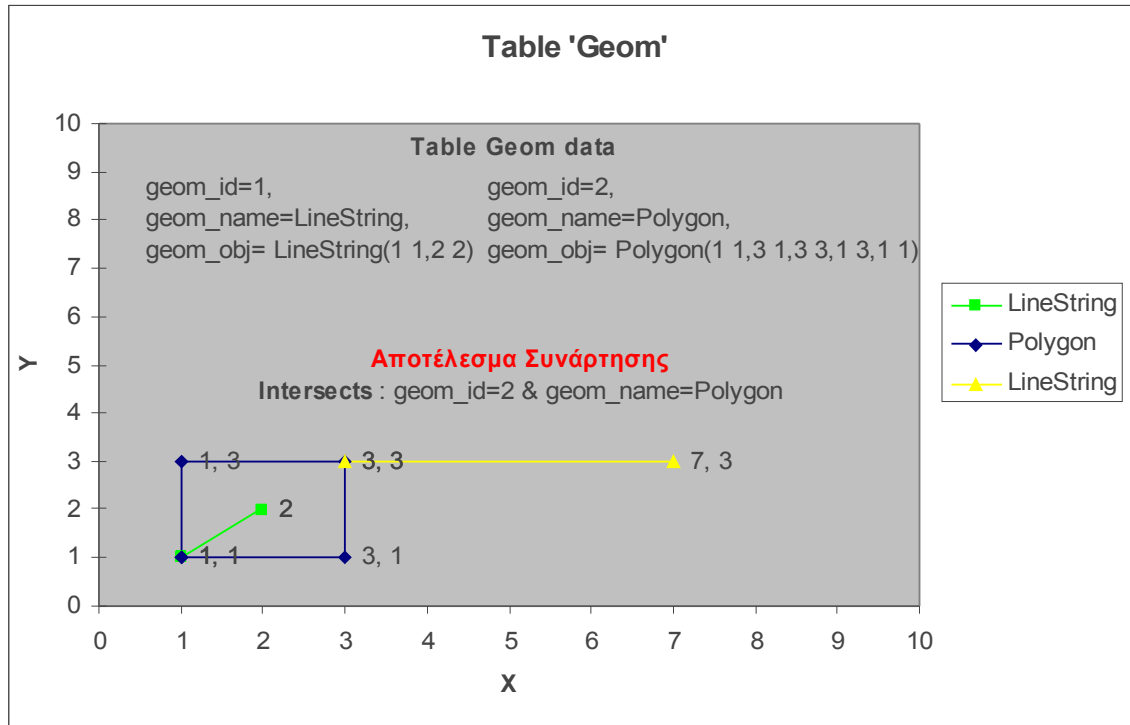
Επιστρέφει ένα εάν τα αντικείμενα διασταυρώνονται

SELECT * FROM geom

WHERE intersects(geom_obj,'linestring(3 3,7 3)');

/ Result : "1 rows fetched (93 ms)" */*

Αποτέλεσμα : geom_id=2 & geom_name=Polygon



- **Touches(g1,g2)**

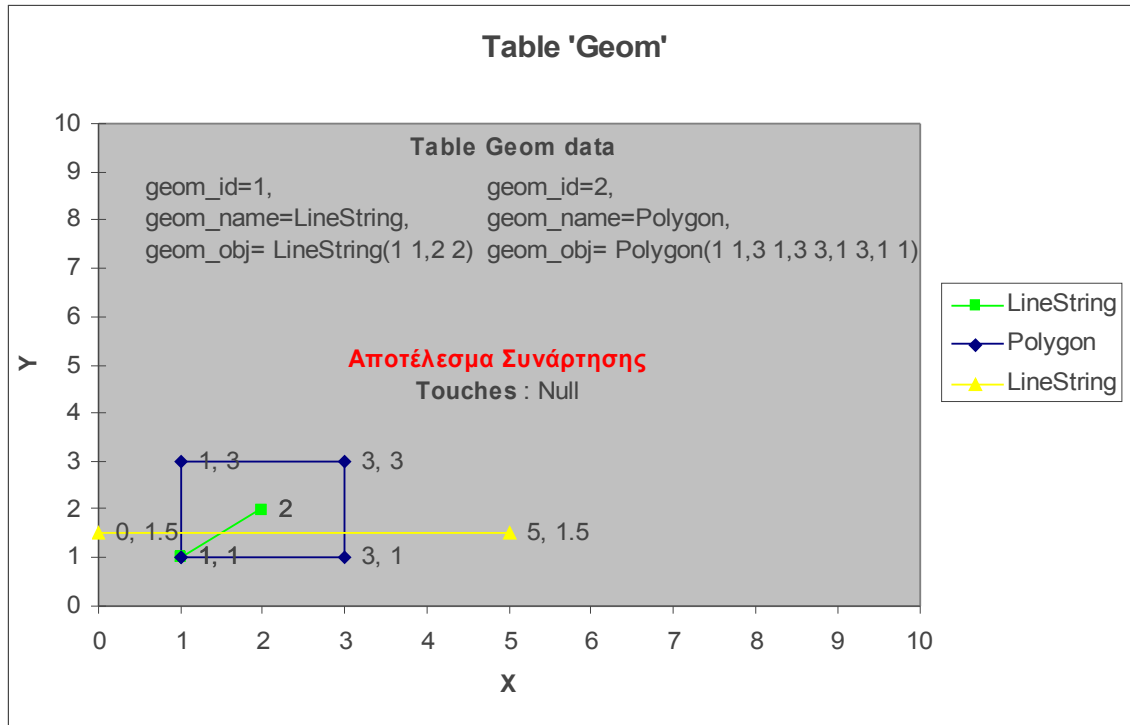
Επιστρέφει 1 εάν τα αντικείμενα εφάπτονται

```
SELECT * FROM geom
```

```
WHERE touches(geom_obj,'linestring(0 1.5,5 1.5)');
```

```
/* Result : "1 rows fetched (93 ms)" */
```

Αποτέλεσμα : geom_id=2 & geom_name=Polygon



- **Within(g1,g2)**

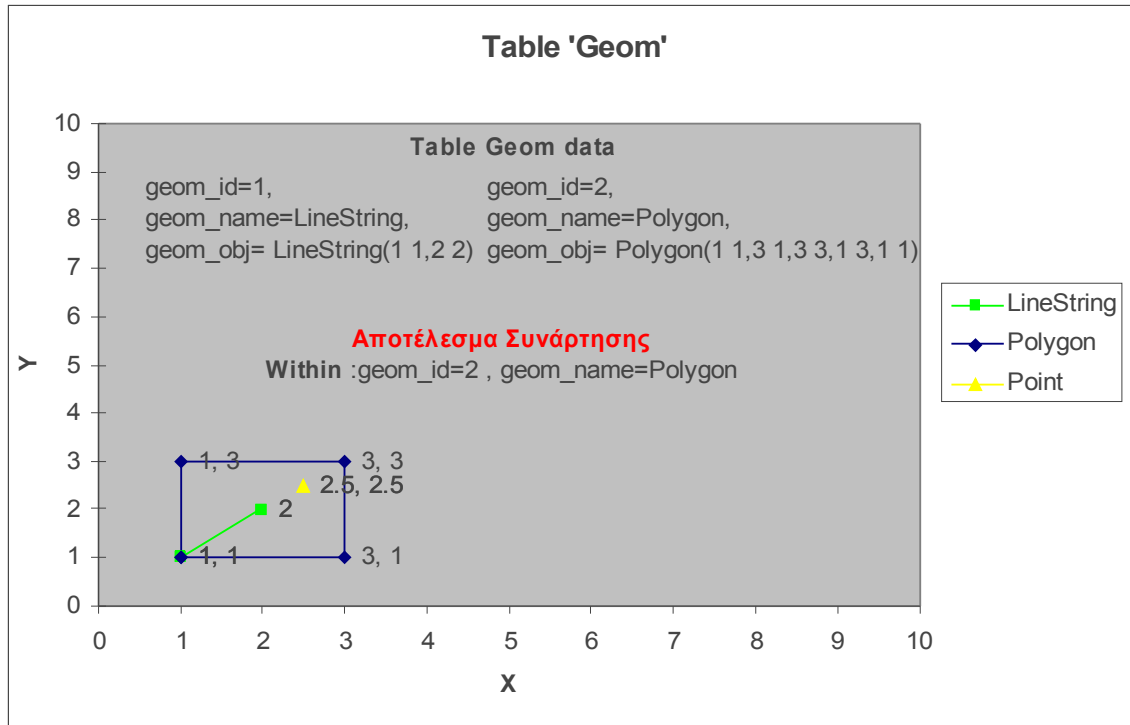
Επιστρέφει 1 εάν το πρώτο γεωμετρικό αντικείμενο περικλείεται από το δεύτερο αντικείμενο

SELECT * FROM geom

WHERE within('POINT(2 2)',geom_obj);

/* Result : "1 rows fetched (109 ms)" */

Αποτέλεσμα : geom_id=2 & geom_name=Polygon



- **Overlaps(g1,g2)**

Επιστρέφει ένα εάν τα αντικείμενα επικαλύπτονται

```
SELECT * FROM geom
```

```
WHERE overlaps(geom_obj,'Polygon((0 0,3 0,3 3,0 3,0 0))');
```

```
/* Result : "Empty set (80 ms)" */
```

Βλέπουμε ότι το αποτέλεσμα είναι το κενό σύνολο

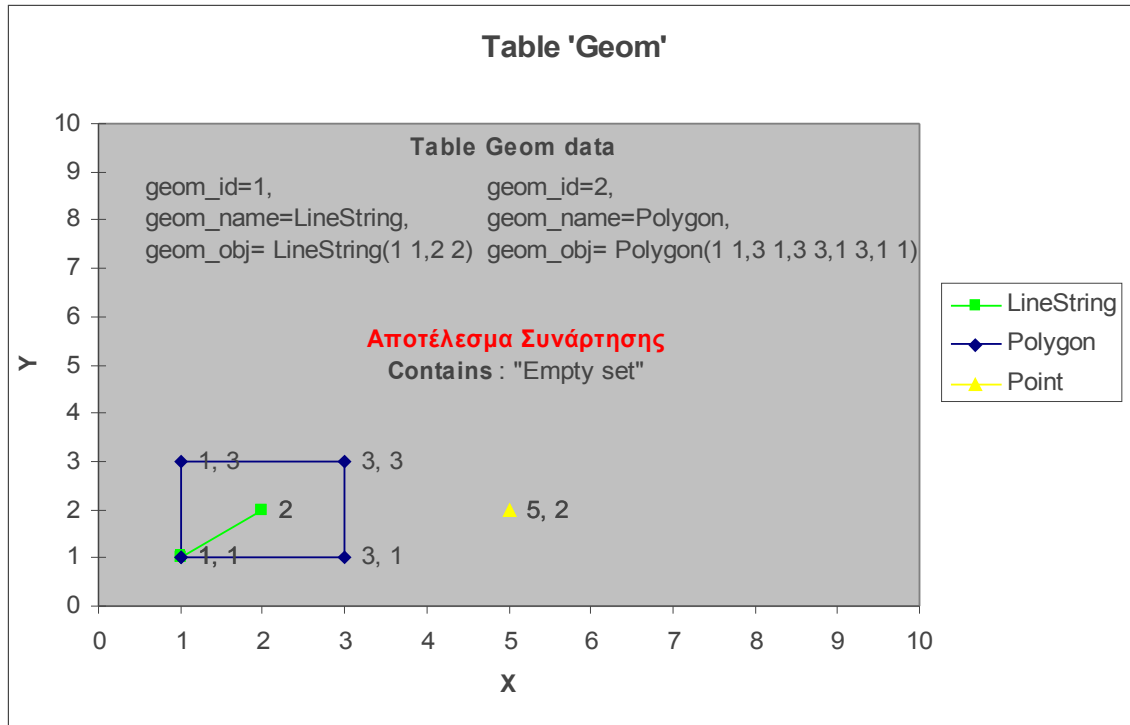
- **Contains(g1,g2)**

Επιστρέφει ένα εάν το πρώτο αντικείμενο περιέχει εξολοκλήρου το δεύτερο

```
SELECT * FROM geom
```

```
WHERE contains(geom_obj,'POINT(5 2)');
```

```
/* Result : "Empty set (78 ms)" */
```



- **Crosses(g1,g2)**

Ελέγχει εάν τα δύο αντικείμενα διασταυρώνονται σε κάποιο σημείο.

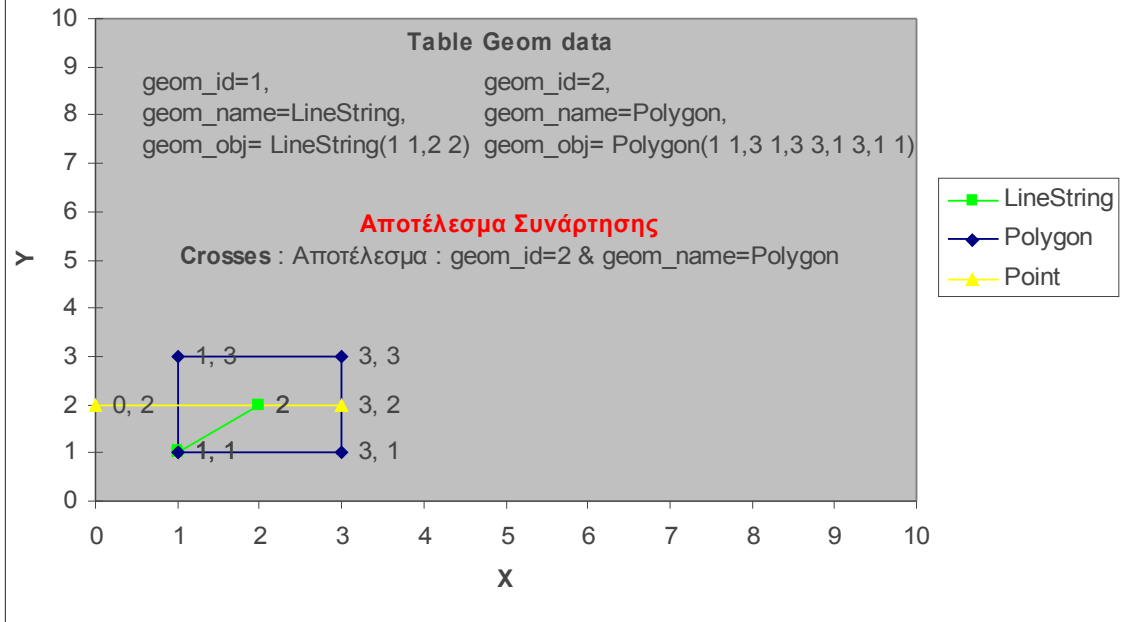
```
SELECT * FROM geom
```

```
WHERE crosses(geom_obj,'linestring(2 1.5,4 1.5)');
```

```
/* Result : "1 rows fetched (79 ms)" */
```

Αποτέλεσμα : geom_id=2 & geom_name=Polygon

Table 'Geom'



7.2.3 2^η Κατηγορία συναρτήσεων της βάσης Oracle

Περνάμε τώρα στις συναρτήσεις της Oracle που ελέγχουν την σχέση μεταξύ γεωμετρικών αντικειμένων.

- **SDO_RELATE**

Καθορίζει εάν δύο γεωμετρικά αντικείμενα αλληλεπιδρούν κατά κάποιον τρόπο. Για να καθορισθεί ο τρόπος αλληλεπίδρασης υπάρχουν αρκετοί τελεστές που μπορούν να χρησιμοποιηθούν για να διαπιστωθεί η ακριβής συσχέτιση των αντικειμένων

- **SDO_ANYINTERACT**

Ελέγχει εάν κάποια από τα γεωμετρικά αντικείμενα μιας στήλης ενός πίνακα έχουν οποιαδήποτε τοπολογική σχέση με ένα καθορισμένο αντικείμενο. Εάν υπάρχει σχέση μεταξύ δύο αντικειμένων επιστρέφει “true” διαφορετικά επιστρέφει “false”

```
SELECT g.geom2_id FROM geom2 g
WHERE SDO_ANYINTERACT(g.geom2_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(3,3, 8,8))
)= 'TRUE';
```

/ Result : “2 rows fetched (282 ms)” */*

Αποτέλεσμα : GEOM4_ID (2), GEOM4_ID (1)

- **SDO_CONTAINS**

Ελέγχει εάν κάποια από τα γεωμετρικά αντικείμενα μιας στήλης ενός πίνακα περιέχουν ένα άλλο γεωμετρικό δεδομένο μιας στήλης ενός πίνακα ή κάποιο μεμονωμένο αντικείμενο.

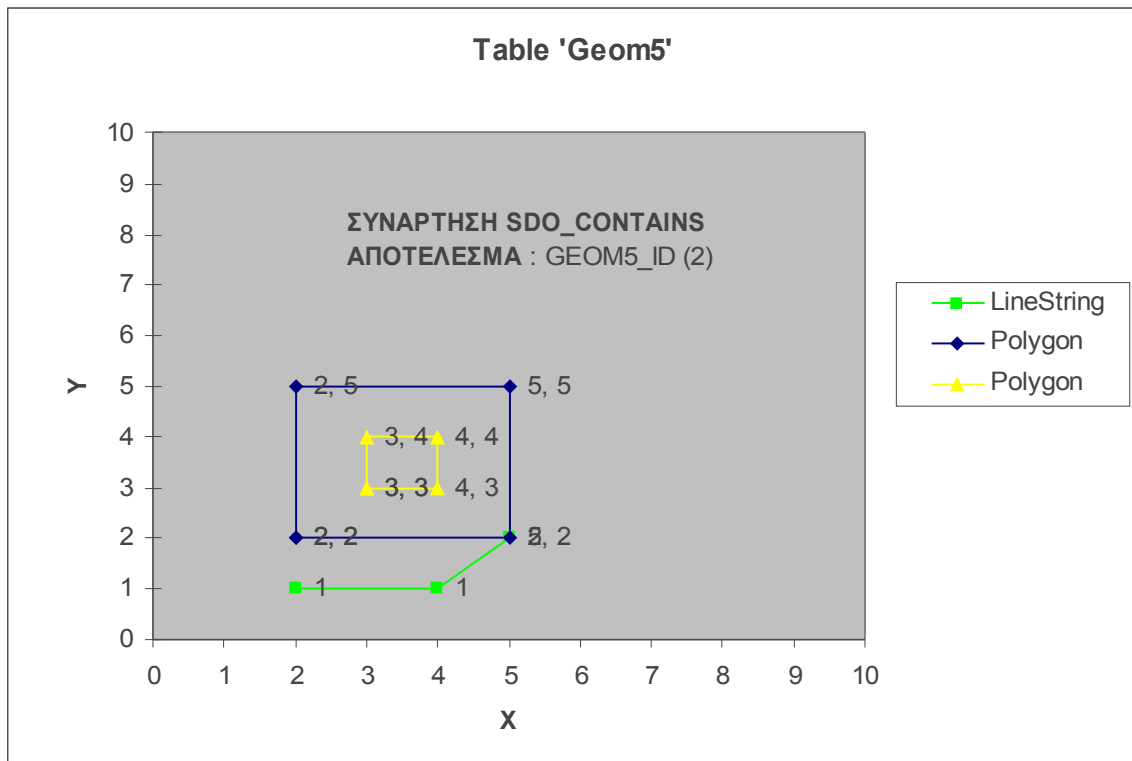
```

SELECT g.geom5_id
FROM geom5 g
WHERE SDO_CONTAINS(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1
,1003,1),
SDO_ORDINATE_ARRAY(3,3, 4,3, 4,4, 3,4, 3,3))
)= 'TRUE';
/* Result : "1 rows fetched (93 ms)" */

```

Αποτέλεσμα : GEOM5_ID (2)

Στο παραπάνω παράδειγμα ελέγχουμε εάν κάποιο από τα αντικείμενα του πίνακα Geom5 περιέχει το πολύγωνο που δίνουμε ως δεύτερο όρισμα στη συνάρτηση. Το παράδειγμα φαίνεται πιο παραστατικά στο διάγραμμα που ακολουθεί.

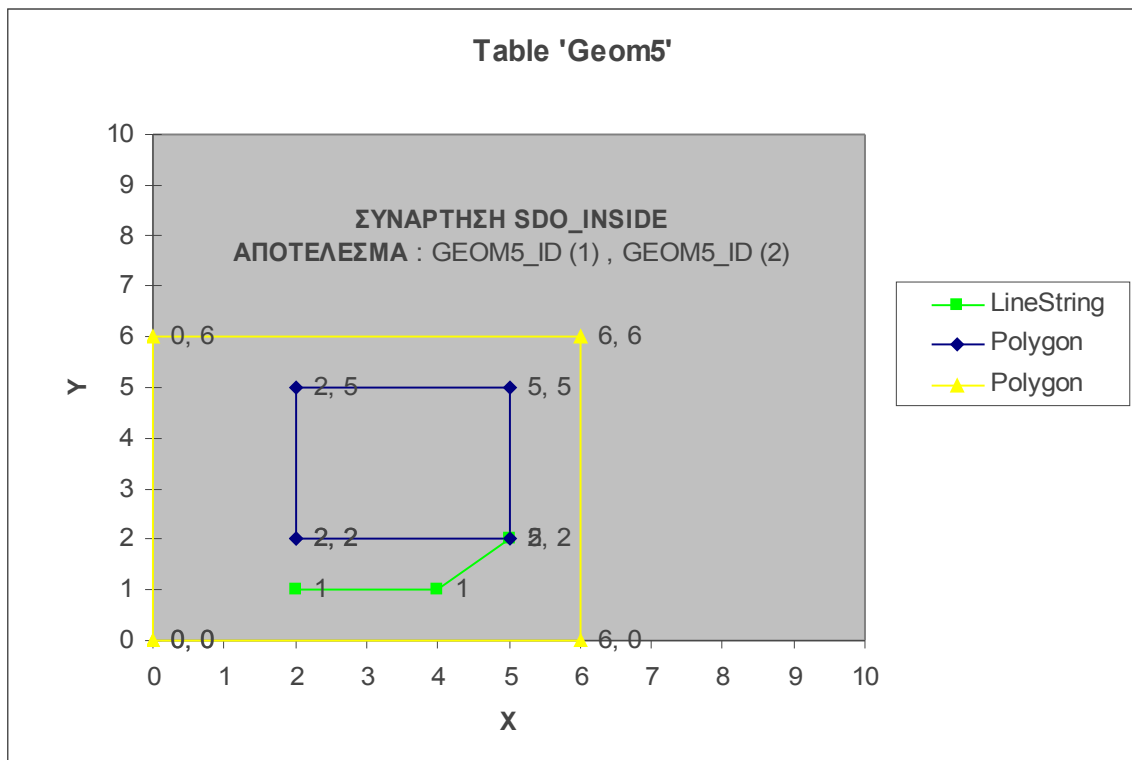


- **SDO_INSIDE**

Είναι η αντίστροφη συνάρτηση της contains που είδαμε παραπάνω και ελέγχει εάν το αντικείμενο που δίνουμε ως δεύτερο όρισμα περιέχει κάποιο από τα αντικείμενα της στήλης του πίνακα Geom5.

```
SELECT g.geom5_id
FROM geom5 g
WHERE SDO_INSIDE(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,1),
SDO_ORDINATE_ARRAY(0,0, 6,0, 6,6, 0,6, 0,0))
)= 'TRUE';
* Result : "2 rows fetched (78 ms)" */
```

Αποτέλεσμα : GEOM5_ID (1) , GEOM5_ID (2)



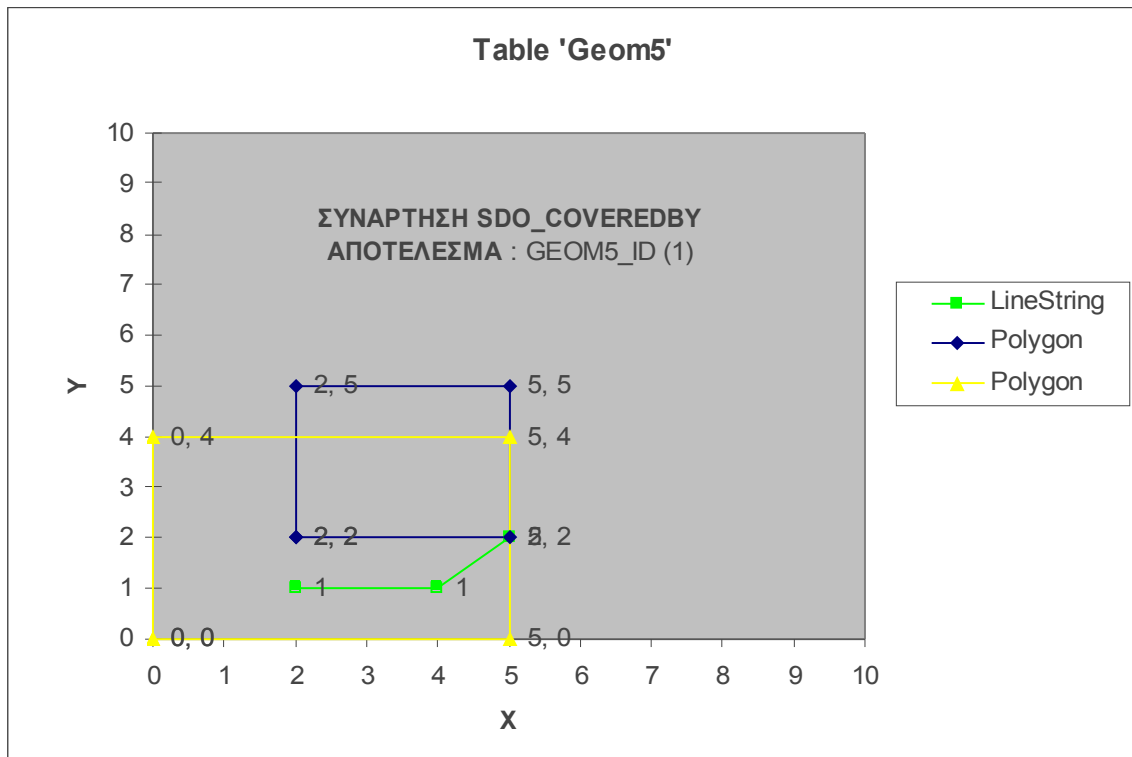
- **SDO_COVEREDBY**

Ελέγχει εάν ένα δεδομένο γεωμετρικό αντικείμενο περιέχει κάποιο από τα γεωμετρικά αντικείμενα ενός πίνακα

```
SELECT g.geom5_id
FROM geom5 g
WHERE SDO_COVEREDBY(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(0,0, 6,6)
)= 'TRUE');
```

1 rows fetched (266 ms)

Αποτέλεσμα : GEOM5_ID (1)



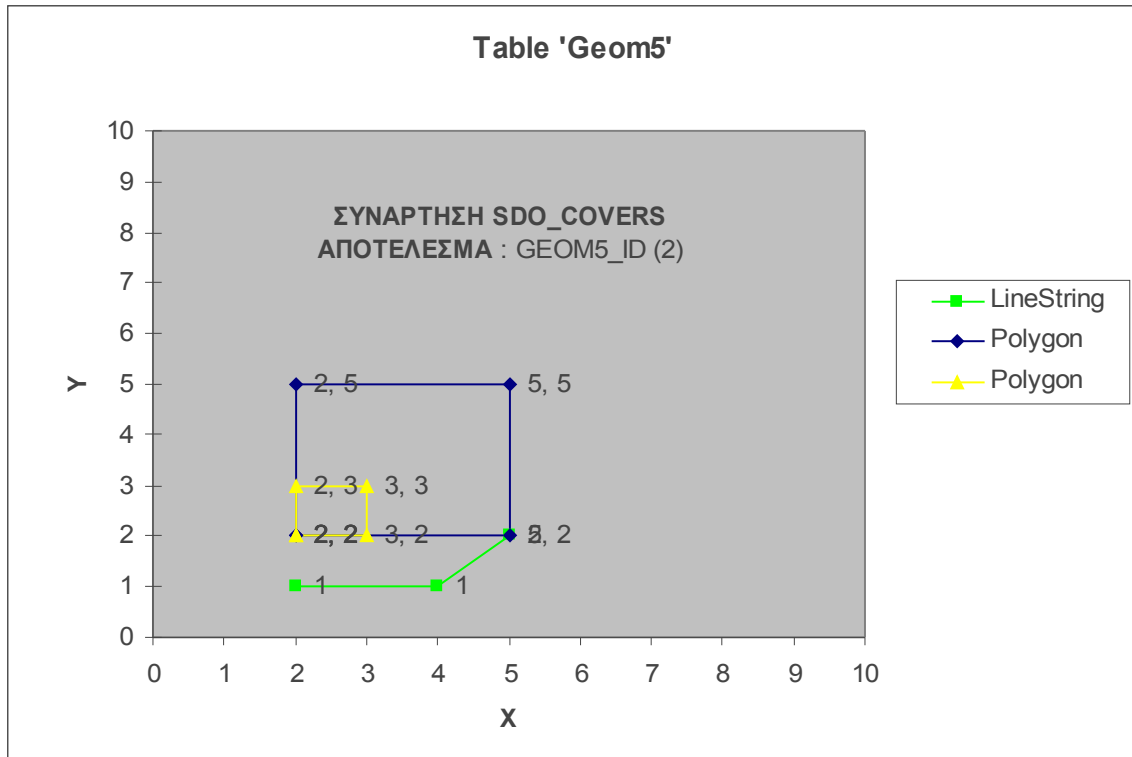
- **SDO_COVERS**

Ελέγχει εάν ένα δεδομένο γεωμετρικό αντικείμενο καλύπτει κάποιο άλλο από τα αυτά που είναι αποθηκευμένα στον πίνακα geom5.

```
SELECT g.geom5_id
FROM geom5 g
WHERE SDO_COVERS(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,1),
SDO_ORDINATE_ARRAY(2,2, 3,2, 3,3, 2,3, 2,2))
)= 'TRUE';
```

/* Result : 1 rows fetched (281 ms)

Αποτέλεσμα : GEOM5_ID (2)

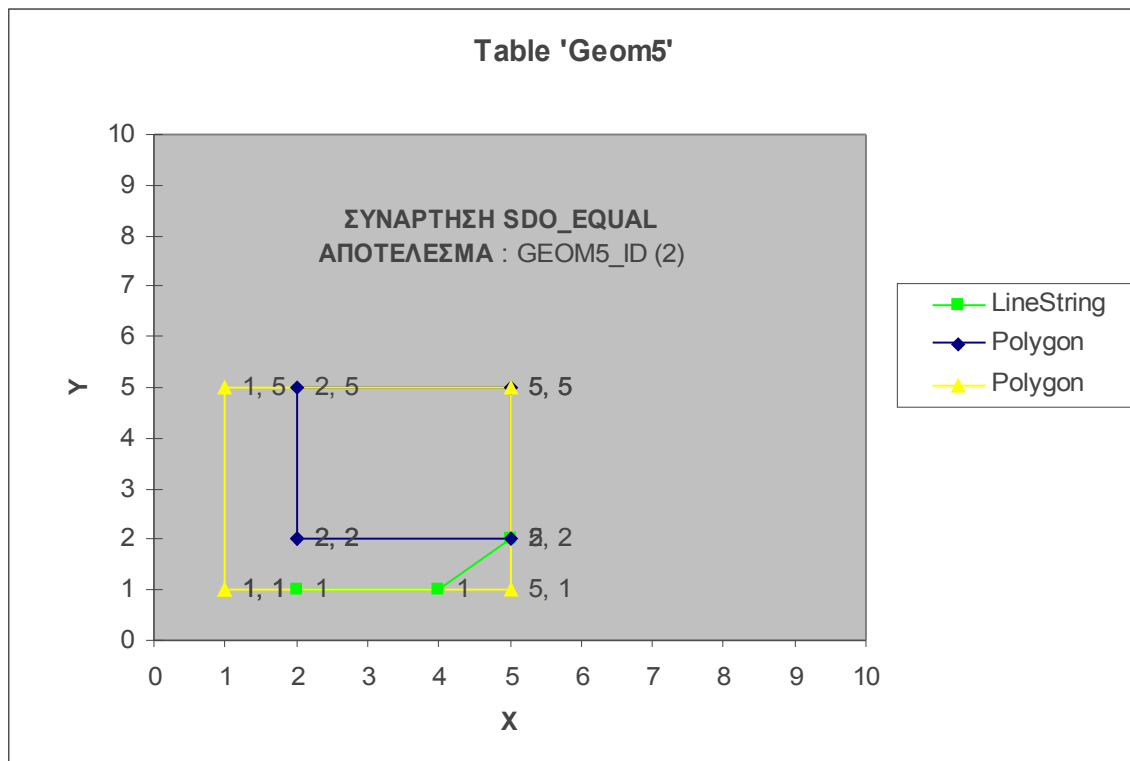


- **SDO_EQUAL**

Ελέγχει εάν κάποιο από τα γεωμετρικά αντικείμενα του πίνακα Geom5 είναι ίσο με το αντικείμενο που δίδεται ως δεύτερο όρισμα στη συνάρτηση

```
SELECT g.geom5_id
FROM geom5 g
WHERE SDO_EQUAL(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(1,1, 5,5))
)= 'TRUE';
```

Αποτέλεσμα : Empty set (265 ms)



- **SDO_ON**

Ελέγχει εάν κάποια γεωμετρικά αντικείμενα σε ένα πίνακα είναι γεωμετρικά 'πάνω' από ένα δεδομένο αντικείμενο

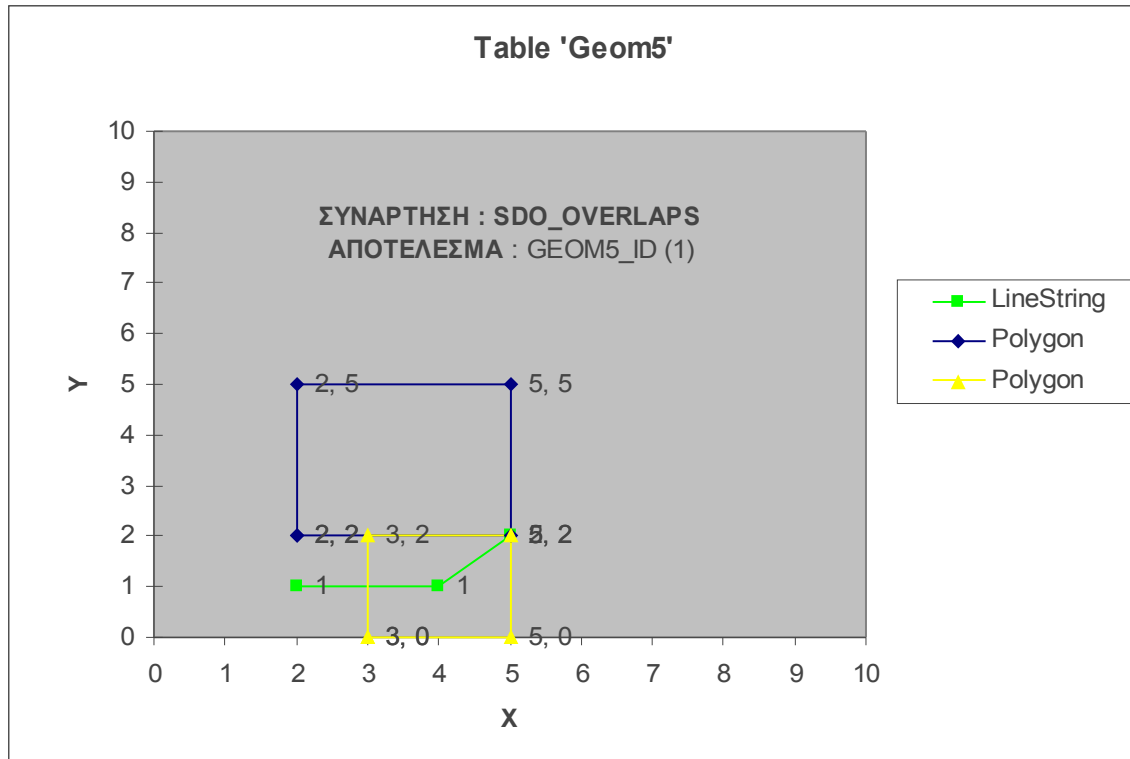
- **SDO_OVERLAPS**

Ελέγχει εάν κάποιο από τα γεωμετρικά αντικείμενα ενός πίνακα επικαλύπτονται με ένα δεδομένο αντικείμενο

```
SELECT g.geom5_id  
FROM geom5 g  
WHERE SDO_OVERLAPS(g.geom5_obj,  
SDO_GEOMETRY(2003, NULL, NULL,  
SDO_ELEM_INFO_ARRAY(1,1003,3),  
SDO_ORDINATE_ARRAY(3,0, 5,2))  
) = 'TRUE';
```

/* Result : 1 rows fetched (297 ms)

Αποτέλεσμα : GEOM5_ID (1)



- **SDO_TOUCH**

Ελέγχει εάν τα γεωμετρικά αντικείμενα ενός πίνακα και συγκεκριμένα στο παράδειγμά μας του πίνακα geom5 εφάπτονται με ένα δεδομένο αντικείμενο που δίδεται ως όρισμα στην συνάρτηση

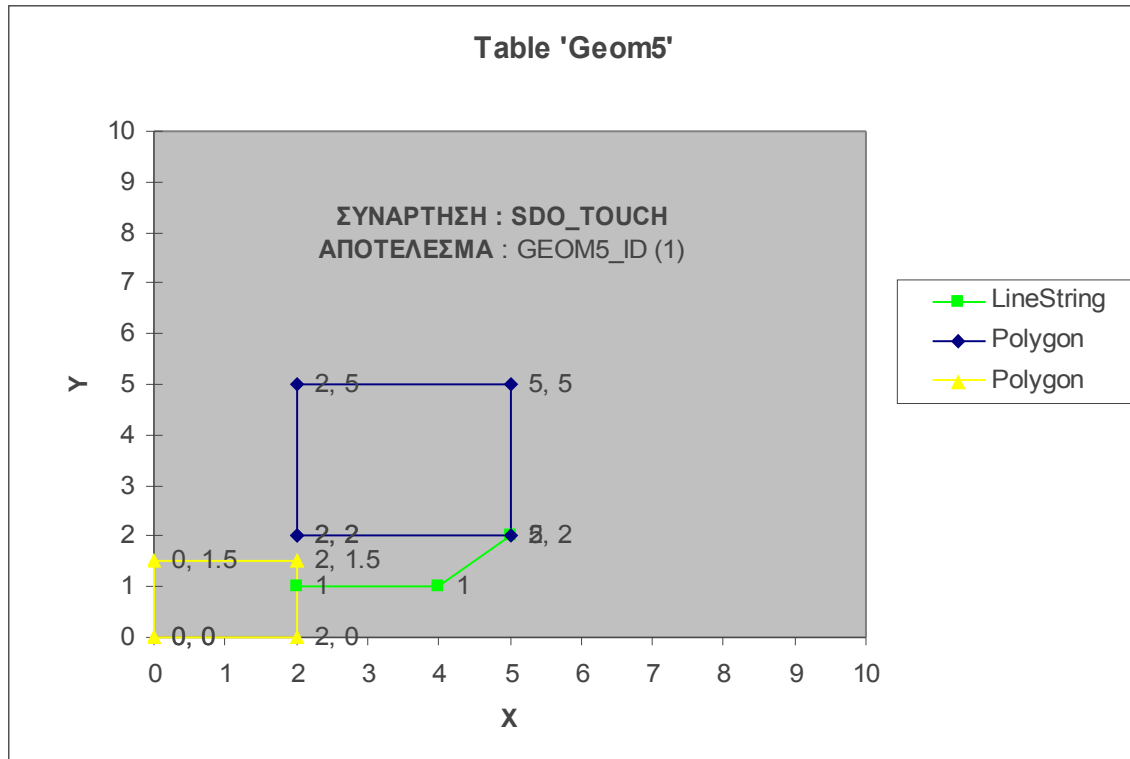
```

SELECT g.geom5_id
FROM geom5 g
WHERE SDO_TOUCH(g.geom5_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,1),
SDO_ORDINATE_ARRAY(0,0, 2,0, 2,1.5, 0,1.5, 0,0))
)= 'TRUE';

```

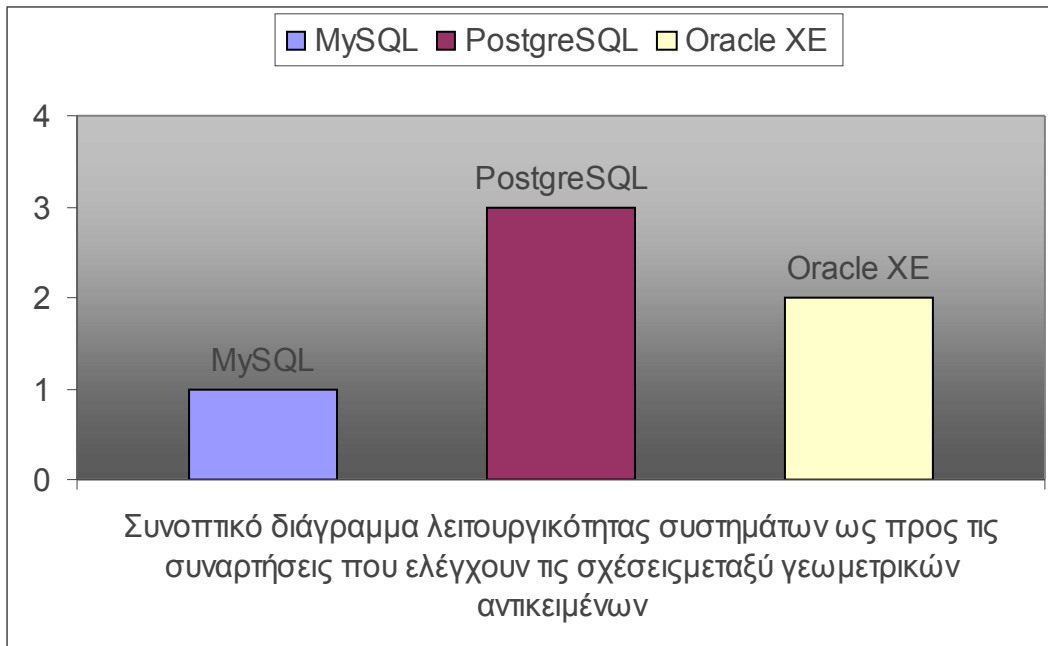
1 rows fetched (641 ms)

Αποτέλεσμα : GEOM5_ID (1)



Σε αυτό το κεφάλαιο μελετήσαμε τις συναρτήσεις που μας βοηθούν να βρούμε την σχέση μεταξύ δύο γεωμετρικών αντικειμένων. Ξεκινώντας από την MySQL και πάλι είδαμε αρκετές συναρτήσεις αυτού του τύπου, γενικώς η βάση αυτή ακολουθεί όπως είδαμε σε όλη την πορεία της μέχρι τώρα ανάλυσης τις προδιαγραφές του οργανισμού OGC. Η μεγάλη όμως διαφορά στις συναρτήσεις που είδαμε τώρα σε σχέση με τα προηγούμενα κεφάλαια είναι ότι οι συναρτήσεις αυτές αναφέρονται στα ελάχιστα τετράγωνα των γεωμετρικών αντικειμένων και όχι στα ίδια τα αντικείμενα, που είναι πολύ σημαντικό γιατί τα αποτελέσματα αυτών των συναρτήσεων δεν δίνουν στην ουσία την πραγματική σχέση μεταξύ των αντικειμένων. Από την άλλη μεριά είδαμε ότι η PostgreSQL υλοποιεί όλες τις συναρτήσεις που προτείνονται από τον οργανισμό και μας δίνουν αποτελέσματα για τα ίδια τα αντικείμενα και όχι για τα ελάχιστα τετράγωνα τους. Τέλος η ORACLE είδαμε ότι υλοποιεί τις περισσότερες από τις συναρτήσεις οπότε μπορούμε να πούμε ότι η βάση που παρουσιάζεται πιο πλήρης σε αυτό τον τομέα είναι και πάλι η PostgreSQL,

ακολουθεί η ORACLE και τέλος η MySQL που όπως είδαμε παρουσιάζει το σημαντικό μειονέκτημα ότι ελέγχει στην ουσία τα ελάχιστα τετράγωνα των αντικειμένων.



7.3 3^η Κατηγορία συναρτήσεων

Σε αυτή την κατηγορία θα μελετήσουμε τις συναρτήσεις με τις οποίες μπορούμε να κάνουμε χωρική ανάλυση στα δεδομένα μας. Θα εξετάσουμε μόνο τις δύο βάσεις δεδομένων διότι όπως είδαμε και παραπάνω η MySQL δεν υποστηρίζει συναρτήσεις αυτής της κατηγορίας

7.3.1 3^η Κατηγορία συναρτήσεων της βάσης PostgreSQL

Όσον αφορά την PostgreSQL λοιπόν έχουμε την δυνατότητα να χρησιμοποιήσουμε τις παρακάτω συναρτήσεις.

Distance(geometry, geometry)

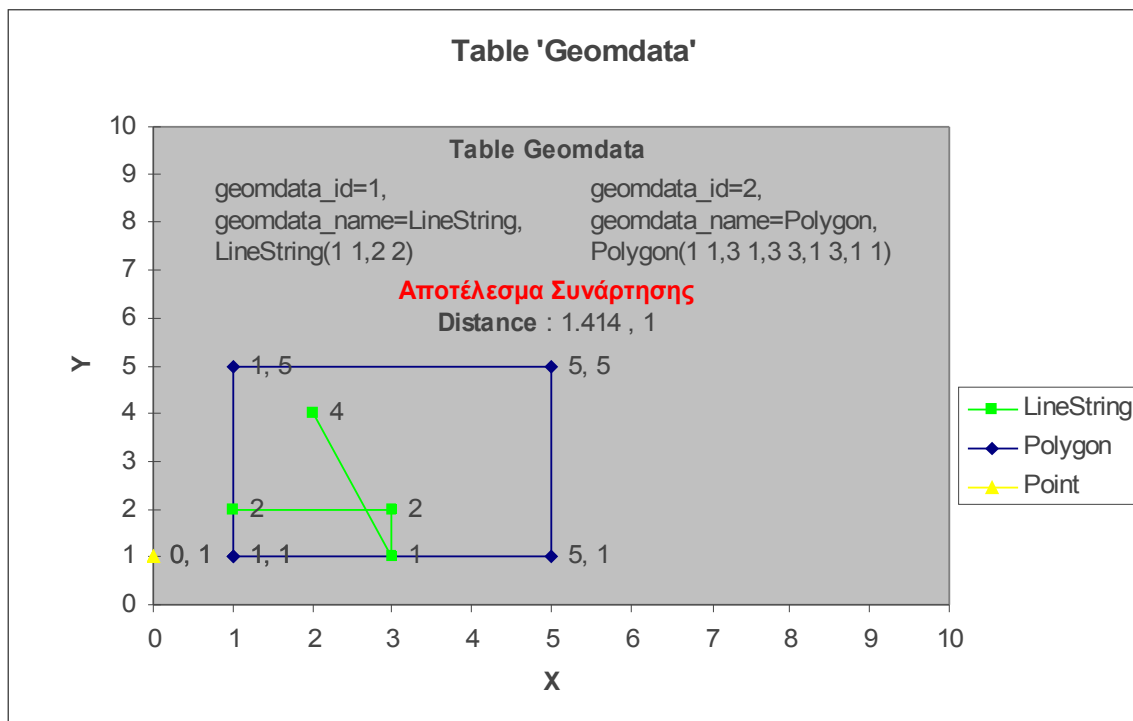
Μας επιστρέφει την απόσταση μεταξύ των γεωμετρικών αντικειμένων.

```
select distance(geomdata_obj, 'POINT(0 1)') from geomdata;
```

```
/* Result : "2 rows fetched (63 ms)" */
```

Αποτέλεσμα : 1.4142135623731

1



Buffer(geometry, double, [integer])

Δημιουργεί μια ζώνη γύρω από το γεωμετρικό αντικείμενο ανάλογη του ακέραιου αριθμού που δίνουμε στην συνάρτηση.

```
SELECT astext(buffer(geomdata_obj,1)) from geomdata
```

```
Where geomdata_id=2;
```

```
/* Result : “1 rows fetched (62 ms)” */
```

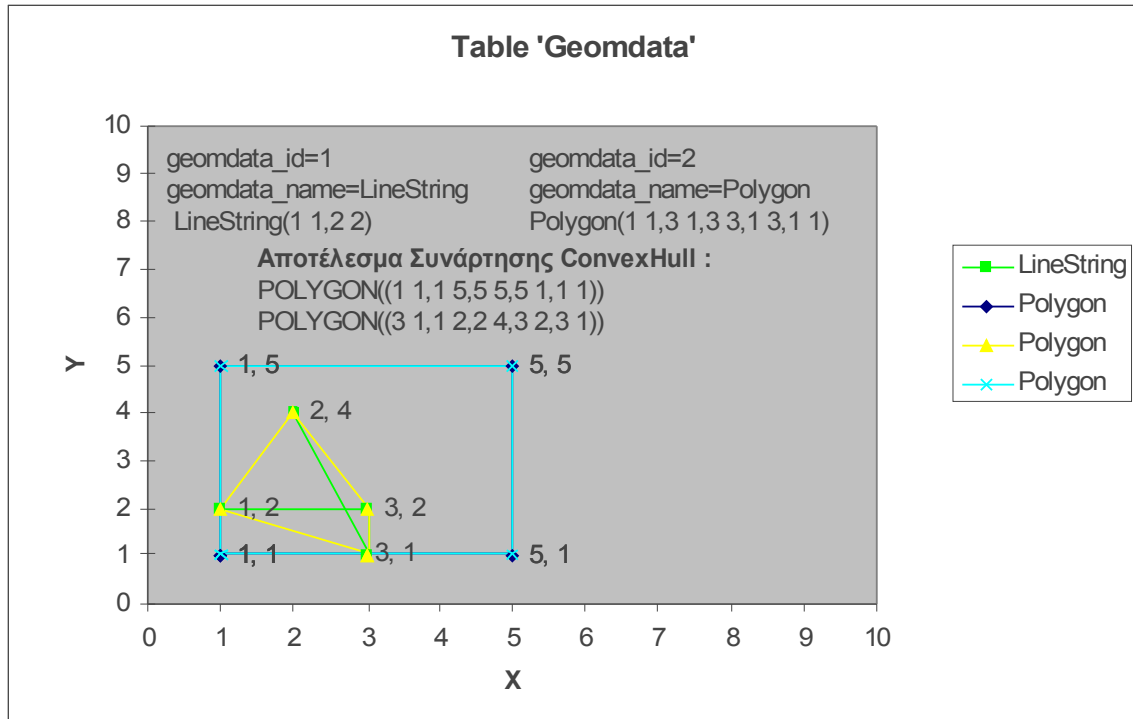
```
Αποτέλεσμα : POLYGON((0 1,0 5,0.0192147195967696  
5.19509032201613,0.0761204674887133 5.38268343236509,0.168530387697455  
5.5555702330196,0.292893218813453 5.70710678118655,0.444429766980398  
5.83146961230255,0.61731656763491 5.92387953251129,0.804909677983872  
5.98078528040323,1 6,5 6,5.19509032201613 5.98078528040323,5.38268343236509  
5.92387953251129,5.5555702330196 5.83146961230254,5.70710678118655  
5.70710678118655,5.83146961230254 5.5555702330196,5.92387953251129  
5.38268343236509,5.98078528040323 5.19509032201613,6 5,6 1,5.98078528040323  
0.804909677983872,5.92387953251129 0.61731656763491,5.83146961230254  
0.444429766980398,5.70710678118655 0.292893218813453,5.5555702330196  
0.168530387697455,5.38268343236509 0.0761204674887133,5.19509032201613  
0.0192147195967696,5 0,1 0,0.804909678005889  
0.0192147195923901,0.617316567676389 0.076120467471532,0.444429767036393  
0.16853038766004,0.292893218876946 0.292893218749959,0.168530387759813  
0.444429766887072,0.0761204675402568 0.617316567510473,0.0192147196274256  
0.804909677829753,0 0.999999999820414,0 1))
```

ConvexHull(geometry)

```
select astext(convexhull(geomdata_obj)) from geomdata;
```

```
/* Result : “2 rows fetched (93 ms)” */
```

```
Αποτέλεσμα : POLYGON((3 1,1 2,2 4,3 2,3 1))  
POLYGON((1 1,1 5,5 5,5 1,1 1))
```



SymDifference(geometry A, geometry B)

```
SELECT astext(SymDifference(geomdata_obj,'point(6 6)'))
from geomdata;
```

/ Result : "2 rows fetched (1.031 sec)" */*

Αποτέλεσμα: GEOMETRYCOLLECTION(POINT(6 6),LINESTRING(1 2,2.666666666666667 2),LINESTRING(2.666666666666667 2,3 2,3 1,2.666666666666667 2),LINESTRING(2.666666666666667 2,2 4))
 GEOMETRYCOLLECTION(POINT(6 6),POLYGON((1 1,1 5,5 5,5 1,1 1)))

Difference(geometry A, geometry B)

```
SELECT astext(Difference(geomdata_obj,'point(6 6)'))
from geomdata;
```

/ Result : "2 rows fetched (125 ms)" */*

Αποτέλεσμα: MULTILINESTRING((1 2,2.666666666666667 2),(2.666666666666667 2,3 2,3 1,2.666666666666667 2),(2.666666666666667 2,2 4))
 POLYGON((1 1,1 5,5 5,5 1,1 1))

7.3.2 3^η Κατηγορία συναρτήσεων της βάσης Oracle

Περνάμε στις συναρτήσεις που υποστηρίζει η Oracle σχετικά με χωρική ανάλυση δεδομένων.

- **SDO_FILTER**

Καθορίζει ποια γεωμετρικά αντικείμενα μπορούν να σχετιστούν με ένα δεδομένο αντικείμενο

```
SELECT g.geom4_id
FROM geom4 g
WHERE SDO_filter(g.geom4_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(5,5, 9,9))) = 'TRUE';
/* Result : «1 rows fetched (203 ms)» */
```

Αποτέλεσμα : GEOM4_ID (2)

- **SDO_JOIN**

Επιστρέφει ένα πίνακα με χωρικά δεδομένα που προκύπτουν από την συνένωση δύο πινάκων από μια σχέση

- **SDO_NN**

Καθορίζει το γεωμετρικό αντικείμενο που είναι στην πιο κοντινή απόσταση από ένα δεδομένο

```
SELECT g.geom4_id
FROM geom4 g
WHERE SDO_nn(g.geom4_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(6,1, 8,6)),
'sdo_num_res=1') = 'TRUE';
/* Result : “1 rows fetched (203 ms)” */
```

Αποτέλεσμα : GEOM4_ID (1)

- **SDO_NN_DISTANCE**

Επιστρέφει την απόσταση ενός γεωμετρικού αντικειμένου το οποίο είναι αποτέλεσμα του τελεστή SDO_NN

```
SELECT g.geom4_id,sdo_nn_distance(1) dist
FROM geom4 g
WHERE SDO_nn(g.geom4_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(7,7, 9,9)),
'sdo_num_res=1',1) = 'TRUE';
```

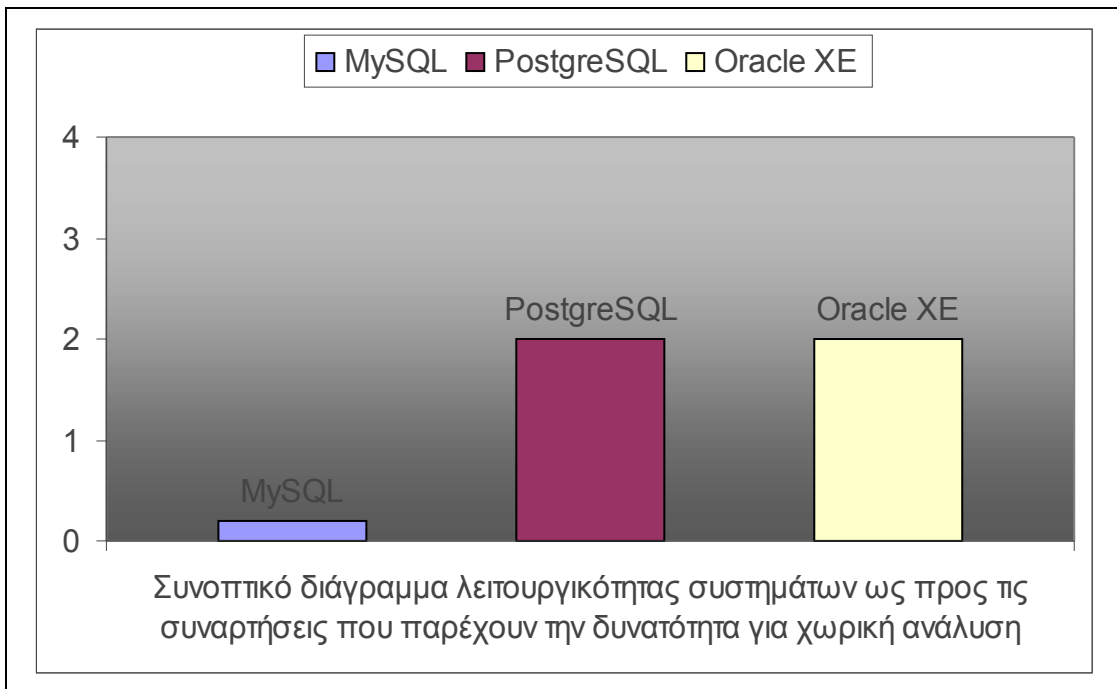
- **SDO_WITHIN_DISTANCE**

Καθορίζει εάν δύο γεωμετρικά αντικείμενα είναι σε απόσταση μικρότερη από μια συγκεκριμένη

```
SELECT g.geom4_id
FROM geom4 g
WHERE SDO_within_distance(g.geom4_obj,
SDO_GEOMETRY(2003, NULL, NULL,
SDO_ELEM_INFO_ARRAY(1,1003,3),
SDO_ORDINATE_ARRAY(3,3, 9,9)),
'distance=2') = 'TRUE';
/* Result : "2 rows fetched (297 ms)" */
```

Αποτέλεσμα : GEOM4_ID (1), GEOM4_ID (2)

Σε ότι αφορά λοιπόν τις συναρτήσεις που μπορούμε να χρησιμοποιήσουμε για χωρική ανάλυση παρατηρούμε ότι και οι δύο βάσεις υποστηρίζουν λίγες μόνο συναρτήσεις και διαφορετικές μεταξύ τους. Έτσι θεωρούμε ότι οι δυνατότητες των δύο λογισμικών σε αυτό τον τομέα είναι περίπου στο ίδιο επίπεδο, σε αντίθεση φυσικά με την MySQL που δεν έχει τέτοιες δυνατότητες. Έτσι σε ένα συνοπτικό διάγραμμα θεωρούμε ισοδύναμα τα δύο λογισμικά.



ΚΕΦΑΛΑΙΟ 8

Χρήση Δεικτών

Η χρήση δεικτών είναι πάρα πολύ σημαντική στις βάσεις δεδομένων λόγω της τεράστιας διαφοράς που παρουσιάζουν τα συστήματα αυτά στην αναζήτηση δεδομένων. Ειδικά όταν πρόκειται για μεγάλο όγκο δεδομένων η χρήση δεικτών κρίνεται απαραίτητη. Στην ουσία οι δείκτες χρησιμοποιούνται για να ελαττώνεται ο αριθμός των αναζητήσεων ενός ερωτήματος σε μια βάση κερδίζοντας πάρα πολύ σε χρόνο.

8.1 Χρήση Δεικτών στην βάση MySQL

Όσον αφορά την MySQL η χρήση χωρικών δεικτών γίνεται κατά τον ίδιο τρόπο με τους κανονικούς δείκτες με την διαφορά ότι χρησιμοποιούμε την λέξη ‘‘χωρικός’’. Επιπλέον θα πρέπει οι στήλες που χρησιμοποιούν δείκτες να είναι δηλωμένες ως μη μηδενικές.

Ένα παράδειγμα δημιουργίας δείκτη με την βοήθεια της γλώσσας ερωτημάτων είναι η εξής εντολή

Alter Table Geom ADD Spatial Index(g);

Όπου τοποθετούμε δείκτη στην γεωμετρική στήλη g του πίνακα Geom.

Η MySQL χρησιμοποιεί δείκτες σε στήλες που περιέχουν χωρικά δεδομένα της μορφής R-Trees. Η χρήση δεικτών γίνεται κάνοντας χρήση το ελάχιστο τετράγωνο ενός γεωμετρικού αντικειμένου. Για οριζόντιες ή κάθετες γραμμές το ελάχιστο τετράγωνο αποτελεί η ίδια η γραμμή καθώς επίσης και όταν πρόκειται για σημεία το ελάχιστο τετράγωνο αποτελείται από το σημείο αυτό.

Τέλος η μόνη μηχανή αποθήκευσης χωρικών δεδομένων που υποστηρίζει χρήση δεικτών στη MySQL είναι η MyISAM.

8.2 Χρήση Δεικτών στην βάση PostgreSQL

Όπως είπαμε και παραπάνω η χρήση δεικτών είναι αυτή που καθιστά μια βάση δεδομένων ικανή να χειριστεί πολλαπλά δεδομένα. Διαφορετικά κάθε αναζήτηση ενός στοιχείου θα απαιτούσε σειριακή αναζήτηση στην βάση σε κάθε μια εγγραφή. Η χρήση δεικτών αυξάνει την ταχύτητα αναζήτησης οργανώνοντας τα δεδομένα σαν ένα δέντρο αναζήτησης που μπορεί γρήγορα να το διατρέξει κανείς για να βρει την εγγραφή που επιθυμεί. Η PostgreSQL χρησιμοποιεί τρία είδη δεικτών

- B-Trees που χρησιμοποιούνται για δεδομένα που μπορούν να κατηγοριοποιηθούν σε ένα άξονα δηλαδή αριθμούς, γράμματα, ημερομηνίες. Τα χωρικά δεδομένα δεν μπορούν να κατηγοριοποιηθούν με αυτό τον τρόπο διότι δεν μπορεί κανείς να πει

εάν κάποιο είναι μεγαλύτερο από κάποιο άλλο οπότε οι δείκτες αυτού του τύπου δεν χρησιμεύουν σε αυτή την περίπτωση

- Δέντρα αναζήτησης τύπου R που χωρίζουν τα δεδομένα σε πολύγωνα και υπό-πολύγωνα και σε περαιτέρω διάσπαση. Αυτού του τύπου τα δέντρα αναζήτησης χρησιμοποιούνται από άλλες βάσεις δεδομένων αλλά η υλοποίησή τους από την PostgreSQL δεν είναι τόσο αποτελεσματική όσο η GiST που θα δούμε παρακάτω.
- Γενικευμένα δέντρα αναζήτησης τύπου GiST (Generalized Search Trees) που χωρίζουν τα δεδομένα με συνθήκες όπως «δεδομένα που επικαλύπτουν» ή «δεδομένα που είναι εντός» και μπορούν να χρησιμοποιηθούν σε μια ευρεία κλίμακα τύπων δεδομένων συμπεριλαμβανομένων και των χωρικών δεδομένων. Το πακέτο PostGIS χρησιμοποιεί δέντρα αναζήτησης τύπου R υλοποιημένα πάνω από δέντρα αναζήτησης τύπου GiST.

Τα γενικευμένα δέντρα αναζήτησης είναι μια γενική μορφή χρήσης δεικτών. Χρησιμοποιούνται για να κάνουν ταχύτερη την αναζήτηση δεδομένων σε μη-κανονικές μορφές δεδομένων. Ένα παράδειγμα δημιουργίας δεικτών φαίνεται παρακάτω :

Create Index Geom_obj_index On Geom Using GiST (g);

Όπου δημιουργούμε δείκτη με όνομα «Geom_obj_index» στην στήλη g του πίνακα Geom.

Η χρήση δεικτών με δέντρα γενικευμένης αναζήτησης έχουν δύο πλεονεκτήματα έναντι σε δέντρα τύπου R. Πρώτον τα δέντρα αυτά μπορούν να χρησιμοποιηθούν σε στήλες γεωμετρικών δεδομένων που μπορεί να έχουν και μηδενικές εγγραφές. Το δεύτερο πλεονέκτημα είναι ότι όταν πρόκειται για δεδομένα με μέγεθος μεγαλύτερο από 8K που είναι το όριο της PostgreSQL με χρήση των δέντρων αυτών αποθηκεύεται μόνο η

πληροφορία που είναι χρήσιμη η οποία είναι το ελάχιστο τετράγωνο του δεδομένου αυτού ενώ αντίστοιχα τα δέντρα τύπου R σε τέτοια περίπτωση αποτυγχάνουν.

Πολύ σημαντικό είναι ότι για να κάνουμε χρήση αυτών των δεικτών σε ένα ερώτημα θα πρέπει να περιλαμβάνεται ο τελεστής ελαχίστου τετραγώνου &&.

8.3 Χρήση Δεικτών στην βάση Oracle

Η Oracle απαιτεί την χρήση δεικτών για πολλά χωρικά ερωτήματα όταν χρησιμοποιούμε τις συναρτήσεις του Locator το οποίο υπενθυμίζουμε είναι το δωρεάν πακέτο της εταιρείας για χειρισμό γεωμετρικών δεδομένων. Η χρήση δεικτών είναι πολύ σημαντική στην Oracle γιατί συμβάλλει καταλυτικά στην απόδοση της βάσης για τα περισσότερα χωρικά ερωτήματα και κυρίως όταν πρόκειται για πληθώρα δεδομένων. Για την χρήση δεικτών σε πίνακα παραθέτουμε στην συνέχεια ένα παράδειγμα με την χρήση της γλώσσας ερωτημάτων SQL. Αρχικά δημιουργούμε δείκτη με όνομα `geom_spatial_idx` στην στήλη `geometry_obj` του πίνακα `geom`.

```
✓ create index geom_spatial_idx  
on geom(geometry_obj)  
indextype is mdsys.spatial_index;
```

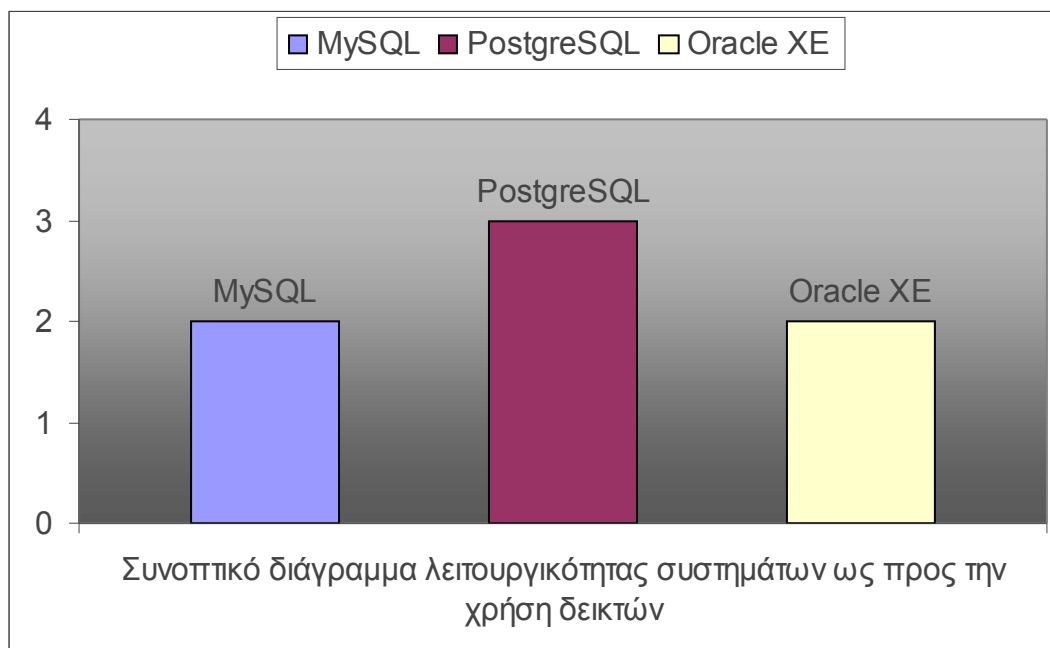
Οι χωρικοί δείκτες όπως και όλοι οι άλλοι δείκτες παρέχουν ένα μηχανισμό για την ελαχιστοποίηση του χρόνου αναζήτησης δεδομένων αλλά στην συγκεκριμένη περίπτωση ο μηχανισμός βασίζεται σε κριτήρια όπως η επικάλυψη ή διασταύρωση. Ένας τέτοιος δείκτης

- Αναζητά αντικείμενα σε δεδομένα στα οποία χρησιμοποιήσαμε δείκτες που αλληλεπιδρούν κατά κάποιον τρόπο με ένα σημείο ή μια περιοχή
- Αναζητά ζεύγη αντικειμένων από δύο διαφορετικούς χώρους δεδομένων στους οποίους χρησιμοποιήσαμε δείκτες οι οποίοι αλληλεπιδρούν χωρικά μεταξύ τους.

Η Oracle χρησιμοποιεί κυρίως δείκτες R δέντρων (R-tree). Ένας τέτοιος δείκτης προσεγγίζει κάθε γεωμετρικό δεδομένο από το ελάχιστο τετράγωνο που εσωκλείει το αντικείμενο.

Είδαμε λοιπόν ότι και τα τρία συστήματα παρέχουν την δυνατότητα χρήσης δεικτών για να μπορούμε να διαχειριζόμαστε μεγάλο όγκο δεδομένων με ταχύτητα. Η MySQL καθώς και η Oracle χρησιμοποιούν δείκτες με την μορφή δέντρων τύπου R (R-tree) όπως είδαμε παραπάνω που κάνουν χρήση το ελάχιστο τετράγωνο του κάθε γεωμετρικού αντικειμένου για να κάνουν τον έλεγχο, άρα μπορούμε να πούμε ότι τα δύο συστήματα αυτά παρουσιάζουν την ίδια λειτουργικότητα. Η PostgreSQL όμως προχωρά ένα βήμα παρακάτω δίνοντας την δυνατότητα να χρησιμοποιήσουμε δέντρα αναζήτησης τύπου R υλοποιημένα πάνω από δέντρα αναζήτησης τύπου GiST τα οποία βελτιώνουν ακόμη περισσότερο την χρήση δεικτών προσφέροντας τα πλεονεκτήματα ότι οι γεωμετρικές στήλες μπορεί να περιέχουν μηδενικές εγγραφές και επίσης χειρίζονται δεδομένα χωρητικότητας άνω των 8K.

Επομένως και πάλι η PostgreSQL είναι αυτή που και στην περίπτωση της χρήσης δεικτών είναι η πιο λειτουργική.



ΚΕΦΑΛΑΙΟ 9

ΣΥΜΠΕΡΑΣΜΑΤΑ

Συνοψίζοντας λοιπόν, όπως αναφέραμε και παραπάνω στην πορεία της ανάλυσης, η σύγκριση των βάσεων δεδομένων ως προς την δυνατότητα χειρισμού χωρικών δεδομένων επιμερίστηκε σε τομείς ώστε να είναι πιο διακριτές οι διαφορές που παρουσιάζουν τα λογισμικά αυτά και να είναι πιο σωστή η σύγκριση διότι γινόταν σε συγκεκριμένο πεδίο κάθε φορά. Στο τέλος κάθε πεδίου ανάλυσης και λαμβάνοντας υπόψη τις δυνατότητες του κάθε λογισμικού κάναμε μια μικρή σύνοψη και τα αποτελέσματα παρουσιάστηκαν σε ένα συνοπτικό διάγραμμα ώστε να μπορούμε εύκολα να διαπιστώσουμε την απόδοση του καθενός σε σχέση με τα υπόλοιπα. Θα αναφέρουμε συνοπτικά για κάθε κεφάλαιο τα βασικά στοιχεία της ανάλυσης που προέκυψαν ώστε να δούμε την λειτουργικότητα των συστημάτων αυτών συνολικά.

➤ **Βασικοί τύποι χωρικών δεδομένων**

Είδαμε ότι και τα τρία συστήματα υποστηρίζουν τους γεωμετρικούς τύπους χωρικών δεδομένων όπως αυτά ορίζονται από τον διεθνή οργανισμό OGC οπότε δεν παρουσιάζουν διαφοροποιήσεις. Υποστηρίζουν στο σύνολο τους όλους τους τύπους γεωμετρικών δεδομένων.

➤ **Δημιουργία και Αποθήκευση Δεδομένων**

Σε αυτό το κεφάλαιο είδαμε αρχικά τον τρόπο που δημιουργούμε τον πίνακα και την κατάλληλη στήλη σε κάθε ένα από τα λογισμικά ώστε στην συνέχεια να δημιουργήσουμε και ταυτόχρονα να αποθηκεύσουμε τα δεδομένα αυτά. Η MySQL και η PostgreSQL ακολουθούν την προτυποποίηση του οργανισμού OGC και παρέχουν πληθώρα συναρτήσεων για την δημιουργία δεδομένων. Άλλες συναρτήσεις δημιουργούν συγκεκριμένο τύπο δεδομένων, άλλες είναι γενικές και μπορούν να δημιουργήσουν γεωμετρικά δεδομένα διάφορων τύπων και όλες

αυτές μπορούν να χρησιμοποιηθούν είτε με βάση την αναπαράσταση κειμένου των δεδομένων είτε με την δυαδική τους μορφή. Η Oracle από την άλλη μεριά έχει τον δικό της τυποποιημένο τρόπο να δημιουργεί τα δεδομένα ουσιαστικά με μία μόνο συνάρτηση ικανή όμως να δημιουργήσει δεδομένα είτε χρησιμοποιώντας αναπαράσταση κειμένου είτε με δυαδική αναπαράσταση είτε με κωδικοποίηση. Οπότε τα δύο λογισμικά σε σχέση με την Oracle προσφέρουν πληθώρα συναρτήσεων οι οποίες είναι πολύ κατανοητές και εύχρηστες, ειδικά αυτές που χρησιμοποιούν την αναπαράσταση κειμένου σε αντίθεση με την Oracle που παρουσιάζει αρκετές δυσκολίες στην χρήση της.

➤ **Μορφές αναπαράστασης των χωρικών δεδομένων**

Τα γεωμετρικά δεδομένα που είναι αποθηκευμένα εσωτερικά στα συστήματα είναι σε τέτοια μορφή που δεν μπορούν να γίνουν κατανοητά. Για αυτό λοιπόν υπάρχουν δύο συναρτήσεις που μετατρέπουν τα δεδομένα αυτά είτε σε μορφή αναπαράστασης κειμένου είτε σε δυαδική μορφή.

Η MySQL και η PostgreSQL υποστηρίζουν τις συναρτήσεις αυτές πλήρως. Στην Oracle όμως και στο δωρεάν πακέτο υποστήριξης χωρικών δεδομένων Oracle Locator, αυτές οι λειτουργίες δεν υποστηρίζονται κάτι που αποτελεί πολύ σημαντικό μειονέκτημα.

➤ **Συναρτήσεις**

Τα αποθηκευμένα δεδομένα σε οποιοδήποτε σύστημα δεν μας είναι χρήσιμα αν δεν υπάρχει τρόπος να τα επεξεργαστούμε και να πάρουμε τις πληροφορίες που θέλουμε. Οι συναρτήσεις που ορίζονται από τον Διεθνή οργανισμό OGC είναι πάρα πολλές και είναι αδύνατο να συγκριθούν τα συστήματα συνολικά. Οπότε χωρίστηκαν οι συναρτήσεις σε κατηγορίες ανάλογα με την χρήση της κάθε μιας για να είναι πιο κατανοητά τα αποτελέσματα της σύγκρισης. Χωρίστηκαν σε τρεις βασικές κατηγορίες

- i. **1^η κατηγορία** : Συναρτήσεις που δέχονται σαν είσοδο ένα γεωμετρικό αντικείμενο και μας δίνουν πληροφορίες για τις ιδιότητες του

- ii. **2^η κατηγορία** : Συναρτήσεις που ελέγχουν την σχέση μεταξύ των γεωμετρικών αντικειμένων
- iii. **3^η κατηγορία** : Συναρτήσεις που υποστηρίζουν χωρική ανάλυση

➤ **1^η Κατηγορία Συναρτήσεων**

Οι συναρτήσεις που επιστρέφουν τις ιδιότητες ενός γεωμετρικού αντικειμένου σύμφωνα με τον διεθνή οργανισμό OGC μπορούν να χωριστούν σε αυτές που ισχύουν για ένα τύπο δεδομένων και σε αυτές που ισχύουν γενικότερα για πολλούς τύπους. Η MySQL και η PostgreSQL ακολουθούν και πάλι τα πρότυπα αυτά με την διαφορά ότι η PostgreSQL υποστηρίζει όλες τις συναρτήσεις ενώ η MySQL σαφώς λιγότερες σε αντίθεση με την Oracle που υποστηρίζει μόνο κάποιες γενικές συναρτήσεις πολύ λιγότερες από τα προηγούμενα λογισμικά.

➤ **2^η Κατηγορία Συναρτήσεων**

Είδαμε σε αυτή την κατηγορία τις συναρτήσεις που ελέγχουν την γεωμετρική σχέση μεταξύ των αντικειμένων. Ας αναφερθούμε και πάλι πρώτα στα δύο λογισμικά που επειδή ακολουθούν σε όλους τους τομείς τα πρότυπα του οργανισμού είναι πιο εύκολο να συγκριθούν. Η MySQL υποστηρίζει αρκετά μεγάλο αριθμό συναρτήσεων αλλά όπως είδαμε παρουσιάζει ένα μεγάλο μειονέκτημα. Οι συναρτήσεις ελέγχουν τα γεωμετρικά δεδομένα με βάση το ελάχιστο τετράγωνο τους πράγμα που δεν μας δίνει μεγάλη ακρίβεια κατά την χρήση τους. Από την άλλη η PostgreSQL υποστηρίζει και πάλι το σύνολο των συναρτήσεων με βάση τα ίδια τα δεδομένα. Τέλος η Oracle παρέχει αρκετές συναρτήσεις που ελέγχουν την σχέση μεταξύ των γεωμετρικών αντικειμένων αλλά σαφώς λιγότερες από την PostgreSQL.

➤ **3^η Κατηγορία Συναρτήσεων**

Σε ότι αφορά συναρτήσεις που χρησιμοποιούνται για χωρική ανάλυση γεωμετρικών δεδομένων είδαμε μόνο τα δύο από τα τρία λογισμικά διότι η MySQL δεν υποστηρίζει συναρτήσεις αυτού του τύπου. Τα άλλα δύο λογισμικά υποστηρίζουν αρκετές, άλλες παρόμοιες και άλλες που μας δίνουν διαφορετικές

δυνατότητες σε κάθε σύστημα. Πάντα όμως ο τρόπος που χρησιμοποιεί η Oracle παρουσιάζει μεγαλύτερες δυσκολίες στην χρήση και κατανόηση των συναρτήσεων όπως είδαμε και στα παραδείγματα.

➤ Χρήση Δεικτών

Σε αυτό το κεφάλαιο μιλήσαμε για την σπουδαιότητα της χρήσης δεικτών στα συστήματα διαχείρισης βάσεων δεδομένων και το πόσο διαφορετικά συμπεριφέρεται το σύστημα όταν πρόκειται για μεγάλο όγκο δεδομένων. Όπως είδαμε και τα τρία συστήματα δίνουν την δυνατότητα χρήσης δεικτών σε γεωμετρικά δεδομένα με την χρήση δέντρων αναζήτησης R (R tree). Η PostgreSQL όμως προχωρά ένα βήμα παραπάνω από τις υπόλοιπες και δίνει την δυνατότητα να χρησιμοποιήσουμε δέντρα αναζήτησης τύπου R υλοποιημένα πάνω από δέντρα αναζήτησης τύπου GiST με καλύτερα αποτελέσματα.

Μετά την σύντομη αναδρομή σε αυτά που συναντήσαμε σε κάθε κεφάλαιο θα χρησιμοποιήσουμε τα διαγράμματα που παραθέσαμε στο τέλος κάθε κεφαλαίου ώστε να συνοψίσουμε τα αποτελέσματα όλων των επιμέρους κεφαλαίων σε ένα συγκεντρωτικό διάγραμμα και να δούμε συνολικά πως αυτά μπορούν να βοηθήσουν τον χρήστη στην επεξεργασία χωρικών δεδομένων.

Από τα επί μέρους λοιπόν διαγράμματα που έδιναν μια εικόνα της απόδοσης των συστημάτων σε κάθε κεφάλαιο προκύπτει το διάγραμμα που φαίνεται παρακάτω και αποτελεί μια γενική εικόνα όσων αναλύσαμε στα παραπάνω κεφάλαια. Να θυμίσουμε όμως ότι η κλίμακα που χρησιμοποιήθηκε στα διαγράμματα σε κάθε κεφάλαιο δεν είναι τίποτε άλλο από μια κατηγοριοποίηση των συστημάτων σε σχέση με την λειτουργικότητα τους.

Ο αριθμός ένα σημαίνει πολύ χαμηλή λειτουργικότητα του συγκεκριμένου συστήματος διότι παρέχει τις πολύ βασικές λειτουργίες σε σχέση με τα υπόλοιπα.

Ο αριθμός δύο αντιστοιχεί σε ένα σύστημα που παρέχει αρκετές λειτουργίες αλλά δεν είναι πλήρες

Ο αριθμός τρία σημαίνει ότι το σύστημα παρέχει όλες τις δυνατότητες που συνήθως ορίζονται σε σχέση με τον Διεθνή οργανισμό OGC, πάντα σε σχέση με την διαχείριση γεωμετρικών δεδομένων.

	Βασικοί Τύποι Χωρικών Δεδομένων	Δημιουργία και Αποθήκευση Δεδομένων	Μορφές αναπαράστ. χωρικών δεδομένων	1η Κατηγορία Συναρτήσεων	2η Κατηγορία Συναρτήσεων	3η Κατηγορία Συναρτήσεων	Χρήση Δεικτών
ORACLE EXPRESS 10G	3	2	1	1	2	2	2
PostgreSQL 8.2	3	3	3	3	3	2	3
MySQL 5.0	3	3	3	2	1	-	2

Βλέπουμε λοιπόν ότι από όλες τις απόψεις η PostgreSQL παρουσιάζεται ως το πιο πλήρες σύστημα, με τις περισσότερες λειτουργίες από τα υπόλοιπα σε όλους τους τομείς.

Το μεγάλο του πλεονέκτημα είναι ότι το πρόσθετο πακέτο που εγκαθιστούμε και του δίνει την δυνατότητα να χειριστεί χωρικά δεδομένα (PostGIS) και υποστηρίζεται από τον οργανισμό Refractions Research εξελίσσεται συνεχώς και υπάρχει η δυνατότητα να εγκαθίσταται σαν ξεχωριστό πακέτο ώστε να μην χρειάζεται εγκαθιστούμε όλο το σύστημα διαχείρισης χωρικών δεδομένων για να εκμεταλλευτούμε τις νέες λειτουργίες των νεότερων εκδόσεων Επίσης μην ξεχνάμε ότι αποτελεί ανοιχτού κώδικα ελεύθερο λογισμικό που είναι τεράστιο πλεονέκτημα. Από την άλλη η MySQL είναι πολύ απλή και παρέχει πολύ βασικές λειτουργίες σε σχέση με την PostgreSQL αλλά και αυτή είναι ανοιχτού κώδικα ελεύθερο λογισμικό και μπορεί να χρησιμοποιηθεί από χρήστες που θέλουν να έρθουν σε επαφή και να πειραματιστούν με τα συστήματα αυτά που προσφέρουν την δυνατότητα διαχείρισης χωρικών δεδομένων. Μπορεί να έχει

περιορισμένες δυνατότητες σε σχέση με την PostgreSQL αλλά είναι πολύ εύχρηστη και απλή. Τέλος η συγκεκριμένη έκδοση της Oracle η οποία διατίθεται δωρεάν προσφέρει πολύ λιγότερες δυνατότητες από το πλήρες πακέτο που παρέχει η εταιρεία και μπορεί να χρησιμοποιηθεί από χρήστες ως προς τις βασικές λειτουργίες της και τον τρόπο που χειρίζεται τα δεδομένα αυτά ώστε να προχωρήσει στην αγορά του πλήρους πακέτου. Να θυμηθούμε ότι το πακέτο Oracle Locator που διατίθεται δωρεάν μαζί με το Σύστημα Διαχείρισης Βάσεων Δεδομένων της Oracle παρέχει ελάχιστες δυνατότητες σε σχέση με το πακέτο Oracle Spatial που διατίθεται με την πλήρη έκδοση. Οπότε το Σύστημα Διαχείρισης Χωρικών Δεδομένων της PostgreSQL είναι μακράν το πιο πλήρες πακέτο ενώ τα υπόλοιπα δύο όπως είπαμε ακολουθούν και θεωρούμε ότι μπορούν να χρησιμοποιηθούν από χρήστες είτε με λιγότερες απαιτήσεις είτε όσον αφορά το πακέτο της Oracle για αξιολόγηση ώστε να προχωρήσουν στην αγορά του πλήρους πακέτου.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. <http://www.mysql.com/> Ο επίσημος δικτυακός τόπος του Συστήματος Διαχείρισης Βάσεων Δεδομένων της MySQL. Διατίθενται στους χρήστες διάφορες εκδόσεις του λογισμικού, πολλά εγχειρίδια χρήσης και πολλά άρθρα. Επίσης υπάρχει φόρουμ επίλυσης προβλημάτων.
2. <http://www.postgresql.org/> Ο επίσημος δικτυακός τόπος του Συστήματος Διαχείρισης Βάσεων Δεδομένων της PostgreSQL με πλήρη υποστήριξη στον χρήστη όσον αφορά λογισμικό, υποστήριξη και εγχειρίδια χρήσης
3. <http://www.oracle.com/> Ο επίσημος δικτυακός τόπος του Συστήματος Διαχείρισης Βάσεων Δεδομένων της Oracle

4. <http://www.postgis.org/> Αποτελεί τον δικτυακό τόπο του λογισμικού PostGIS το οποίο δίνει την δυνατότητα στο Σύστημα Διαχείρισης Βάσεων Δεδομένων PostgreSQL να χειρίζεται χωρικά δεδομένα. Το λογισμικό αναπτύχθηκε από τον οργανισμό Refractions Research.