Ανάπτυξη Μεθόδων Χωρο-κειμενικής ευρετηρίασης σε Μη-σχεσιακές Βάσεις Δεδομένων

Κωνσταντίνος Νεστοράκης

Επιβλέπων καθηγητής: Χρήστος Δουλκερίδης

Μεταπτυχιακό πρόγραμμα “Πληροφοριακά Συστήματα και Υπηρεσίες”

Ειδίκευση “Μεγάλα Δεδομένα και Αναλυτική”

Τμήμα Ψηφιακών Συστημάτων

Πανεπιστήμιο Πειραιώς

Διπλωματική Εργασία

Φεβρουάριος 2021

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου Χρήστο Δουλκερίδη. Είχα την ευκαιρία στα πλαίσια του μεταπτυχιακού προγράμματος και κατά την διεκπεραίωση της διπλωματικής μου εργασίας να εισπράξω ανεκτίμητες συμβουλές και γνώσεις υπό τις οδηγίες του.

Επίσης, θα ήθελα να ευχαριστήσω τον υποψήφιο διδάκτορα Νικόλαο Κουτρουμάνη για την βοήθεια που μου έδωσε σε επίπεδο κώδικα, καθώς και για τις δημιουργικές συζητήσεις που είχαμε κατά την διάρκεια της διπλωματικής μου εργασίας. Θα ήθελα να κάνω ειδική αναφορά στον αναπληρωτή καθηγητή Μιχαήλ Φιλιππάκη. Μέσω της διδασκαλίας του και μέσω των συζητήσεων που κάναμε, με βοήθησε να κατανοήσω καλύτερα τεχνικές και αλγόριθμους μηχανικής μάθησης.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια που με στήριξε για να φέρω εις πέρας το συγκεκριμένο μεταπτυχιακό πρόγραμμα.

Περίληψη

Στις μέρες μας, υπάρχουν πολλές εταιρείες πληροφορικής που χρησιμοποιούν GPS υπηρεσίες στα προϊόντα που προσφέρουν στους πελάτες τους. Το πιο χαρακτηριστικό παράδειγμα είναι εταιρεία Google με την εφαρμογή Google maps. Η βασική υπηρεσία της εφαρμογής, είναι να παρέχει οδηγίες μεταφοράς στους χρήστες της, από ένα γεωγραφικό στίγμα σε ένα άλλο. Επίσης, παραδείγματα είναι οι εταιρείες Uber και Beat που προσφέρουν υπηρεσίες οδικής μεταφοράς των πελατών τους, από ένα σημείο σταθμό σε κάποιον προορισμό. Οι συγκεκριμένες εφαρμογές κατά την εκτέλεση τους επεξεργάζονται γεωγραφικά δεδομένα. Ένα απλό ερώτημα που μπορεί να εκτελέσει η εφαρμογή Google maps είναι μια αναζήτηση των πλησιέστερων cafe από ένα σημείο στίγματος, Αντίστοιχα, για τις εφαρμογές Uber και Beat, ο πελάτης μπορεί να θέλει να επιλέξει το φύλλο του οδηγού που θα εκτελέσει την οδική μεταφορά π.χ. να είναι γυναίκα. Εύκολα μπορεί να αντιληφθεί κάποιος ότι οι συγκεκριμένες εφαρμογές πέρα από γεωγραφικά δεδομένα, επεξεργάζονται και κειμενικά δεδομένα. Φυσικά, επεξεργάζονται και χρονική πληροφορία αλλά αυτό δεν θα εξεταστεί στη συγκεκριμένη διπλωματική εργασία.

Η αποθήκευση και η επεξεργασία όλων αυτών των δεδομένων, από τις συγκεκριμένες εφαρμογές που αναφέρθηκαν στην προηγούμενη παράγραφο, απαιτούν βάσεις δεδομένων που προσφέρουν υψηλή απόδοση (performance), διαθεσιμότητα (availability) και επεκτασιμότητα (scalability). Ένα Σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (Relational Database Management System) δεν μπορεί να καλύψει τις συγκεκριμένες ανάγκες. Σε αντίθεση με τις NoSQL βάσεις δεδομένων όπου η χρήση τους, ενδείκνυται για τέτοιου είδους δεδομένα. Η ιδιαιτερότητα των NoSQL βάσεων δεδομένων, είναι ότι δεν διαθέτουν όλες απευθείας χωρική ή χωρο-κειμενική ευρετηρίαση, αλλά παρέχουν τεχνικές που μπορούν να υποστηρίξουν τέτοιου είδους δυνατότητες.

Σε αυτή την διπλωματική, θα παρουσιαστούν τεχνικές ευρετηρίασης πάνω σε χωρο-κειμενικά δεδομένα. Αυτές οι τεχνικές θα υλοποιηθούν πάνω στο NoDA API [1] για MongoDB και HBase Stores, επεκτείνοντας προηγούμενη δουλεία της Big Data ερευνητικής ομάδας του Πανεπιστημίου Πειραιώς, πάνω σε χωρο-κειμενικά δεδομένα. Το NoDA API είναι ένα ενδιάμεσο επίπεδο ανάμεσα στην εφαρμογή και τα NoSQL Stores υποστηρίζοντας χωρο-χρονικές και χωρο-κειμενικές τεχνικές ευρετηρίασης.

Abstract

Nowadays, many IT companies provide GPS services and products to their customers. The most distinctive example is Google via the Google maps app. The app's fundamental service is to provide its users with directions from one point to another. The companies Uber and Beat also have apps to provide their customers transport services from one start point to any destination. One simple query that the Google maps app supports is the search of nearest cafes from a specific location. Uber and Beat give their customers the option to select the driver's sex before transport. The conclusion is that all apps mentioned above can process both spatial and textual operating queries. In addition to spatial and textual data processing, they can process temporal data, but this case will not be examined in this study.

All apps in the previous paragraph store and process their data using Data Bases with the following specific features: high performance, availability and scalability. A Relational Database Management System (RDBMS) cannot cover all these needs, except for NoSQL Stores. However, some NoSQL Stores do not support direct spatial or spatiotextual indexing, even though they do have some techniques to support this issue.

This study will present spatio-textual techniques implemented on NoDA API for MongoDB and Hbase Stores. The NoDA API is an abstract layer between an app and NoSQL Stores, providing one query language and supporting spatial, spatio-temporal and spatio-textual indexing.

Περιεχόμενα

Λίστα Εικόνων

Λίστα Πινάκων

Κεφάλαιο 1

1. Εισαγωγή

Στην εποχή μας, όλο και περισσότεροι άνθρωποι χρησιμοποιούν την λειτουργία GPS που διαθέτουν οι mobile συσκευές τους, για να καλύψουν ανάγκες τους. Αυτό σημαίνει, ότι παράγεται ένας τεράστιος όγκος δεδομένων με γεωγραφική πληροφορία. Παράλληλα, στον κόσμο μας υπάρχει και ένα ασύλληπτα μεγάλο μέγεθος κειμενικής πληροφορίας. Για παράδειγμα, για την κατηγορία φαγητό υπάρχουν πολλές λέξεις που μπορεί κάποιος να εντοπίσει, Burger, Pasta, Mediterranean κ.τ.λ. Ο συνδυασμός της γεωγραφικής πληροφορίας με την κειμενική, παράγει τα χωρο-κειμενικά ερωτήματα. Ένα απλό χωρο-κειμενικό ερώτημα, είναι να γίνει αναζήτηση της λέξης Burger σε μια συγκεκριμένη γεωγραφική περιοχή. Η απαίτηση που δημιουργείται, είναι η απάντηση τέτοιου είδους ερωτημάτων διατηρώντας υψηλή απόδοση. Δεν θα ήταν λειτουργικό κάποιος να πραγματοποιήσει ένα χωρο-κειμενικό ερώτημα και η απάντηση να έρθει με μεγάλη καθυστέρηση. Είναι πολλές οι προκλήσεις που αντιμετωπίζουν τα συστήματα διαχείρισης δεδομένων καθώς τα χωρο-κειμενικά δεδομένα αυξάνονται με τεράστια ταχύτητα. Για αυτό τον λόγο τα παραδοσιακά σχεσιακά συστήματα διαχείρισης δεδομένων αδυνατούν να διαχειριστούν χωρο-κειμενικά δεδομένα.

Τα NoSQL stores επιτρέπουν την διαχείριση μεγάλου όγκου δεδομένων καθώς παρέχουν ευέλικτη κατανεμημένη αποθήκευση των δεδομένων και είναι κατάλληλα για να υποστηρίξουν χωρο-κειμενικά ερωτήματα. Ωστόσο, κάποια από αυτά δεν υποστηρίζουν απευθείας χωρική ή χωρο-κειμενική ευρετηρίαση, αλλά παρέχουν τεχνικές που μπορούν να υποστηρίξουν τέτοιου είδους δυνατότητες.

Στην διπλωματική εργασία θα εφαρμοστούν και θα αναλυθούν τέτοιου είδους τεχνικές για τα MongoDB και Hbase stores. Οι συγκεκριμένες εφαρμογές θα ενσωματωθούν στο NoDA API, προσφέροντας σε έναν προγραμματιστή την δυνατότητα να εκτελέσει χωρο-κειμενικά ανεξάρτητα με την γλώσσα ερωτημάτων που διαθέτει το NoSQL store που θέλει να χρησιμοποιήσει. Ό τύπος των χωρο-κειμενικών ερωτημάτων που θα χρησιμοποιηθούν, θα είναι boolean range queries (BRQ). Τα ερωτήματα τύπου BRQ αναλύονται στην ενότητα 2.1.1.

1.1 Σκοπός διπλωματικής εργασίας

Στόχοι της διπλωματικής εργασίας είναι να παρουσιαστούν και αναλυθούν, βέλτιστες τεχνικές χωρο-κειμενικής ευρετηρίασης για τα MongoDB και HBase stores. Να δοθεί έμφαση στις ιδιαιτερότητες και στο μοντέλο δεδομένων του κάθε store. Να αναφερθούν οι τελεστές που διαθέτουν τα συγκεκριμένα stores για την υποστήριξη χωρο-κειμενικών ερωτημάτων. Να εξεταστεί η αποδοτικότητα των παραπάνω συστημάτων και η επεκτασιμότητα που διαθέτουν σε χωρο-κειμενικά δεδομένα. Τέλος, να πραγματοποιηθεί επέκταση του NoDA API για χωρο-κειμενικά ερωτήματα.

1.2 Δομή διπλωματικής εργασίας

Η δομή της εργασίας είναι ακόλουθη:

- Στο Κεφάλαιο 2 θα αναφερθούν οι βασικοί τύποι χωρικών και χωρο-κειμενικών ερωτημάτων, καθώς τεχνικές ευρετηρίασης που μπορούν χρησιμοποιηθούν για το χωρικό και κειμενικό μέρος. Για κάθε τύπο ερωτήματος θα δοθούν παραδείγματα.

- Στο Κεφάλαιο 3 θα πραγματοποιηθεί περιγραφή των NoSQL stores και θα αναφερθούν πλεονεκτήματα-μειονεκτήματα έναντι των SQL stores. Επίσης, θα αναλυθούν τα NoSQL stores MongoDB και Hbase. Θα αναφερθούν τα βασικά χαρακτηρίστηκα του κάθε store καθώς και οι ιδιαιτερότητές τους. Θα αναλυθεί το μοντέλο δεδομένων που παρέχει το κάθε data store, καθώς και οι τελεστές που διαθέτουν για την πραγματοποίηση χωρικής και χωρο-κειμενικής ευρετηρίασης.

- Στο Κεφάλαιο 4 θα πραγματοποιηθεί επεξήγησή της μεθοδολογίας και τεχνικών που χρησιμοποιήθηκαν. Συγκεκριμένα, θα επεξηγηθεί το πως μπορούν να χρησιμοποιηθούν τεχνικές καμπύλης πλήρωσης χώρου σε συνδυασμό με κειμενικές λίστες θέσης, ώστε να πραγματοποιηθεί μια χωρο-κειμενική ευρετηρίαση.

- Στο Κεφάλαιο 5 θα δοθεί η εφαρμογή της μεθοδολογίας που αναπτύχθηκε με την πειραματική διαδικασία που εφαρμόστηκε. Συγκεκριμένα, η πειραματική διαδικασία εκτελέστηκε σε κεντρικοποιημένο και κατανεμημένο περιβάλλον για το MongoDB store και για το HBase store εκτελέστηκε μόνο σε κεντρικοποιημένο περιβάλλον. Θα παρουσιαστούν τα ερωτήματα που εκτελέστηκαν, καθώς θα δοθούν και συγκριτικά αποτελέσματα.

- Στο Κεφάλαιο 6 θα δοθούν τα συμπεράσματα της διπλωματικής εργασίας. Καθώς, θα παρατεθούν ιδέες για το πως μπορεί να συνεχιστεί η συγκεκριμένη μελέτη σε επόμενο στάδιο.

Κεφάλαιο 2

2. Ανασκόπηση χωρο-κειμενικών ερωτημάτων

Οι βασικοί τύποι χωρο-κειμενικών ερωτημάτων είναι οι εξής:

- Boolean range queries (BRQ).
- Boolean knn queries (BNQ).
- Top-k queries (TkQ).
- Join queries (JQ).

Έστω D ένα χωρο-κειμενικό σύνολο δεδομένων. Το o.ID δηλώνει το object ID. Για κάθε αντικείμενο o ∈ D ορίζεται ένα ζευγάρι (o.geo o.keyword). Το o.geo συμβολίζει τις γεωγραφικές συντεταγμένες του object (lon,lat) και το o.keyword συμβολίζει τα keywords του αντικείμενου o. Το o.geo αποτελεί την χωρική πληροφορία ενός αντικειμένου. Ένα ζευγάρι (o.geo, o.keyword) αποτελεί ένα χωρο-κειμενικό αντικείμενο.

Σε αυτό το κεφάλαιο θα γίνει αναφορά στους τύπους χωρικών και χωρο-κειμενικών ερωτημάτων και θα αναφερθούν τεχνικές ευρετηρίασης.

2.1 Τύποι χωρο-κειμενικών ερωτημάτων

2.1.1 Boolean range queries

Για να πραγματοποιηθεί ένα boolean range query, απαιτείται αρχικά να καθοριστεί ένα γεωμετρικό σχήμα εντός μιας συγκεκριμένης περιοχής. Αυτό μπορεί να είναι ένα ορθογώνιο, ένας κύκλος ή ένα πολύγωνο. Στην συνέχεια ορίζονται οι λέξεις προς αναζήτηση, καθώς και ο τελεστής αναζήτησης που μπορεί να είναι AND ή OR. Το boolean range query, θα επιστρέψει τα αντικείμενα που περιέχουν τις λέξεις κλειδιά βάσει ενός ορισμένου τελεστή, τα οποία βρίσκονται εντός του γεωμετρικού σχήματος αναζήτησης που περιέχει το ερώτημα [2].

Έστω ότι ορίζεται η γεωγραφική περιοχή της Αττικής. Για το boolean range query, χρησιμοποιείται ένα ορθογώνιο γεωμετρικό σχήμα όπως φαίνεται στην Εικόνα 2.1. Εντός του ορθογωνίου, υπάρχουν 4 χωρο-κειμενικά αντικείμενα, το o1, o2, o3 και το o4. Το ερώτημα που πραγματοποιείται είναι να βρεθούν τα χωρο-κειμενικά αντικείμενα που βρίσκονται εντός του ορθογωνίου και περιέχουν τουλάχιστον μια από τις λέξεις cafe ή garden.

Εικόνα 2.1 Boolean range query

Σε αυτή την περίπτωση η αναζήτηση των λέξεων θα γίνει βάσει του τελεστή OR. Τα χωρο-κειμενικά αντικείμενα που θα επιτρέψει το συγκεκριμένο ερώτημα είναι τα o1, o3 και o4. Στην περίπτωση που το ερώτημα ήταν, να βρεθούν τα χωρο-κειμενικά αντικείμενα που βρίσκονται εντός του ορθογωνίου και περιέχουν τις λέξεις cafe και garden τότε, η αναζήτηση των λέξεων θα πραγματοποιηθεί βάσει του τελεστή AND. Αυτό το ερώτημα δεν θα επιστρέψει κανένα αντικείμενο.

2.1.2 Boolean knn queries

Ένα boolean knn query αναζητεί τους πλησιέστερους γείτονες ενός χωρο-κειμενικού αντικειμένου βάσει ορισμένης ακτίνας εντός μιας συγκεκριμένης γεωγραφικής περιοχής, ικανοποιώντας τους τελεστές AND ή OR για την κειμενική αναζήτηση [3].

Έστω ότι ορίζεται η γεωγραφική περιοχή της Αττικής. Επιπλέον ορίζεται η απόσταση r=2.9 km Εικόνα 2.2. Το ερώτημα που πραγματοποιείται είναι, να βρεθούν τα χωρο-κειμενικά αντικείμενα που περιέχουν τουλάχιστον μια από τις λέξεις cafe ή garden και απέχουν το πολύ 2,9 km από το o2. Σε αυτή την περίπτωση η αναζήτηση των λέξεων θα γίνει βάσει του τελεστή OR.

Εικόνα 2.2 Boolean knn query

Τα χωρο-κειμενικά αντικείμενα που θα επιτρέψει το ερώτημα με συγκεκριμένη σειρά είναι, τα o1 και o3. Στο ερώτημα, να βρεθούν τα χωρο-κειμενικά αντικείμενα που περιέχουν τις λέξεις cafe και garden και απέχουν το πολύ 2,9 km από το o2. Η αναζήτηση των λέξεων, θα πραγματοποιηθεί βάσει του τελεστή AND. Αυτό το ερώτημα δεν θα επιστρέψει κανένα αντικείμενο.

2.1.3 Top-k queries

Το χαρακτηριστικό των top-k queries, είναι ότι λαμβάνουν υπόψιν τους και την κειμενική ομοιότητα (text similarity) των λέξεων κατά την διαδικασία αναζήτησης. Μια από τις πιο γνωστές μεθόδους που χρησιμοποιείται στα top-k queries είναι, η μέθοδος Jaccard [4]. Η μέθοδος Jaccard ορίζεται ως μια τομή δύο αντικειμένων η οποία διαιρείται με την ένωση αυτών των δύο αντικειμένων λαμβάνοντας υπόψιν τον αριθμό των κοινών λέξεων σε σχέση με τον συνολικό αριθμό λέξεων που έχουν. Ο τύπος της μεθόδου Jaccard είναι $J(o1.\text{keyword1}, o2.\text{keyword2}) = \frac{o1.\text{keyword1} \cap o.2\text{keyword2}}{o1.\text{keyword1} \cup o.2\text{keyword2}}$.

Έστω ότι ορίζεται η γεωγραφική περιοχή της Αττικής. Το ερώτημα που πραγματοποιείται είναι, να βρεθούν τα top-2 χωρο-κειμενικά αντικείμενα εντός της Αττικής που περιέχουν την λέξη cafe Εικόνα 2.3.

Εικόνα 2.3 Top-k query

Τα αποτελέσματα που θα επιστρέψει το ερώτημα είναι, το o1 και o4. Σε περίπτωση που έπρεπε να βρεθούν τα top-3 χωρο-κειμενικά αντικείμενα εντός της Αττικής που

περιέχουν την λέξη cafe. Θα επέστεφαν πρώτα τα o1, o4 και στη συνέχεια το o3 με μικρότερο score.

2.1.4 Join queries

Για να σχηματιστούν τα join queries, χρειάζεται να υπάρχουν τουλάχιστον 2 κατηγορίες από χωρικά ή χωρο-κειμενικά. Το join query θα επιστρέψει τα ζευγάρια αντικειμένων. Για παράδειγμα μία κατηγορία μπορεί να απαρτίζεται από χωρο-κειμενικά αντικείμενα που αναπαριστούν ξενοδοχεία και η άλλη κατηγορία να απαρτίζεται από χωρο-κειμενικά αντικείμενα που αναπαριστούν εστιατόρια [5].

Ένα να αντιπροσωπευτικό join query είναι να βρεθούν τα εστιατόρια που απέχουν από τα ξενοδοχεία μέχρι 2,9 km και προσφέρουν ιταλικό φαγητό Εικόνα 2.4. Θα ανακτηθούν τα ζεύγη που είναι εντός της 2,9 km και που οι λέξεις τους έχουν ομοιότητα πάνω από θ.

Εικόνα 2.4 Join query

2.2 Τεχνικές χωρικής και κειμενικής ευρετηρίασης

Για την ευρετηρίαση σε χωρικά δεδομένα. Υπάρχουν 3 βασικές κατηγορίες [6]:

- **R-tree based indices:** Βασίζεται στη δημιουργία δέντρων που ανήκουν στην οικογένεια R-tree. Συνήθως, όταν χρησιμοποιείται η συγκεκριμένη κατηγορία για την χωρική ευρετηρίαση, συνδυάζεται με αντεστραμμένα αρχεία (inverted files) για την ευρετηρίαση της κειμενικής πληροφορίας. Η οργάνωση της χωρικής και κειμενικής πληροφορίας γίνεται ξεχωριστά.

- **Grid based indices:** Χωρίζει τον χώρο σε τετράγωνα ή ορθογώνια ίσου μεγέθους. Όπως στην προηγούμενη περίπτωση, έτσι και σε αυτήν για την κειμενική ευρετηρίαση, συνήθως χρησιμοποιούνται inverted files. Η οργάνωση της χωρικής και κειμενικής πληροφορίας και σε αυτήν την περίπτωση γίνεται ξεχωριστά.

- **Space filling curve (SFC) based indices:** Οι πιο γνωστές τεχνικές SFC είναι η Z-curve και η Hilbert curve. Κάθε σημείο x,y στον χώρο που έχει οριστεί, να αναπαρίσταται με 1 μονοδιάστατη τιμή που παράγουν αυτές οι

τεχνικές. Αυτή η τιμή μπορεί να συνδυαστεί με inverted files για να δημιουργηθεί η χωρο-κειμενική ευρετηρίαση.

Παρακάτω βασικές τεχνικές κειμενικής ευρετηρίασης [7][8].

- **Inverted File:** Ένα inverted file αποτελείται από ένα λεξικό από όρους. Ο κάθε όρος σχετίζεται με ένα inverted list. Κάθε inverted list περιλαμβάνει εγγραφές όπου η κάθε εγγραφή περιέχει ένα oID που είναι το object ID και ένα o.text (o.keyword). Για κάθε όρο που περιέχεται μέσα στο o.text, αναγράφεται η συχνότητα του στο o.text. Τις περισσότερες φορές μέσα στο inverted list γίνεται ταξινόμηση βάσει του object ID.
- **PostList:** Στην περίπτωσή που χρησιμοποιηθεί SFC ευρετηρίαση, οι συντεταγμένες των χωρο-κειμενικών αντικειμένων αναπαριστώνται με μια μονοδιάστατη τιμή. Σε αυτήν την μονοδιάστατη τιμή προστίθεται στο τέλος η λέξη που περιέχεται στο χωρο-κειμενικό αντικείμενο.

2.2.1 Ευρετηρίαση καμπύλης πλήρωσης χώρου

Η συγκεκριμένη μελέτη έχει βασιστεί στην μέθοδο καμπύλης πλήρωσης χώρου (Space Filling Curve). Το βασικό χαρακτηριστικό της μεθόδου SFC είναι ότι μετατρέπει δεδομένα πολλών διαστάσεων σε 1 διάσταση. Με αυτόν τον τρόπο έχει δοθεί λύση σε NoSQL stores που δεν υποστηρίζουν απευθείας χωρική ή χωρο-κειμενική ευρετηρίαση [1]. Βασικές τεχνικές που χρησιμοποιούνται για την υλοποίηση της μεθόδου SFC είναι οι ακόλουθες:

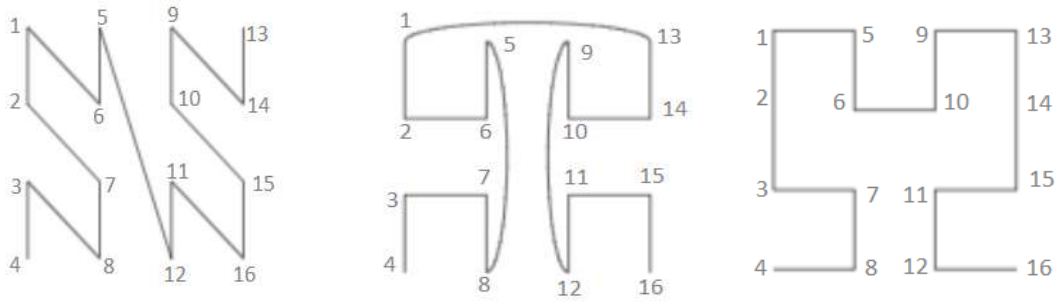
- Z curve (Z-order).
- RGB curve.
- Hilbert curve.

Για την επεξήγηση των παραπάνω τεχνικών δίνεται ο Πίνακας 2.1 που αναπαριστά κάποιες περιοχές στον χώρο.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

Πίνακας 2.1 Διακριτές τιμές στο χώρο

Στην Εικόνα 2.6 παρουσιάζεται πως πραγματοποιείται θα γίνει η διάσχιση του Πίνακα 2.1 για τους αλγορίθμους Z curve, RGB curve, Hilbert curve.



Εικόνα 2.5 Z curve, RGB curve, Hilbert curve

Δίνεται ένα παράδειγμα με το πως παράγονται τα z-values για την μέθοδο Z curve. Ο Πίνακας 2.2 περιέχει τα z-values για τις συντεταγμένες x,y με $0 \leq x \leq 7$, $0 \leq y \leq 7$. Τα z-values δημιουργούνται με χρήση δυαδικής αναπαράστασης.

	x:000	**x:001**	**x:010**	**x:011**	**x:100**	**x:101**	**x:110**	**x:111**
y: 000	0**00**0**0**	0**00**0**1**	0**00**1**0**	0**00**1**1**	0**10**0**0**	0**10**0**1**	0**10**1**0**	**0**1**0**101
y: 001	0**00**0**1**0	0**00**0**11**	0**00**1**10**	0**00**1**11**	0**10**0**10**	0**10**0**11**	0**10**1**10**	**0**1**0**111
y: 010	0**01**0**0**0	0**01**0**01**	0**01**1**00**	0**01**1**01**	0**11**0**00**	0**11**0**01**	0**11**1**00**	**0**1**1**101
y: 011	0**01**0**10**	0**01**0**11**	0**01**1**10**	0**01**1**11**	0**11**0**10**	0**11**0**11**	0**11**1**10**	**0**1**1**111
y: 100	1**00**0**0**0	1**00**0**01**	1**00**1**00**	1**00**1**01**	1**10**0**00**	1**10**0**01**	1**10**1**00**	**1**1**0**101
y: 101	1**00**0**10**	1**01**0**11**	1**00**1**10**	1**00**1**11**	1**10**0**10**	1**10**0**11**	1**10**1**10**	**1**1**0**111
y: 110	1**01**0**0**0	1**01**0**01**	1**01**1**00**	1**01**1**01**	1**11**0**00**	1**11**0**01**	1**11**1**00**	**1**1**1**101
y: 111	1**01**0**10**	1**01**0**11**	1**01**1**10**	1**01**1**11**	1**11**0**10**	1**11**0**11**	1**11**1**10**	**1**1**1**111

Πίνακας 2.2 Z-values

Κεφάλαιο 3

3. Ανασκόπηση NoSQL λύσεων για χωρο-κειμενικά δεδομένα

Οι βάσεις δεδομένων NoSQL εμφανίστηκαν στα τέλη της δεκαετίας του 2000 καθώς το κόστος αποθήκευσης μειώθηκε δραματικά. Πέρασαν οι μέρες που χρειάζονταν να δημιουργηθεί ένα περίπλοκο, δύσκολο στη διαχείριση μοντέλο δεδομένων απλώς και μόνο με σκοπό τη μείωση της αναπαραγωγής δεδομένων. Οι προγραμματιστές (και όχι ο χώρος αποθήκευσης) γινόταν το κύριο κόστος ανάπτυξης λογισμικού, έτσι οι βάσεις δεδομένων NoSQL βελτιστοποιήθηκαν για την παραγωγικότητα του προγραμματιστή. Καθώς το κόστος αποθήκευσης μειώθηκε γρήγορα, ο αριθμός των εφαρμογών δεδομένων που απαιτούνται για την αποθήκευση και το ερώτημα αυξήθηκε. Αυτά τα δεδομένα ήρθαν σε όλα τα σχήματα και μεγέθη - δομημένα, ημιδομημένα και πολυμορφικά - και ο ορισμός του σχήματος εκ των προτέρων έγινε σχεδόν αδύνατος. Οι βάσεις δεδομένων NoSQL επιτρέπουν στους προγραμματιστές να αποθηκεύουν τεράστιες ποσότητες μη δομημένων δεδομένων, δίνοντάς τους μεγάλη ευελιξία.

Σε αυτό το κεφάλαιο θα γίνει ανασκόπηση των NoSQL stores και θα αναλυθούν τα MongoDB και HBase stores για το πως μπορούν να διαχειριστούν την χωρική και κειμενική πληροφορία.

Πολλοί άνθρωποι χρησιμοποιούν τον όρο NoSQL βάσεις δεδομένων για να αναφερθούν σε μια μη σχεσιακή βάση δεδομένων. Μερικοί χρησιμοποιούν τον όρο

NoSQL εννοώντας “non SQL” και άλλοι εννοώντας “not only SQL.”. Η αλήθεια είναι ότι οι NoSQL βάσεις δεδομένων αποθηκεύουν τα δεδομένα σε διαφορετική μορφή από τους σχεσιακούς πίνακες που χρησιμοποιεί μια σχεσιακή βάση δεδομένων. Υπάρχει η παρανόηση ότι οι NoSQL βάσεις δεδομένων δεν αποθηκεύουν βέλτιστα δεδομένα με συσχετίσεις. Αυτό δεν ισχύει. Οι NoSQL βάσεις δεδομένων αποθηκεύουν τέτοιου δεδομένα απλά με διαφορετικό τρόπο από τις σχεσιακές βάσεις δεδομένων. Στην πραγματικότητα ίσως είναι ευκολότερο να αποθηκευτούν δεδομένα με συσχετίσεις σε NoSQL βάσεις δεδομένων γιατί δεν απαιτείται να χωρίζονται σε πίνακες.

Όλα τα NoSQL stores χαρακτηρίζονται από το data model το οποίο υποστηρίζουν. Οι βασικές κατηγορίες NoSQL Stores είναι τα key-value, wide-column, document και graph stores [9]. Η επιλογή του τύπου NoSQL Stores που θα χρησιμοποιηθεί κάθε φορά, εξαρτάται από το πως το εκάστοτε NoSQL Store ανταποκρίνεται στα χαρακτηριστικά Ατομικότητα (Atomicity), Συνέπεια (Consistency), Απομόνωση (Isolation) και Αντοχή (Durability) (ACID) [9].

Document Stores: Αποθηκεύουν δεδομένα σε έγγραφα παρόμοια με τα αντικείμενα JSON. Κάθε έγγραφο περιέχει ζεύγη πεδίων και τιμών. Υποστηρίζεται μια ποικιλία τύπων τιμών, όπως συμβολοσειρές, αριθμοί, booleans, πίνακες και λίστες. Οι βάσεις δεδομένων εγγράφων είναι ιδανικές για μια μεγάλη ποικιλία περιπτώσεων χρήσης και μπορούν να χρησιμοποιηθούν ως βάση δεδομένων γενικού σκοπού. Μπορούν να κλιμακωθούν οριζόντια για να φιλοξενήσουν μεγάλους όγκους δεδομένων.

Key-value Stores: Αποτελούν έναν απλούστερο τύπο βάσης δεδομένων όπου κάθε στοιχείο περιέχει κλειδιά και τιμές. Μια τιμή μπορεί συνήθως να ανακτηθεί μόνο με αναφορά του κλειδιού της, οπότε η εκμάθηση του ερωτήματος για ένα συγκεκριμένο ζεύγος κλειδιού-τιμής είναι συνήθως απλή. Οι βάσεις δεδομένων κλειδιού-τιμής είναι ιδανικές για περιπτώσεις χρήσης όπου πρέπει να αποθηκεύονται μεγάλες ποσότητες δεδομένων, αλλά δεν χρειάζεται να εκτελούνται σύνθετα ερωτήματα για την ανάκτηση των δεδομένων.

Wide-column Stores: Αποθηκεύουν δεδομένα σε πίνακες, σειρές και δυναμικές στήλες. Παρέχουν μεγάλη ευελιξία έναντι των σχεσιακών βάσεων δεδομένων, επειδή κάθε σειρά δεν απαιτείται να έχει τις ίδιες στήλες. Πολλοί θεωρούν τις wide-column βάσεις δεδομένων ως δισδιάστατες βάσεις δεδομένων key-value. Οι wide-column βάσεις δεδομένων είναι ιδανικές όταν πρέπει να αποθηκευτούν μεγάλες ποσότητες δεδομένων και μπορούν να προβλεφθούν ποια θα είναι τα μοτίβα των ερωτημάτων.

Graph Stores: Αποθηκεύουν δεδομένα σε κόμβους και άκρα. Οι κόμβοι συνήθως αποθηκεύουν πληροφορίες για άτομα, μέρη και πράγματα, ενώ οι άκρες αποθηκεύουν πληροφορίες σχετικά με τις σχέσεις μεταξύ των κόμβων. Υπερέχουν σε περιπτώσεις χρήσης όπου πρέπει να διασχιστούν σχέσεις για να αναζητηθούν μοτίβα όπως κοινωνικά δίκτυα, εντοπισμό απάτης και μηχανές προτάσεων.

3.1 MongoDB store

Η MongoDB[1] είναι μια από τις πιο γνωστές NoSQL βάσεις δεδομένων. Το data model που χρησιμοποιεί είναι document-oriented. Αποθηκεύει τα δεδομένα σε collections. Τα δεδομένα που αποθηκεύει έχουν μορφή document. Το κάθε document είναι ένα binary JSON (BSON). Η βασική δομή των documents, περιλαμβάνει τα ονόματα των πεδίων μαζί με τις τιμές τους σε ζευγάρια. Κάθε document περιέχει ένα μοναδικό πεδίο ID. Ένα από τα πλεονεκτήματα της MongoDB είναι ότι δεν απαιτεί προκαθορισμένο σχήμα στην δομή των documents σε σύγκριση με ένα RDBMS σύστημα που χρησιμοποιεί προκαθορισμένο σχήμα για τα tables [10]. Έτσι, η MongoDB μπορεί να υποστηρίξει περιπτώσεις, που το σχήμα της βάσης δεδομένων αλλάζει με μεγάλη συχνότητα.

Η σχεδίαση της δομής των documents, είναι μια από τις πιο σημαντικές διεργασίες. Οι 2 βασικές τεχνικές είναι τα Embedded Data Models και τα Normalized Data Models. Τα Embedded Data Models βασίζονται στην λογική, ότι όλη η πληροφορία περιέχεται μέσα στα ίδια τα documents, σε αντίθεση με τα Normalized Data Models που περιέχουν την όλη την πληροφορία σε πολλά documents και για την ανάκτηση της χρησιμοποιούνται references. Τα Normalized Data Models μπορούν να συγκριθούν με τα σχεσιακά συστήματα διαχείρισης δεδομένων, διότι χρησιμοποιούν ίδια λογική στην δομή των tables. Το βασικό πλεονέκτημα της χρήσης Embedded Data Models είναι ότι για την ανάκτηση της πληροφορίας δεν απαιτούνται joins που θεωρείται ακριβή χρονικά διαδικασία.
Η MongoDB υποστηρίζει οριζόντια κλιμάκωση σε κατανεμημένο περιβάλλον. Αυτό το παρέχει με την διαδικασία Sharding που υποστηρίζει. Για το replication των δεδομένων η MongoDB διαθέτει τα replica sets.
Η MongoDB υποστηρίζει απευθείας ευρετηρίαση για χωρικά και κειμενικά δεδομένα.

3.1.1 MongoDB τελεστές ευρετηρίασης

Σε μια βάση δεδομένων χρησιμοποιώντας τεχνικές ευρετηρίασης, μειώνεται ο χρόνος απάντησης των ερωτημάτων με αποτέλεσμα τα συστήματα να γίνονται πιο αποδοτικά. Η MongoDB διαθέτει 2 βασικούς τρόπους ευρετηρίασης. Το Single και Compound index. Ο Single index εφαρμόζεται σε ένα πεδίο του document και σχετίζεται μόνο με αυτό το πεδίο. Ο Compound index δημιουργεί references για πολλά πεδία του document. Υπάρχει όριο 32 πεδίων. Με το που δημιουργηθεί το document στο πεδίο ID εφαρμόζεται Single index. Όταν υπάρχει μια εμφωλευμένη δομή μέσα σε ένα document, τότε μπορεί να χρησιμοποιηθεί ένα Multikey index στο αρχικό πεδίο της δομής που θα ευρετηριάζει όλη την εμφωλευμένη δομή. Η MongoDB διαθέτει τεχνικές ευρετηρίασης για κειμενική αλλά και για χωρική πληροφορία.

[1] https://docs.mongodb.com/manual/

3.1.2 MongoDB κειμενική ευρετηρίαση

Για την textual ευρετηρίαση η MongoDB διαθέτει το Text index. Με το Text index ευρετηριάζεται ένα πεδίο στο document που περιέχει string ή ένα string array. Για να εκτελεστεί ένα ερώτημα, οι βασικοί τελεστές που χρησιμοποιούνται είναι οι $text, $search και $meta. Ο τελεστής $text αντιπροσωπεύει το πεδίο που έχει εφαρμοστεί το index, Ο τελεστής $search εκτελεί την διαδικασία αναζήτησης μιας συμβολοσειράς. Αυτό σημαίνει ότι εάν υπάρχουν παραπάνω από 1 keyword προς αναζήτηση, πρέπει όλα τα keywords να συνενωθούν και να αναπαρασταθούν σε μια συμβολοσειρά. Για παράδειγμα, έστω τα keywords “Italian” “Coffee” “Tea” η αναπαράσταση τους σε μια συμβολοσειρά θα είναι “Italian Coffee Tea”. Η συγκεκριμένη αναπαράσταση εκτελεί τον τελεστή $or για την αναζήτηση των λέξεων. Για να εκτελεστεί ο τελεστής $and, η αναπαράσταση πρέπει να γίνει "\"Italian\" \"Coffee\" \"Tea\"". Ο τελεστής $meta εξετάζει το text similarity. Το score που επιστρέφει ο τελεστής $meta για τις κειμενικές συγκρίσεις που κάνει, υπολογίζεται με τον ακόλουθο τρόπο. Έστω ότι αναζητούνται οι λέξεις “Italian” “Coffee” “Tea” υπάρχουν δυο documents το νούμερο 1 περιέχει τις λέξεις “Italian” “Coffee” “American” “Tea” και το νούμερο 2 “Italian” “Coffee”. Το score που θα επιστραφεί για το document 1 είναι 3.2 και για το νούμερο 2 θα επιστραφεί το score 3.1. Το ακέραιο μέρος υποδηλώνει τις λέξεις που αναζητήθηκαν και το δεκαδικό μέρος τις λέξεις που βρέθηκαν ξεκινώντας την αρίθμηση των λέξεων από το 0. Ο συγκεκριμένος τρόπος υπολογισμού του score αγνοεί το συνολικό πλήθος των λέξεων που υπάρχουν στο κάθε document.

Αντί για το Text index μπορεί να χρησιμοποιηθεί ένα Single index στο πεδίο που περιέχει την κειμενική αναζήτηση. Το συγκεκριμένο πεδίο και σε αυτή την περίπτωση περιέχει string ή ένα string array. Για την εκτέλεση του τελεστή $or χρησιμοποιείται ο τελεστής $in πάνω σε μια λίστα από Strings. Για την εκτέλεση του τελεστή $and χρησιμοποιείται ο τελεστής $all πάνω σε μια λίστα από Strings.

3.1.3 MongoDB χωρική ευρετηρίαση

Για την χωρική ευρετηρίαση η MongoDB χρησιμοποιεί 26 bit binary αναπαράσταση για τα ζευγάρια longitude και latitude. Για τον καθορισμό του χώρου ορίζεται ένα box με τις μέγιστες και ελάχιστες τιμές των longitude και latitude . Η μέγιστη τιμή για το longitude είναι 180 και η ελάχιστη -180 και αντίστοιχα για το latitude 90 και -90. Μέσα στον box δημιουργούνται συνολικά 6.7108.863 κελιά που περιέχουν την geohash τιμή των longitude και latitude του κάθε document. Με αυτό τον τρόπο πραγματοποιείται η χωρική ευρετηρίαση.

Η MongoDB διαθέτει το 2d και το 2dsphere index. Το 2d index υπολογίζει τις γεωμετρίες βάσει 2 διαστάσεων. Η γεωγραφική πληροφορία στα documents αποθηκεύεται ως legacy coordinate pairs. Η δομή legacy coordinate pairs είναι η ακόλουθη:

- location : [longitude, latitude].

Οι αποστάσεις υπολογίζονται μέσω της Ευκλείας απόστασης. Για να καθοριστεί το γεωμετρικό σχήμα του ερωτήματος, χρησιμοποιείται ο τελεστής $geometry. Τα σχήματα που χρησιμοποιεί η MongoDB για το 2d index, ορίζονται από τους τελεστές $box και $polygon. Για το γεωμετρικό σχήμα του κύκλου, χρησιμοποιείται ο τελεστής $geoWithin μαζί με τον τελεστή $center.

Το 2dsphere index υπολογίζει τις γεωμετρίες, βάσει των διαστάσεων της γης. Η γεωγραφική πληροφορία στα documents αποθηκεύεται ως GeoJSON object. Η δομή ενός GeoJSON object είναι η ακόλουθη**:**

- location: {type: "Point", coordinates: [longitude, latitude]}.

Για να καθοριστεί το γεωμετρικό σχήμα του ερωτήματος χρησιμοποιούνται οι τελεστές $geoWithin και $geometry και στην συνέχεια το ερώτημα εκφράζεται σαν GeoJSON object.

3.2 HBase store

Το 2006 η Google ανακοινώνει ένα εσωτερικό κατανεμημένο χώρο αποθήκευσης που ονομάζεται BigTable [11]. Σχεδιάστηκε για να αποθηκεύει petabytes δεδομένων σε χιλιάδες διακομιστές. Το Bigtable είναι ένας αραιός, κατανεμημένος, πολυδιάστατος ταξινομημένος χάρτης. Ο χάρτης ευρετηριάζεται με ένα κλειδί γραμμής, ένα κλειδί στήλης και μια χρονική σήμανση. Κάθε τιμή στον χάρτη είναι ένας πίνακας από bytes.

Η HBase είναι βάση δεδομένων NoSQL που βασίζεται στην αρχιτεκτονική του Google BigTable. Η HBase είναι πολύ περισσότερο ένα "Data Store" από το "Database", επειδή δεν διαθέτει πολλές από τις δυνατότητες που παρέχει ένα σχεσιακό σύστημα διαχείρισης δεδομένων, όπως απευθείας ευρετηρίαση και την υποστήριξη μιας προηγμένης γλώσσας ερωτημάτων. Ωστόσο, η HBase έχει πλεονέκτημα έναντι ενός σχεσιακού συστήματος διαχείρισης δεδομένων, προσφέροντας γραμμική κλιμάκωση. Ένα σχεσιακό σύστημα διαχείρισης δεδομένων μπορεί να κλιμακωθεί καλά, αλλά μόνο μέχρι ενός σημείου το μέγεθος της βάσης δεδομένων. Για καλύτερη απόδοση απαιτείται εξειδικευμένο υλικό και συσκευές αποθήκευσης. Τα συμπλέγματα HBase επεκτάθηκαν προσθέτοντας RegionServers που φιλοξενούνται σε διακομιστές. Εάν ένα σύμπλεγμα επεκτείνεται από 10 σε 20 RegionServers, για παράδειγμα, αυξάνει τόσο από την άποψη της χωρητικότητας αποθήκευσης όσο και της επεξεργασίας.

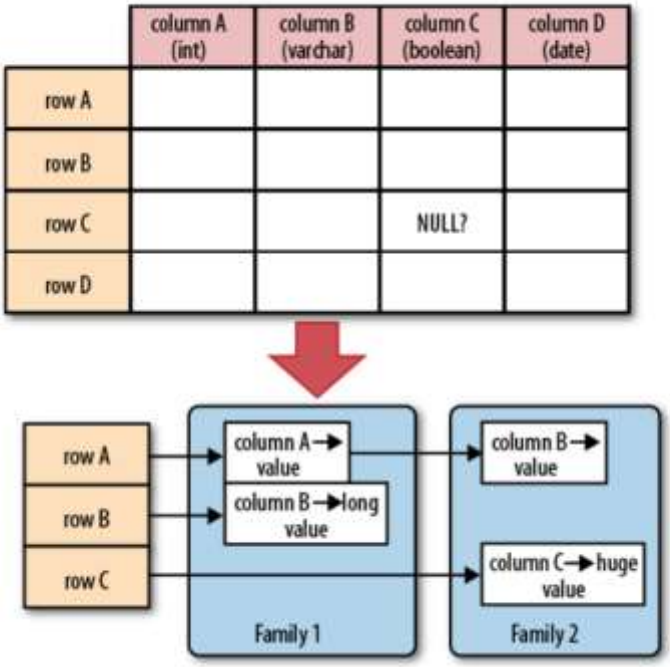
Αξιοσημείωτα χαρακτηριστικά της HBase:

- Οι πίνακες της HBase κατανέμονται στο cluster μέσω των regions. Τα regions χωρίζονται αυτόματα και κατανέμονται σε διακομιστές καθώς αυξάνονται τα δεδομένα.

- Όταν στο HBase cluster ανιχνευθεί ότι, κάποιος κόμβος απέτυχε, τότε μεταφέρει τα δεδομένα σε άλλο κόμβο.
- Αποθηκεύει τα νέα δεδομένα στην μνήμη και στον δίσκο του μηχανήματος.
- Λόγω του τα timestamp που προσθέτει στα δεδομένα και λόγω της ταξινομημένης σειράς που αποθηκεύει τα δεδομένα. Είναι κατάλληλη για την επίλυση προβλημάτων μηχανικής μάθησης που περιέχουν χρονοσειρές.

Η Apache Hbase[2] χρησιμοποιεί column oriented (Bigtable) data model και τρέχει πάνω σε Apache Hadoop περιβάλλον. Η HBase αποθηκεύει τα δεδομένα σε tables. Κάθε table περιέχει rows και column families. Κάθε column family περιέχει τουλάχιστον 1 column. Τα rows αντιπροσωπεύουν τα row keys, δηλαδή ένα μοναδικό ID. Δεν υπάρχει κάποιος τύπος δεδομένος στην HBase. Η HBase αποθηκεύει τα δεδομένα σαν byte arrays μέσα στα cells των HBase tables. Οι τιμές μέσα στα cells χαρακτηρίζονται από ένα timestamp, όταν αυτές αποθηκεύονται στα cells. Κάθε column χαρακτηρίζεται από το row key, το column family και ένα timestamp. Η HBase παρέχει απευθείας πρόσβαση στα δεδομένα μέσω του rowkey. Επίσης, υποστηρίζει random read/write πρόσβαση στα δεδομένα.

- (Row (ColumnFamily(Column(Cell)))).



Εικόνα 3.1 HBase μοντέλο δεδομένων.

Η HBase δεν υποστηρίζει απευθείας ευρετηρίαση για χωρικά και κειμενικά δεδομένα.

[2] https://hbase.apache.org/

3.2.1 HBase βασικές εντολές

Για να εκτελεστούν βασικές λειτουργίες στην HBase χρησιμοποιούνται οι παρακάτω εντολές.

Put: Προσθέτει νέες σειρές σε έναν πίνακα (εάν το κλειδί είναι νέο) ή ενημερώνει υπάρχουσες σειρές (εάν το κλειδί υπάρχει ήδη). Το Put μπορεί να αλλάξει ατομικά πολλά κελιά σε μια συγκεκριμένη σειρά.

Increment: Ειδικός τύπος λειτουργίας που παρέχει έναν τρόπο αύξησης της τιμής που περιέχεται σε κάποιο κελί. Μπορεί να χρησιμοποιηθεί για την εφαρμογή μετρητή σε υπηρεσίες κατανεμημένης συγκέντρωσης.

Delete: Διαγράφει ένα ή περισσότερα κελιά που σχετίζονται με συγκεκριμένη σειρά ή αφαιρεί ολόκληρη τη σειρά. Η HBase δεν καταργεί τα δεδομένα και έτσι οι διαγραφές πραγματοποιούνται με χρήση νέων δεικτών που ονομάζονται tombstones.

CAS: Αυτό το σύνολο λειτουργίας μοιάζει με τις οδηγίες σύγκρισης και ρύθμισης της CPU και περιλαμβάνει τις ακόλουθες λειτουργίες: check-and-post ή check-and-delete.

Get: Βασική λειτουργία «ανάγνωσης». Η λήψη μπορεί να επιστρέψει ολόκληρη τη σειρά ή ορισμένες συγκεκριμένες στήλες σειράς.

Scan: Πιο προηγμένη λειτουργία ανάγνωσης. Παρέχει έναν τρόπο σάρωσης εύρους γραμμών που ορίζεται από τα κλειδιά έναρξης και λήξης. Η σάρωση μπορεί να διασχίσει το εύρος των κλειδιών ξεκινώντας από το κλειδί «έναρξης» έως το κλειδί «τερματισμού» καθώς και από το κλειδί «τερματισμού» έως το κλειδί «έναρξης».

Κεφάλαιο 4

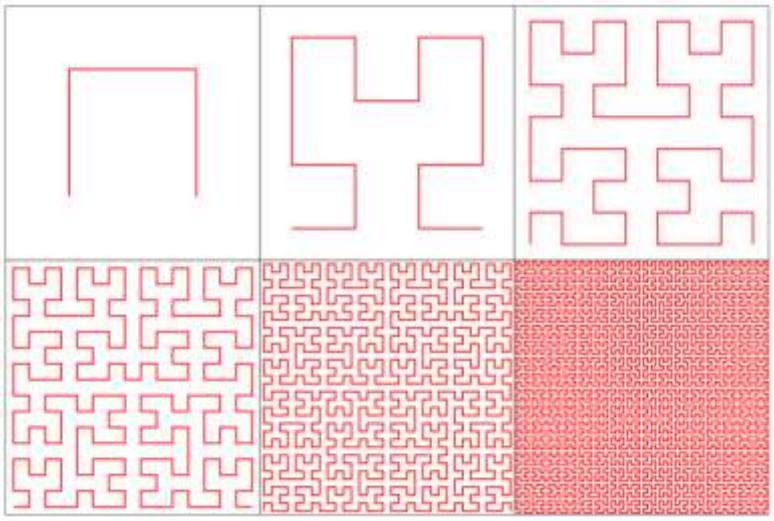
4. Μεθοδολογία

Σε αυτό το κεφάλαιο, θα πραγματοποιηθεί επεξήγηση της μεθοδολογίας που ακολουθήθηκε για την εφαρμογή χωρο-κειμενικής ευρετηρίασης πάνω από NoSQL stores, καθώς θα πραγματοποιηθεί αναφορά για την υλοποίηση της μεθοδολογίας πάνω στο NoDA API για τα MongoDB και HBase stores.

4.1 Περιγραφή μεθοδολογίας

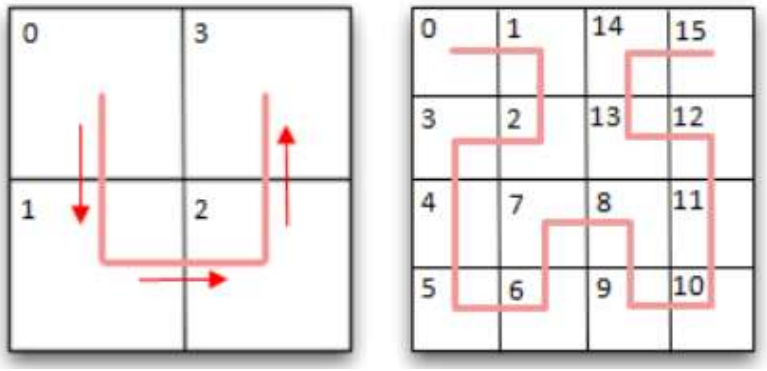
Στη μεθοδολογία εφαρμόζονται Boolean Range Queries σε χωρο-κειμενικά δεδομένα πάνω από NoSQL stores. Η μεθοδολογία βασίζεται στην τεχνική καμπύλη πλήρωσης χώρου (Space Filling Curve) Hilbert Curve για την διαχείριση της χωρικής πληροφορίας και για την διαχείριση της κειμενικής πληροφορίας χρησιμοποιούνται Post Lists.

Μια καμπύλη πλήρωσης χώρου είναι μια καμπύλη της οποίας το εύρος περιέχει ολόκληρο το δισδιάστατο τετράγωνο της μονάδας. Αυτό σημαίνει ότι, κάθε σημείο μέσα σε ένα τετράγωνο (2 διαστάσεων) μπορεί να το χαρτογραφηθεί σε ένα σημείο στην καμπύλη (1 διάσταση) και να επιστραφεί ξανά.
Στην πραγματικότητα, η καμπύλη πλήρωσης χώρου δημιουργείται αναδρομικά μέσω διαδοχικών διασχίσεων. Κάθε διάσχιση ονομάζεται συνήθως μια σειρά για αυτήν την καμπύλη. Για παράδειγμα, ακολουθούν 1 έως 6 σειρές της καμπύλης Hilbert.



Εικόνα 4.1 Hilbert curve 1-6 διασχίσεις

Παρακάτω φαίνονται πως διαμορφώνεται καμπύλη Hilbert για τις 2 πρώτες σειρές.



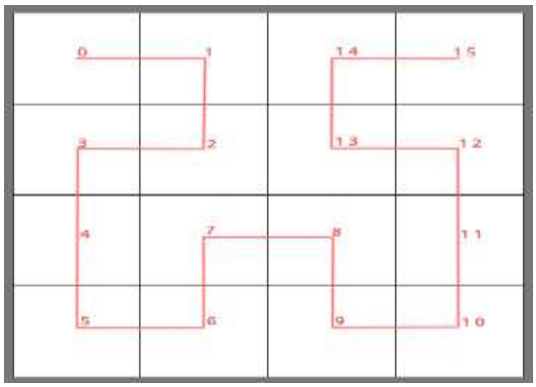
Εικόνα 4.2 Hilbert curve 2 διασχίσεις.

Η καμπύλη Hilbert της πρώτης σειράς επιτρέπει 4 τιμές από 0 έως 3. Η καμπύλη της δεύτερης σειράς Hilbert επιτρέπει 16 τιμές από 0 έως 15. Υπάρχει αντιστοιχία ένας προς έναν μεταξύ της τιμής Hilbert και της τιμής του τετραγώνου. Παρατηρείται ότι η δυαδική αναπαράσταση των τιμών κατά μήκος της καμπύλης Hilbert, σχετίζεται με κάθε κελί του τετραγώνου, για την πρώτη σειρά: 00, 01, 10 και 11. Για τη δεύτερη σειρά, τα πρώτη τέσσερα κελιά από 0 έως 3 μπορούν να αναπαρασταθούν ως 0000, 0001, 0010, 0011. Τα επόμενα τέσσερα κελιά από 4 έως 7 έχουν δυαδική αναπαράσταση 0100, 0101, 0110, 0111 και ούτω καθεξής.

Έστω ένα χωρο-κειμενικό αντικείμενο περιέχει την ακόλουθη πληροφορία { longitude =x, latitude =y και keywords = { keyword1, keyword2 }}. Η εισαγωγή των δεδομένων στο NoSQL Store γίνεται βάσει, στο ότι το κάθε χωρο-κειμενικό αντικείμενο έχει

χωριστεί σε τόσα αντικείμενα όσο είναι το πλήθος των λέξεων που περιέχει. Στο κάθε καινούργιο αντικείμενο που παράγεται, η γεωγραφική συντεταγμένη x,y είναι ίδια και προέρχεται από το αρχικό χωρο-κειμενικό αντικείμενο. Αυτό που αλλάζει είναι το keyword. Κάθε νέο αντικείμενο έχει ένα πεδίο με όνομα groupid που υποδεικνύει από ποιο χωρο-κειμενικό αντικείμενο προέρχεται. Επίσης, υπάρχει ένα πεδίο με όνομα hilbertindex όπου περιέχει μια μονοδιάστατη τιμή της γεωγραφική συντεταγμένης x,y που έχει εφαρμοστεί η Hilbert Curve μέθοδος, συνενωμένη με το keyword.

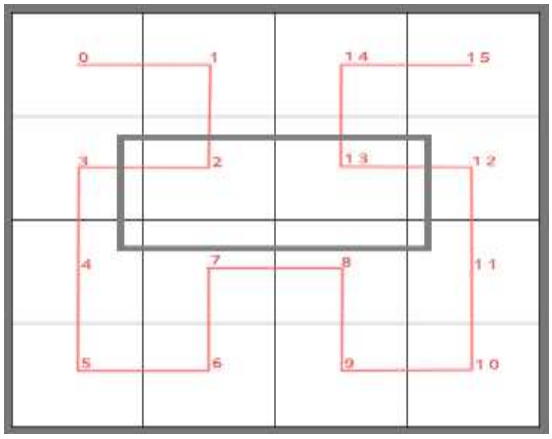
Για να παραχθεί μία μονοδιάστατη τιμή, ώστε να πραγματοποιηθεί η χωρο-κειμενική ευρετηρίαση. Η τιμή του πεδίου hilbertindex έχει υπολογιστεί βάσει της μονοδιάστατης τιμής Hilbert Curve για την γεωγραφική συντεταγμένη x,y του object, προσθέτοντας το αντίστοιχο keyword. Για τον υπολογισμό της μονοδιάστατης τιμής, ακολουθείται η εξής μεθοδολογία. Αρχικά υπάρχει η ανάγκη να δημιουργηθεί κάποιος συγκεκριμένος χώρος που θα περιέχει τα σημεία x,y για κάθε object. Γι' αυτό δημιουργείται ένα box [xmax,ymax],[xmin,ymin] που περιέχει όλα τα σημεία. Η μέγιστη τιμή για το longitude είναι 180 και η ελάχιστη -180 και αντίστοιχα για το latitude 90 και -90. Στη συνέχεια εντός του box δημιουργούνται κελία βάσει της Hilbert Curve αναπαράστασης που έχει οριστεί. Ορίζεται ακρίβεια αναπαράστασης 10 bits για τα x,y Για παράδειγμα, έστω ότι όλα τα x αναπαρίστανται με 2 bits και αντίστοιχα όλα τα y αναπαρίστανται με 2 bits, τότε θα προκύψουν 15 κελία όπου το καθένα θα περιέχει μια μονοδιάστατη τιμή hilbert. Η δυαδική τιμή hilbert 1111 αντιστοιχεί στην τιμή 15 στο δεκαδικό σύστημα. Υπάρχει το ενδεχόμενο κάθε κελί να έχει παραπάνω από 1 σημεία. Η τεχνική φαίνεται αναλυτικά στην Εικόνα 4.4.



Εικόνα 4.3 Hilbert curve - χωρικό ευρετήριο

Μέσα σε κάθε κελί υπάρχουν πολλά x,y που αντιστοιχούν σε objects. Η τιμή στο πεδίου hilbertindex του κάθε object, προκύπτει από την τιμή hilbert σε συνδυασμό με το keyword που περιλαμβάνει το κάθε object. Στην συνέχεια ορίζεται το BRQ.

BRQ – Να βρεθούν όλα τα objects που βρίσκονται εντός του box1 [x1,y1], [x2,y2] και περιέχουν τις λέξεις k1,k2.



Εικόνα 4.4 BRQ ερώτημα σε Hilbert τιμές

Το πρόβλημα που δημιουργείται είναι ότι θα επιστραφούν περισσότερα σημεία απ' όσα ανήκουν μέσα στο Box1 και αυτό γιατί θα επιστραφούν όλα τα objects από τα κελιά ακουμπάει το Box1.

Η MongoDB επιλύει το συγκεκριμένο πρόβλημα μέσω του aggregation pipeline εκμεταλλεύεται πλήρως το index που δίνεται στο query και φέρνει στην μνήμη πρώτα τα δεδομένα που υπάρχουν στο index. Άρα στην περίπτωση που σχηματιστεί ένα query με χωρικούς και hilbert παραμέτρους όπου για την παράμετρο hilbertindex θα υπάρχει index. Τότε λύνεται το πρόβλημα βέλτιστα.
Η HBase επιλύει το συγκεκριμένο πρόβλημα προσθέτοντας στο pipeline του ερωτήματος τις ανισότητες για το σχηματισμό του γεωμετρικού σχήματος πέρα από την αναζήτηση της hilbertindex τιμής.

4.1.1 Μεθοδολογία σε MongoDB

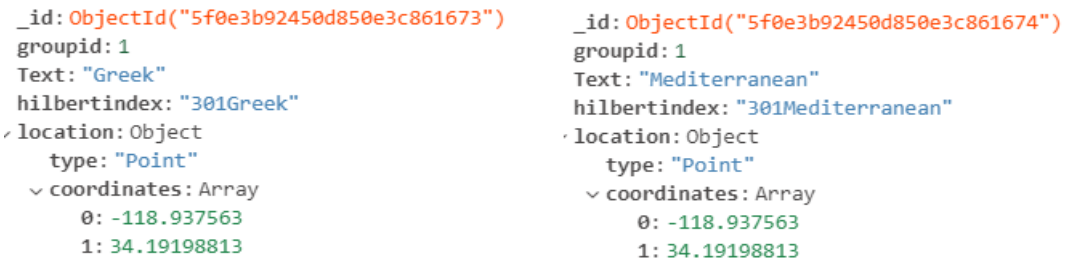
Ο αλγόριθμος της μεθοδολογίας για την MongoDB χωρίζεται σε 2 φάσεις την εισαγωγή των δεδομένων στη MongoDB και στην εκτέλεση των ερωτημάτων.

Διαδικασία εισαγωγής:

Στην περίπτωση της MongoDB, ένα χωρο-κειμενικό αντικείμενο με μια λίστα από λέξεις αναπαρίσταται, σε πολλά MongoDB έγραφα. Δημιουργούνται τόσα έγγραφα όσο είναι και το πλήθος των λέξεων που περιέχει το χωρο-κειμενικό αντικείμενο. Η γεωγραφική πληροφορία του χωρο-κειμενικού αντικείμενου περιέχεται σε κάθε παραγόμενο έγγραφο. Οι hilbert τιμές παράγονται βάσει των μέγιστων και ελάχιστων γεωγραφικών συντεταγμένων που περιέχουν τα δεδομένα. Το κάθε έγγραφο περιέχει μια μονοδιάστατη τιμή που συνδυάζει την hilbert τιμή του εγγράφου με την αντίστοιχη λέξη. Το κάθε έγγραφο, περιέχει ένα πεδίο groupid που υποδηλώνει από ποιο χωρο-κειμενικό αντικείμενο προέρχεται, την γεωγραφική πληροφορία του χωρο-κειμενικού αντικειμένου, την αντίστοιχη λέξη προς αναζήτηση, καθώς και την hilbert τιμή συνενωμένη με την αντίστοιχη λέξη προς αναζήτηση.

Δίνεται το ακόλουθο παράδειγμα, έστω το ένα χωρο-κειμενικό αντικείμενο με την ακόλουθη πληροφορία { longitude =-118.937563, latitude =34.19198813 και

keywords = { “Greek”, “Mediterranean” }}. Τα έγγραφα που θα δημιουργηθούν φαίνονται στην εικόνα 4.3.



Εικόνα 4.5 MongoDB μορφή εγγράφων

```
1.  documents= getallspatiotextualobjects();          //Ανάκτηση όλων των χωρο-κειμενικών αντικειμένων
2.
3.  //Πεδίο ορισμού σύμφωνα με το περιεχόμενο των δεδομένων
4.  maxlon=getmaxlon(); minlon=getminlon(); maxlat=getmaxlat(); minlat=getminlat();
5.
6.  documentswithindex=[];
7.
8.  For each doc in documents:
9.    //Δημιουργία Hilbert τιμών
10.   hilbertvalue=hilbertfunction(doc.geo, maxlon, minlon, maxlat,minlat);
11.
12.    For each keyword in doc.keywords: // Εισαγωγή των documents στην MongoDB
13.     Importdoc(documentswithindex.append(doc.geo,hilbertvalue+keyword,  groupid, keyword));
14.     groupid=groupid+1;
```

Πίνακας 4.1 Ψευδοκώδικας – εισαγωγή δεδομένων στην MongoDB

Εκτέλεση ερωτημάτων:

Για την εκτέλεση ενός ερωτήματος, αρχικά ορίζονται οι λέξεις προς αναζήτηση καθώς και ο χώρος αναζήτησης που περιλαμβάνει ελάχιστο και μέγιστο γεωγραφικό πλάτος και μήκος δημιουργώντας ένα κουτί που θεωρείται το πεδίο ορισμού του ερωτήματος. Στη συνέχεια για το συγκεκριμένο πεδίο ορισμού εφαρμόζεται η τεχνική Hilbert curve, παράγοντας μονοδιάστατες τιμές από τις γεωγραφικές συντεταγμένες

που βρίσκονται μέσα στο κουτί. Στο επόμενο στάδιο, για κάθε μια μονοδιάστατη τιμή που παράχθηκε, προστίθενται οι λέξεις προς αναζήτηση. Έστω μια hilbert τιμή 301 και οι λέξεις “Italian”, “Coffee” και “Tea” θα παραχθούν τα κλειδιά “301Italian”, “301Coffee”, “301Tea”. Στο τέλος, αναζητούνται τα παραγόμενα κλειδιά στην MongoDB.

```
1.  searchkeywords= [“Italian”,”Coffee”, “Tea”];   //Λέξεις προς αναζήτηση
2.
3.  //Δημιουργία Hilbert τιμών βάσει των του χώρου αναζήτησης που ορίζει ο χρήστης
4.  range= hilbertfunction(querymaxlon, queryminlon, querymaxlat, queryminlat);
5.  searchlist=[];
6.  For each hilbertvalue in range:
7.    For keyword  in searchkeywords:
8.        searchlist.append(hilbertvalue+ keyword )
9.
10. performquery(filter(searchlist, or)); // εκτέλεση ερωτήματος
```

Πίνακας 4.2 Ψευδοκώδικας – εκτέλεση ερωτήματος στην MongoDB

4.1.2 Μεθοδολογία σε HBase

Ο αλγόριθμος της μεθοδολογίας και στην HBase χωρίζεται σε 2 φάσεις, την εισαγωγή των δεδομένων στην HBase και στην εκτέλεση των ερωτημάτων.

Διαδικασία εισαγωγής:

Στην περίπτωση της HBase, ένα χωρο-κειμενικό αντικείμενο με μια λίστα από λέξεις αναπαρίσταται, σε πολλές γραμμές. Το πλήθος των γραμμών που αντιπροσωπεύει ένα χωρο-κειμενικό αντικείμενο στην HBase, εξαρτάται από το πλήθος των πεδίων που περιέχει το χωρο-κειμενικό αντικείμενο * το πλήθος των λέξεων προς αναζήτηση. Οι hilbert τιμές παράγονται βάσει των μέγιστων και ελάχιστων γεωγραφικών συντεταγμένων που περιέχουν τα δεδομένα. Κάθε αντικείμενο στην HBase αποτελείται από τόσες γραμμές όσα είναι και τα πεδία του χωρο-κειμενικού αντικειμένου. Κάθε γραμμή του αντικείμενου έχει ένα αναγνωριστικό rowkey που περιέχει την hilbert τιμή μαζί με έναν αύξοντα αριθμό. Επιπλέον, η κάθε γραμμή περιέχει το αναγνωριστικό column family, έχει δοθεί η λέξη “data”. Το κάθε αντικείμενο, περιέχει ένα πεδίο groupid που υποδηλώνει από ποιο χωρο-κειμενικό αντικείμενο προέρχεται, την γεωγραφική πληροφορία του χωρο-κειμενικού αντικειμένου, την αντίστοιχη λέξη προς αναζήτηση, καθώς και την hilbert τιμή συνενωμένη με την αντίστοιχη λέξη προς αναζήτηση. Επεξεργασία των δεδομένων στην HBase, πραγματοποιείται σε bytes.

Δίνεται το ακόλουθο παράδειγμα, έστω το ένα χωρο-κειμενικό αντικείμενο με την ακόλουθη πληροφορία { longitude =-118.937563, latitude =34.19198813 και keywords = { “Greek”, “American” }}. Οι γραμμές που θα δημιουργηθούν φαίνονται παρακάτω.

- **HilbertValue /1**, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- **HilbertValue/1**, column=data:hilbertindex, timestamp=**DefaultTimestampValue**, value= **HilbertValueGreek.**
- **HilbertValue/1**, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.
- **HilbertValue/1**, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.
- **HilbertValue/1**, column=data:keyword, timestamp=**DefaultTimestampValue**, value=Greek.
- **HilbertValue/2**, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- **HilbertValue/2**, column=data:hilbertindex, timestamp=**DefaultTimestampValue**, value= **HilbertValueAmerican.**
- **HilbertValue/2**, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.
- **HilbertValue/2**, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.
- **HilbertValue/2**, column=data:keyword, timestamp=**DefaultTimestampValue**, value= American.

```
1.  objects= getallspatiotextualobjects();          //Ανάκτηση όλων των χωρο-κειμενικών αντικειμένων
2.
3.  //Πεδίο ορισμού σύμφωνα με το περιεχόμενο των δεδομένων
4.  maxlon=getmaxlon(); minlon=getminlon(); maxlat=getmaxlat(); minlat=getminlat();
5.
6.  objectswithindex=[];
7.  counter=1.
8.  groupid=1;
9.  For each o in objects:
10.   //Δημιουργία Hilbert τιμών
11.   hilbertvalue=hilbertfunction(o.geo, maxlon, minlon, maxlat,minlat);
12.
13.    For each keyword in o.keywords: // Εισαγωγή δεδομένων στην Hbase
14.      row= (hilbertvalue+counter)toBytes();
15.      columnfamily= (“data”) toBytes();
16.      columnkeyword=(keyword) toBytes();
17.      columnspatial=(o.geo) toBytes();
18.      columngroupid=(groupid) toBytes();
19.      columnhilbertindex=( hilbertvalue+ keyword) toBytes();
20.
```

```
21.  Importdata(objectswithindex.append(row, columnfamily, columnkeyword,
     columnspatial, columngroupid, columnhilbertindex));
22.    counter=counter+1;
23.  groupid= groupid+1;
24.
```

Πίνακας 4.3 Ψευδοκώδικας – εισαγωγή δεδομένων στην HBase

Εκτέλεση ερωτημάτων:

Για την εκτέλεση ενός ερωτήματος, αρχικά ορίζονται οι λέξεις προς αναζήτηση καθώς και ο χώρος αναζήτησης που περιλαμβάνει ελάχιστο και μέγιστο γεωγραφικό πλάτος και μήκος δημιουργώντας ένα κουτί που θεωρείται το πεδίο ορισμού του ερωτήματος όπως ακριβώς και στην περίπτωση της MongoDB. Στη συνέχεια για το συγκεκριμένο πεδίο ορισμού εφαρμόζεται η τεχνική Hilbert curve, παράγοντας μονοδιάστατες τιμές από τις γεωγραφικές συντεταγμένες που βρίσκονται μέσα στο κουτί. Στο επόμενο στάδιο, για κάθε μια μονοδιάστατη τιμή που παράχθηκε, προστίθενται οι λέξεις προς αναζήτηση. Τέλος χρησιμοποιούνται φίλτρα της HBase για να σχηματιστεί το ερώτημα βάσει του γνωρίσματος column family και του παραγόμενου κλειδιού αναζήτησης. Επεξεργασία των δεδομένων στην HBase, πραγματοποιείται σε bytes.

```
1.  searchkeywords= [“Italian”,”Coffee”, “Tea”];  //Λέξεις προς αναζήτηση
2.
3.   //Δημιουργία Hilbert τιμών βάσει των του χώρου αναζήτησης που ορίζει ο χρήστης
4.   range= hilbertfunction(querymaxlon, queryminlon, querymaxlat, queryminlat);
5.  searchlist=[];
6.   For each hilbertvalue in range:
7.    For keyword  in searchkeywords:
8.      columnfamily= (“data”) toBytes();
9.      columnhilbertindex=( hilbertvalue+ keyword) toBytes();
10.     searchlist.append(columnfamily, columnhilbertindex)
11.
12.  performquery(filter(searchlist, or)); // εκτέλεση ερωτήματος
```

Πίνακας 4.4 Ψευδοκώδικας – εκτέλεση ερωτήματος στην HBase

4.2 Εφαρμογή στο NoDA API

Το NoDA API βασίζεται σε προηγούμενη δουλεία της Big Data ερευνητικής ομάδας του Πανεπιστημίου Πειραιώς. Αντιπροσωπεύει ένα abstract layer, το οποίο θα διευκολύνει την πρόσβαση σε δεδομένα που είναι αποθηκευμένα σε NoSQL stores. Δίνει την δυνατότητα εκτέλεσης χωρο-κειμενικών ερωτημάτων ανεξάρτητα από το NoSQL Store που είναι αποθηκευμένα τα δεδομένα. Υποστηρίζει τεχνικές χωρο-χρονικής και χωρο-κειμενικής ευρετηρίασης σε NoSQL stores. Η γλώσσα

προγραμματισμού που χρησιμοποιείται στο NoDA API είναι η Java. Περισσότερες πληροφορίες για το NoDA υπάρχουν στην παραπομπή [1].

Αναπτύχθηκαν για τα NoSQL Stores MongoDB, HBase οι παρακάτω κλάσεις για την χωρο-κειμενική ευρετηρίαση:

- **OperatorInGeoTextualRectangle:** Υποστηρίζει χωρο-κειμενικά Boolean Range Queries με γεωμετρικό σχήμα ορθογώνιο.
- **OperatorInGeoTextualPolygon:** Υποστηρίζει χωρο-κειμενικά Boolean Range Queries με γεωμετρικό σχήμα πολύγονο.
- **OperatorInGeoTextualCircle:** Υποστηρίζει χωρο-κειμενικά Boolean Range Queries με γεωμετρικό σχήμα κύκλο.

Ακολουθεί δείγμα του κώδικα που εκτελείται για τις παραπάνω κλάσεις.

```
Ranges rangesList = h.query(HilbertUtil.scaleGeoTextualPoint(lon1,
minLon,maxLon,lat1, minLat,maxLat, max),

HilbertUtil.scaleGeoTextualPoint(lon2,minLon,maxLon,lat2,
minLat,maxLat, max));

String [] keywords=conditionalTextualOperator.getKeywords();

StringBuilder sb = new StringBuilder();

StringBuilder sbkewords = new StringBuilder();

for(int word =0;word<keywords.length;word++){

String keyword=keywords[word];

sbkewords.append("\""+keyword+"\""+",");

rangesList.stream().forEach(i->{

for(long k=i.low(); k<= i.high(); k++){

sb.append("\""+k+keyword+"\""+",");

}});}
```

```
sb.deleteCharAt(sb.length()-1);
```

Πίνακας 4.5 Κώδικας χωρο-κειμενικής ευρετηρίασης στο NoDA API

Κεφάλαιο 5

5. Πειραματική διαδικασία

Σε αυτό το κεφάλαιο θα παρουσιαστούν δοκιμές και πειράματα για την μεθοδολογία που αναπτύχθηκε στο Κεφάλαιο 4 στα NoSQL stores MongoDB και HBase.

5.1 Εφαρμογή σε MongoDB store

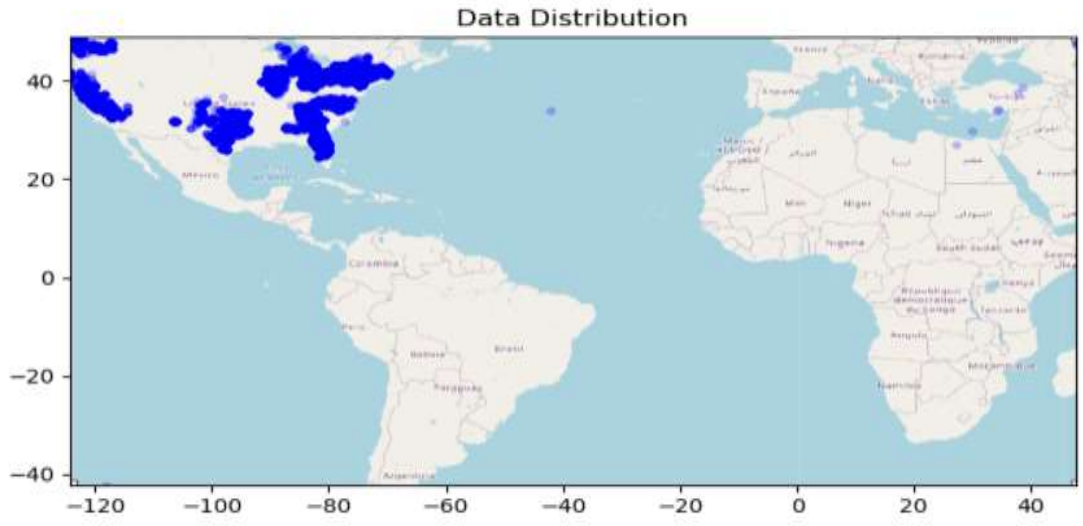
5.1.1 MongoDB σε κεντρικοποιημένο περιβάλλον

Η MongoDB υποστηρίζει την εκτέλεση ερωτημάτων είτε μέσω την εντολής find είτε μέσω aggregation pipeline. Σε αυτό το στάδιο, θα συγκριθούν οι 2 τρόποι εκτέλεσης των ερωτημάτων. Ο παρακάτω πίνακας περιέχει τα τεχνικά χαρακτηρίστηκα της υπολογιστικής μονάδας που έτρεξε η πειραματική διαδικασία.

CPU	RAM	HARD DISK
4 cores (Intel Core i7)	16 GB	256 GB SSD

Πίνακας 5.1 MongoDB – πόροι κεντρικοποιημένης υποδομής

Τα δεδομένα της πειραματικής διαδικασίας προήλθαν από το dataset restaurants-ver1 με 78981 εγγραφές. Δημιουργήθηκαν 4 collections στη MongoDB. Τα 2 πρώτα geoindex και geotextualindex περιείχαν 78981 Documents. Στο ένα collection δημιουργήθηκε μόνο χωρική ευρετηρίαση στο επόμενο προστέθηκε και η κειμενική. Στην συνέχεια το περιεχόμενο του αρχείου restaurants-ver1 αντιγράφηκε 13 φορές και παράχθηκαν τα 2 επόμενα collections geoindexbigdata και geotextualindexbigdata με 1026753 Documents το καθένα. Έπειτα, ακολουθήθηκε η ίδια διαδικασία όπως και στα 2 πρώτα collection. Τα collections geoindex και geoindexbigdata έχουν 2d index. Τα collections geotextualindex και geotextualindexbigdata περιέχουν 2d index και Text index. Παρακάτω δίνεται η γεωγραφική απεικόνιση της κατανομής των δεδομένων του dataset restaurants-ver1.



Εικόνα 5.1 Κατανομή δεδομένων

Παρακάτω δίνονται τα ερωτήματα που δοκιμάστηκαν στην πειραματική διαδικασία.

Q1 - να βρεθούν όλα τα documents που βρίσκονται εντός του πολυγώνου [[-118.937563,34.19198813], [-96.6995,32.974964],[-95.389055,29.723262], [-

118.937563,34.19198813]] και περιέχουν μια ή περισσότερες από τις λέξεις "Italian","Coffee","Tea".

Q2 - να βρεθούν όλα τα documents που βρίσκονται εντός του πολυγώνου [[-118.937563,34.19198813], [-96.6995,32.974964],[-95.389055,29.723262], [-118.937563,34.19198813]] και περιέχουν όλες τις ακόλουθες λέξεις "Italian","Coffee","Tea".

Q3 - να βρεθούν όλα τα documents που βρίσκονται εντός του κύκλου [-118.937563,40.78422] με ακτίνα 200/3963.2 miles και περιέχουν μια ή περισσότερες από τις λέξεις "Italian","Coffee","Tea".

Q4 - να βρεθούν όλα τα documents που βρίσκονται εντός του κύκλου [-118.937563,40.78422] με ακτίνα 200/3963.2 miles και περιέχουν όλες τις ακόλουθες λέξεις "Italian","Coffee","Tea".

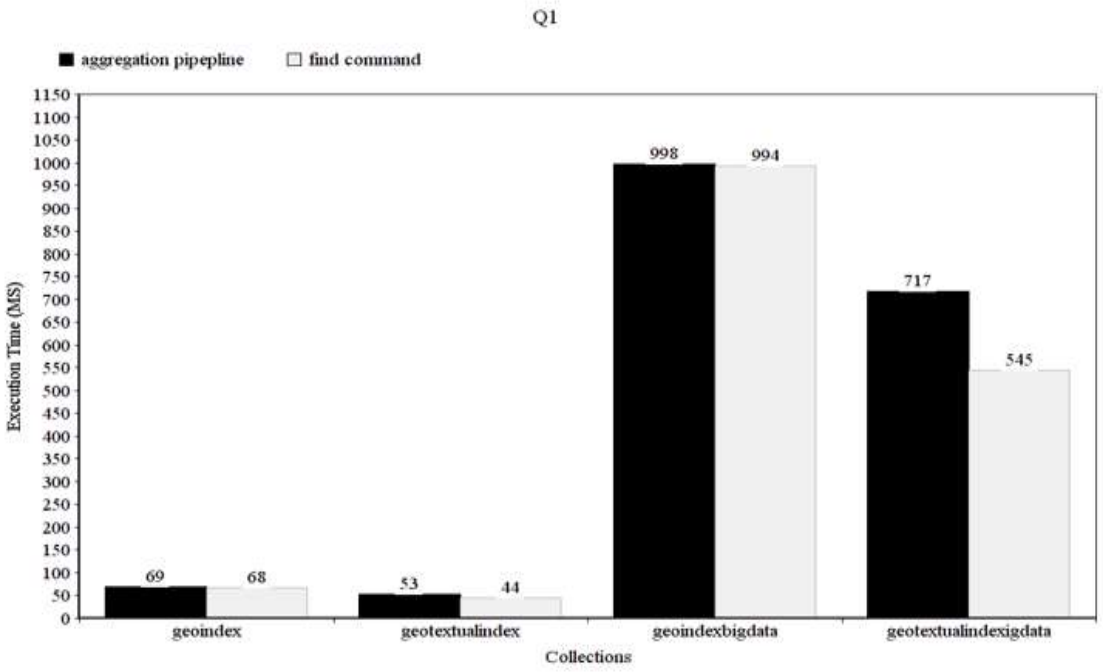
Q5 – να βρεθούν τα top 3 αντικείμενα βάσει του text similarity για τις λέξεις “cake”, “tea” και βρίσκονται εντός του πολυγώνου [[-118.937563,34.19198813], [-96.6995,32.974964],[-95.389055,29.723262], [-118.937563,34.19198813]].

Q6 – να βρεθούν τα top 3 αντικείμενα βάσει του text similarity για τις λέξεις “cake”, “tea” και βρίσκονται εντός του κύκλου [-118.937563,40.78422] με ακτίνα 200/3963.2 miles.

Τα παραπάνω queries υλοποιήθηκαν στη MongoDB με 2 τρόπους. Με χρήση της τεχνικής aggregation pipeline και με χρήση της εντολής find. Στο σημείο αυτό αξίζει να σημειωθεί ότι στο manual της MongoDB αναφέρει ότι με χρήση της τεχνικής aggregation pipeline πραγματοποιείται καλύτερη διαχείριση των ευρετηρίων.

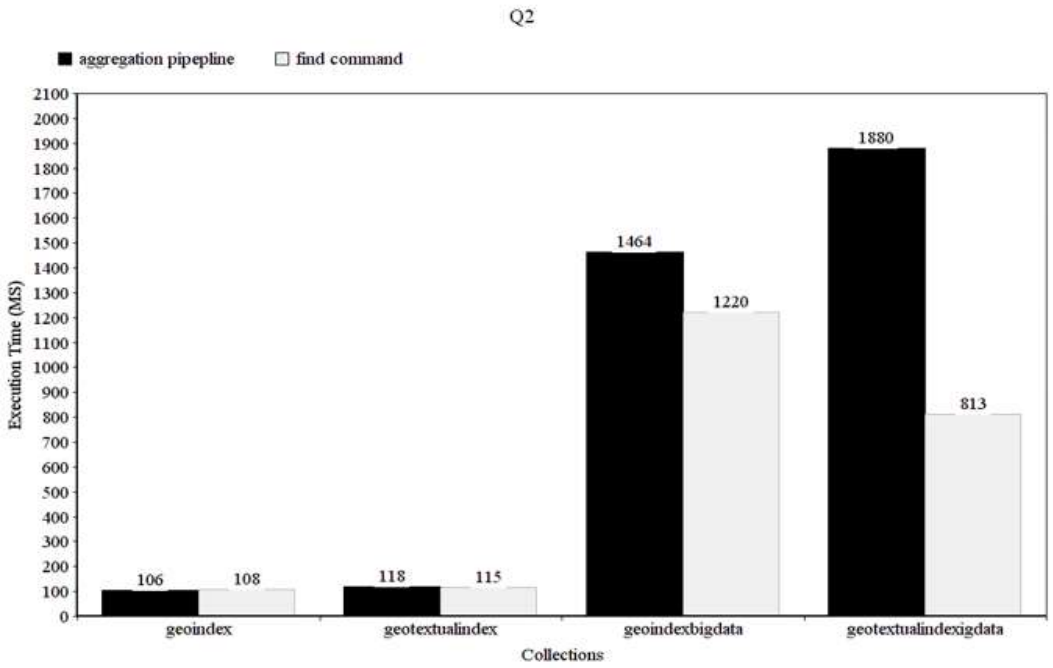
Αποτελέσματα ερωτημάτων:

Στα collection geoindex και geoindexbigdata, έχει χρησιμοποιηθεί μόνο χωρική ευρετηρίαση. Για το collections geotextualindex και geotextualindexbigdata, έχει εφαρμοστεί η χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB.



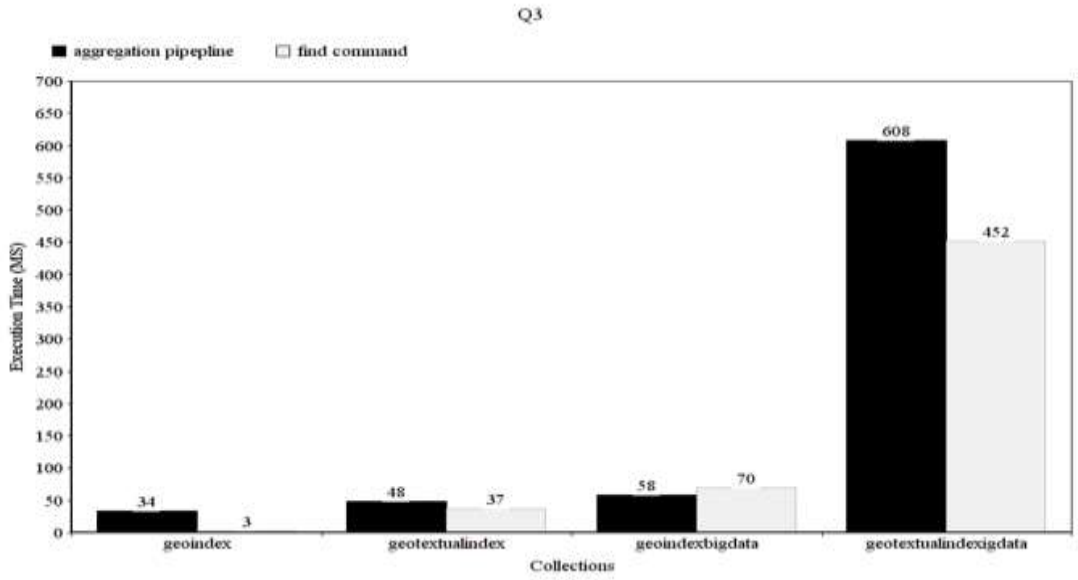
Εικόνα 5.2 Q1 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.2 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q1 με χρήση της εντολής find και με χρήση aggregation pipeline στα collections geoindex, geotextualindex, geoindexbigdata και geotextualindexbigdata. Στο collection geoindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν σχεδόν ίδια απόδοση. Με χρήση της εντολής find το ερώτημα εκτελείται σε 68 ms και με του aggregation pipeline στα 68 ms., οι δυο τρόποι εκτέλεσης παρουσιάζουν μια σημαντική διαφορά στον χρόνο εκτέλεσης, της τάξης των 9 ms. Με χρήση της εντολής find το ερώτημα εκτελείται σε 44 ms και με του aggregation pipeline στα 53 ms. Βέβαια, αυτό που αξίζει να σημειωθεί ότι τα αποτελέσματα της χωρο-κειμενικής ευρετηρίασης είτε μέσω της εντολής find είτε μέσω του aggregation pipeline παρουσιάζουν καλύτερα αποτελέσματα, από την χρήση μόνο χωρικής ευρετηρίασης. Για τα Big Data collection, η κατάσταση είναι παρόμοια. Στο collection geoindexbigdata, οι δυο τρόποι εκτέλεσης έχουν σχεδόν ίδια αποτελέσματα για τον χρόνο εκτέλεσης του ερωτήματος. Με χρήση της εντολής find το ερώτημα εκτελείται σε 994 ms και με του aggregation pipeline στα 998 ms. Για το collection geotextualindexbigdata, οι δυο τρόποι εκτέλεσης παρουσιάζουν μεγάλη διαφορά στον χρόνο εκτέλεσης. Με χρήση της εντολής find το ερώτημα εκτελείται σε 545 ms και με του aggregation pipeline στα 717 ms. Συνολικά η εντολή find έχει παρουσιάζει απόδοση. Η χρήση χωρο-κειμενικής ευρετηρίασης εμφανίζει καλύτερα αποτελέσματα.



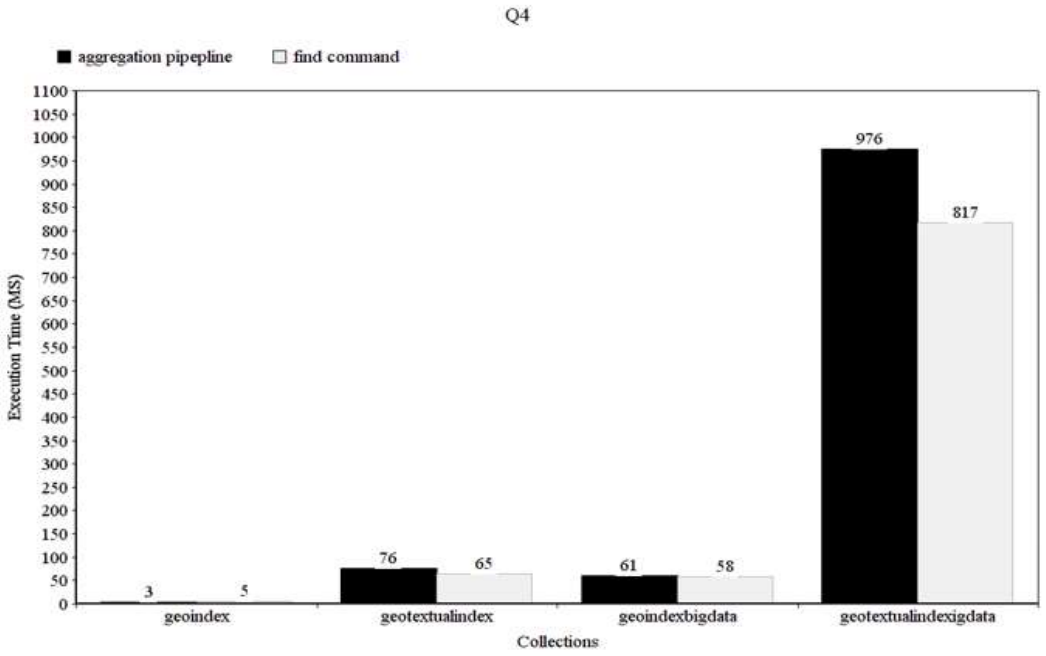
Εικόνα 5.3 Q2 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.3 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q2 με χρήση της εντολής find και με χρήση aggregation pipeline στα collections geoindex, geotextualindex, geoindexbigdata και geotextualindexbigdata. Στο collection geoindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν σχεδόν ίδια απόδοση. Με χρήση της εντολής find το ερώτημα εκτελείται σε 108 ms και με του aggregation pipeline στα 106 ms. Για το collection geotextualindex, οι δυο τρόποι εκτέλεσης εμφανίζουν μια μικρή διαφορά 3 ms. Με χρήση της εντολής find το ερώτημα εκτελείται σε 115 ms και με του aggregation pipeline στα 118 ms. Για τα Big Data collection, η κατάσταση είναι λίγο διαφορετική. Στο collection geoindexbigdata, οι δυο τρόποι εκτέλεσης παρουσιάζουν σημαντική διαφορά στο χρόνο εκτέλεσης του ερωτήματος. Μέσω της εντολής find το ερώτημα, εκτελείται πιο γρήγορα. Με χρήση της εντολής find το ερώτημα εκτελείται σε 1220 ms και με του aggregation pipeline στα 1464 ms. Για το collection geotextualindexbigdata και σε αυτήν την περίπτωση, σημειώνεται υπεροχή της εντολής find με μεγάλη διαφορά Με χρήση της εντολής find το ερώτημα εκτελείται σε 813 ms και με του aggregation pipeline στα 1880 ms. Όπως στην περίπτωση του ερωτήματος Q1, έτσι και στο Q2 η εντολή find εμφανίζει καλύτερη απόδοση.



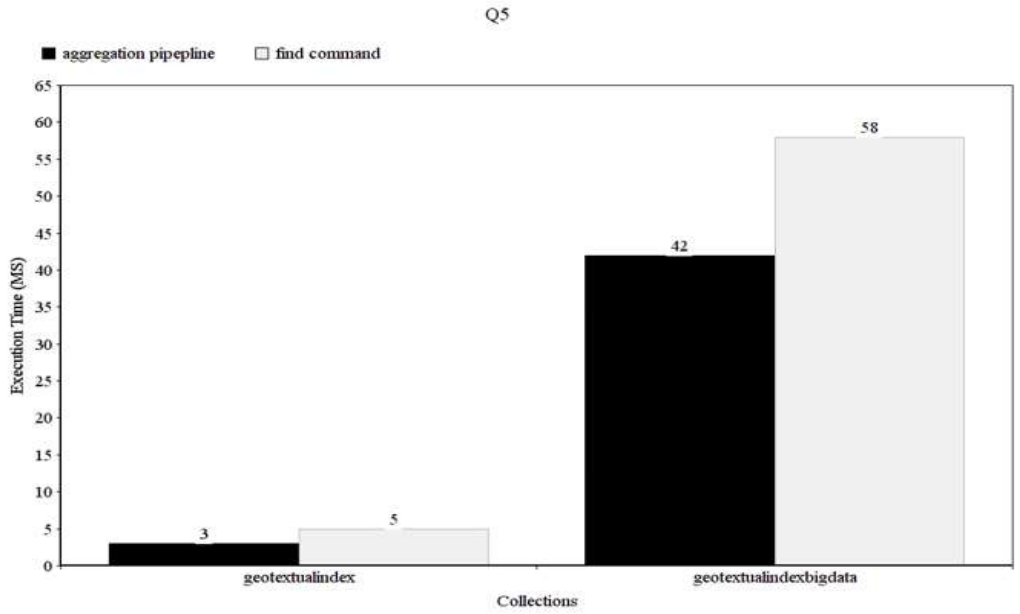
Εικόνα 5.4 Q3 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.4 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q3 με χρήση της εντολής find και με χρήση aggregation pipeline στα collections geoindex, geotextualindex, geoindexbigdata και geotextualindexbigdata. Στο collection geoindex, οι δυο τρόποι παρουσιάζουν μια σημαντική διαφορά στον χρόνο εκτέλεσης. Με χρήση της εντολής find το ερώτημα εκτελείται σε 3ms και με του aggregation pipeline στα 34 ms. Για το collection geotextualindex, η χρήση της εντολής find και εδώ παρουσιάζει καλύτερα αποτελέσματα. Με χρήση της εντολής find το ερώτημα εκτελείται σε 37 ms και με του aggregation pipeline στα 48 ms. Βέβαια, αυτό που αξίζει να σημειωθεί ότι τα αποτελέσματα της χωρο-κειμενικής ευρετηρίασης είτε μέσω της εντολής find είτε μέσω του aggregation pipeline παρουσιάζουν χειρότερα αποτελέσματα, από την χρήση μόνο χωρικής ευρετηρίασης. Για τα Big Data collection, η κατάσταση αλλάζει λίγο. Στο collection geoindexbigdata, η χρήση του aggregation pipeline έχει καλύτερη απόδοση. Με χρήση της εντολής find το ερώτημα εκτελείται σε 70 ms και με του aggregation pipeline στα 58 ms. Για το collection geotextualindexbigdata, οι δυο τρόποι εκτέλεσης παρουσιάζουν μεγάλη διαφορά στον χρόνο εκτέλεσης. Με χρήση της εντολής find το ερώτημα εκτελείται σε 452 ms και με του aggregation pipeline στα 608 ms. Συνολικά η εντολή find έχει παρουσιάζει απόδοση. Η χρήση μόνο χωρικής ευρετηρίασης εμφανίζει καλύτερα αποτελέσματα.



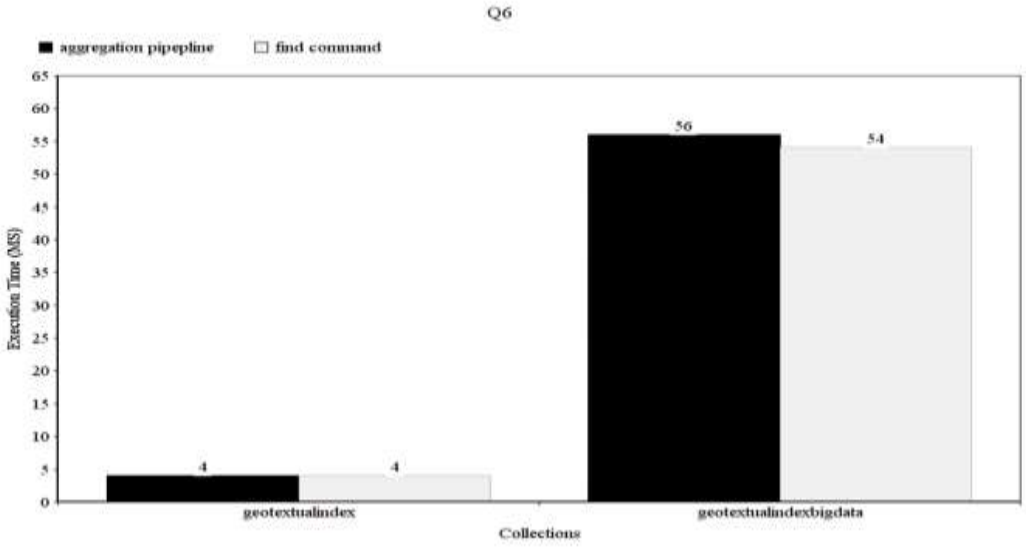
Εικόνα 5.5 Q4 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.5 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q4 με χρήση της εντολής find και με χρήση aggregation pipeline στα collections geoindex, geotextualindex, geoindexbigdata και geotextualindexbigdata. Στο collection geoindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν σχεδόν ίδια απόδοση. Με χρήση της εντολής find το ερώτημα εκτελείται σε 5 ms και με του aggregation pipeline στα 3 ms. Το aggregation pipeline εμφανίζει καλύτερα αποτελέσματα. Για το collection geotextualindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν μια σημαντική διαφορά στον χρόνο εκτέλεσης, της τάξης των 11 ms.Υπάρχει υπεροχή της εντολής find. Με χρήση της εντολής find το ερώτημα εκτελείται σε 65 ms και με του aggregation pipeline στα 76 ms. Τα αποτελέσματα της χωρο-κειμενικής ευρετηρίασης είτε μέσω της εντολής find είτε μέσω του aggregation pipeline παρουσιάζουν χειρότερα αποτελέσματα, από την χρήση μόνο χωρικής ευρετηρίασης. Στο collection geoindexbigdata, οι δυο τρόποι εκτέλεσης έχουν σχεδόν ίδια αποτελέσματα για τον χρόνο εκτέλεσης του ερωτήματος. Με χρήση της εντολής find το ερώτημα εκτελείται σε 58 ms και με του aggregation pipeline στα 61 ms. Για το collection geotextualindexbigdata, οι δυο τρόποι εκτέλεσης παρουσιάζουν μεγάλη διαφορά στον χρόνο εκτέλεσης. Με χρήση της εντολής find το ερώτημα εκτελείται σε 817 ms και με του aggregation pipeline στα 976 ms. Συνολικά η εντολή find έχει παρουσιάζει απόδοση. Η χρήση μόνο χωρικής ευρετηρίασης εμφανίζει καλύτερα αποτελέσματα.



Εικόνα 5.6 Q5 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.6 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q5 με χρήση της εντολής find και με χρήση aggregation pipeline στα collections geotextualindex και geotextualindexbigdata. Για το collection geotextualindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν μια μικρή διαφορά στον χρόνο εκτέλεσης, της τάξης των 2 ms. Με χρήση της εντολής find το ερώτημα εκτελείται σε 5 ms και με του aggregation pipeline στα 3 ms. Για το collection geotextualindexbigdata και σε αυτήν την περίπτωση, σημειώνεται υπεροχή της εκτέλεσης μέσω aggregation pipeline με σημαντική διαφορά. Μέσω της εντολής find το ερώτημα εκτελείται σε 58 ms και με του aggregation pipeline στα 42 ms. Το aggregation pipeline εμφανίζει καλύτερη απόδοση.



Εικόνα 5.7 Q6 - Σύγκριση aggregation pipeline με την εντολή find

Στο διάγραμμα της Εικόνας 5.7 παρουσιάζονται τα αποτελέσματα από την εκτέλεση του ερωτήματος Q6 με χρήση της εντολής find και με χρήση aggregation pipeline, στα collections geotextualindex και geotextualindexbigdata. Για το collection geotextualindex, οι δυο τρόποι εκτέλεσης παρουσιάζουν ίδιο χρόνο εκτέλεσης 4 ms. Για το collection geotextualindexbigdata, σημειώνεται μια μικρή υπεροχή της εντολής find. Με χρήση της εντολής find το ερώτημα εκτελείται σε 54 ms και με του aggregation pipeline στα 56 ms. Οι 2 τρόποι εκτέλεσης παρουσιάζουν παρόμοια απόδοση.

Συμπεράσματα

- Στα περισσότερα ερωτήματα που εκτελέστηκε η κειμενική αναζήτηση, βάσει του τελεστή $or. Η υλοποίηση μέσω της εντολής find ήταν η καλύτερη.
- Για τα ερωτήματα που εκτελέστηκε που εκτελέστηκε η κειμενική αναζήτηση, βάσει του τελεστή $and. Η υλοποίηση μέσω της εντολής aggregation pipeline ήταν η καλύτερη.
- Για τα περισσότερα ερωτήματα που εκτελέστηκε η κειμενική αναζήτηση, βάσει του τελεστή $and. Παρουσιάστηκε καλύτερη απόδοση στα queries που χρησιμοποιήσαν μόνο χωρική ευρετηρίαση.
- Τα περισσότερα ερωτήματα με γεωμετρικό σχήμα κύκλου είχαν καλύτερη απόδοση όταν χρησιμοποιήθηκε μόνο χωρική ευρετηρίαση.
- Τα περισσότερα ερωτήματα με γεωμετρικό σχήμα πολυγώνου όπου πραγματοποίησαν το match της κειμενικής αναζήτησης βάσει του τελεστή $or, εμφάνισαν καλύτερη απόδοση όταν χρησιμοποιήθηκε χωρική και κειμενική ευρετηρίαση.
- Η χρήση της χωρο-κειμενικής ευρετηρίασης που διαθέτει η MongoDB, δεν οδηγεί πάντα σε καλύτερα αποτελέσματα. Όταν αυξάνει το πλήθος των λέξεων που ικανοποιούν την συνθήκη αναζήτησης, τότε αυξάνεται ο χρόνος εκτέλεσης σε σχέση με την χρήση μόνο χωρικής ευρετηρίασης.
- Τα περισσότερα ερωτήματα εμφάνισαν καλύτερη απόδοση όταν χρησιμοποιήθηκε μόνο χωρική ευρετηρίαση.
- Τα περισσότερα ερωτήματα εμφάνισαν συνολικά, καλύτερη απόδοση με την εντολή find.

Η MongoDB παρέχει την δυνατότητα είτε επίπεδης (2d index) είτε σφαιρικής (2dsphere index) ευρετηρίασης των χωρικών δεδομένων. Για να επιλεχθεί ποιος από τους 2 τρόπους ευρετηρίασης είναι αποδοτικότερος για τα collection geoindex και geotextualindex index. Ορίστηκαν τα παρακάτω 6 κουτιά. Η σειρά των κουτιών δίνεται κατά αύξουσα σειρά βάσει το μέγεθος που έχει το κάθε κουτί.

- **Box1** [-40.937563, 60.19198813], [**-65.6995,** 20.974964].
- **Box2** [-40.937563, 60.19198813], [**-66.6995**, 20.974964].
- **Box3** [-40.937563, 60.19198813], [**-67.6995**, 20.974964].
- **Box4** [-40.937563, 60.19198813], [**-68.6995**, 20.974964].
- **Box5** [-40.937563, 60.19198813], [**-69.6995**, 20.974964].
- **Box6** [-40.937563, 60.19198813], [**-70.6995**, 20.974964].

Εικόνα 5.8 Box1

Εικόνα 5.9 Box2

Εικόνα 5.10 Box3

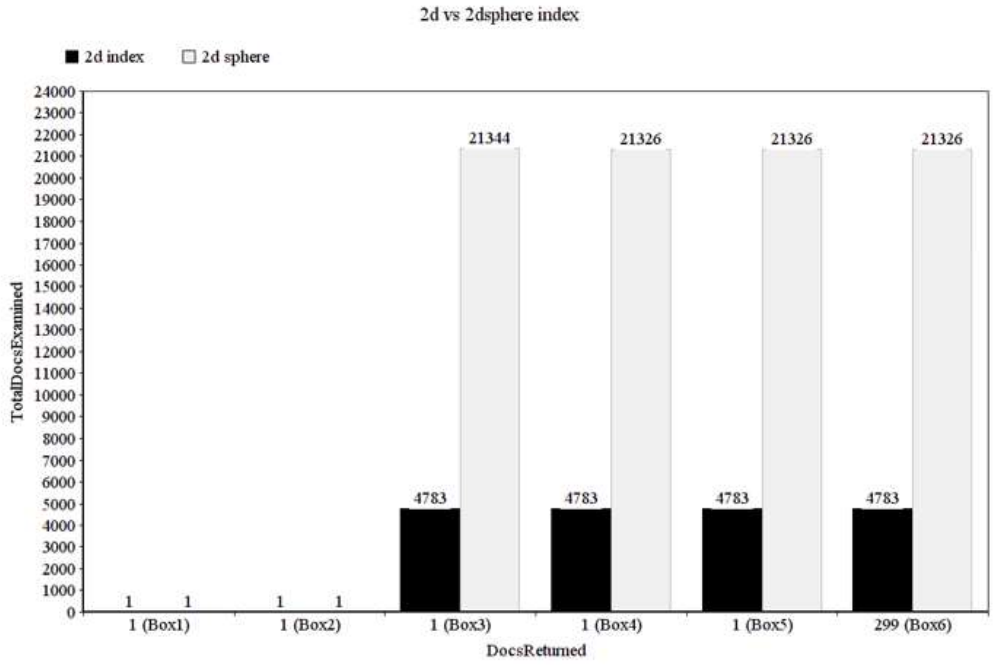
Εικόνα 5.11 Box4

Εικόνα 5.12 Box5

Εικόνα 5.13 Box6

Καθώς μεγαλώνουν τα κουτιά περιέχουν και περισσότερα δεδομένα. Στη συνέχεια εφαρμόζεται στο collection geoindex, 2d και 2dsphere ευρετηρίαση ξεχωριστά. Το κάθε κουτί αποτελεί ένα ερώτημα. Ο στόχος είναι να μετρηθεί πόσα documents εξετάζει και επιστρέφει η κάθε ευρετηρίαση για το κάθε ερώτημα. Για το συγκεκριμένο πείραμα, επειδή εξετάζεται μόνο η χωρική ευρετηρίαση, το collection geotextualindex θα έχει ακριβώς την ίδια συμπεριφορά με το collection geoindex.

Ακολουθεί διάγραμμα με συγκριτικά αποτελέσματα των παραπάνω τρόπων ευρετηρίασης 2d και 2dsphere στο collection geoindex.



Εικόνα 5.14 Σύγκριση 2d με 2dsphere ευρετηρίαση

Από το διάγραμμα της εικόνας 5.14 παρατηρείται ότι, η ευρετηρίαση 2d εξετάζει λιγότερα documents καθώς τα κουτιά μεγαλώνουν, άρα εμφανίζει καλύτερη απόδοση. Αξίζει να σημειωθεί ότι, κατά την ευρετηρίαση 2d στα box3, box4, box5 και box6 εξετάζονται 4783 documents και επιστρέφονται 1, 1, 1, 299 documents

αντίστοιχα. Το κάθε box μπορεί να ακουμπάει κελία που περιέχουν πολλά documents αλλά δεν σημαίνει ότι όλα τα documents περιέχονται μέσα στο box. Στο τέλος επιστρέφονται τα documents που βρίσκονται εντός του εκάστοτε box αλλά εξετάζονται όλα τα documents που περιέχονται στα κελιά που ακουμπάει το box.

Στην ευρετηρίαση 2dsphere παρουσιάζεται το εξής. Ενώ το box3 είναι μικρότερο από το box 4 φαίνεται να εξετάζει περισσότερα documents. Η ευρετηρίαση 2dsphere παρουσιάζει αισθητά χειρότερη απόδοση από την ευρετηρίαση 2d. Στα ερωτήματα που θα ακολουθήσουν για την χωρική ευρετηρίαση χρησιμοποιείται 2d.

Στη συνέχεια δημιουργείται το hilbertcollection όπου σαν τεχνική ευρετηρίασης, χρησιμοποιείται η μεθοδολογία που αναπτύχθηκε στο κεφάλαιο 4. Στο πεδίο hilbertindex εφαρμόζεται Single index. Το κάθε χωρο-κειμενικό αντικείμενο έχει χωριστεί σε τόσα documents όσο είναι το πλήθος των keywords που περιείχε. Συνολικά το collection περιείχε 239694 documents.

Έπειτα, δημιουργούνται τα ακόλουθα 3 κουτιά και 3 λίστες. Παρατίθενται κατά αύξουσα σειρά ανά κατηγορία βάσει το μέγεθος τους.

- **Box1** [-30.937563, 60.19198813] [-70.6995, 20.974964] (Μικρό Box).
- **Box2** [-82.937563, 40.19198813] [-50.6995, 20.974964] (Μεσαίο Box).
- **Box3** [-98.937563, 60.191988133] [-50.6995, 20.974964] (Μεγάλο Box).
- **List1** {Cambodian, Lebanese, Persian} (Σπάνιες λέξεις).
- **List2** {Greek, Mediterranean, Grill } (Κανονική συχνότητα λέξεων).
- **List3** {Burgers, Pizza , Sandwiches } (Συχνές λέξεις).

Ο παρακάτω πίνακας περιέχει τον αριθμό των documents που περιέχουν τα κουτιά και οι λίστες ανά collection.

	Geoindex	**Geotextualindex**	**Hilbertcollection**
Box1	299	299	820
Box2	14793	14793	44327
Box3	57122	57122	169148
List1	184	184	185
List2	3735	3735	4295
List3	34515	34515	46254

Πίνακας 5.2 Κουτιά και λίστες ερωτημάτων

Παρακάτω δίνονται τα ερωτήματα της πειραματικής διαδικασίας:

Q1 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q2 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q3 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q4 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q5 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q6 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q7 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q8 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

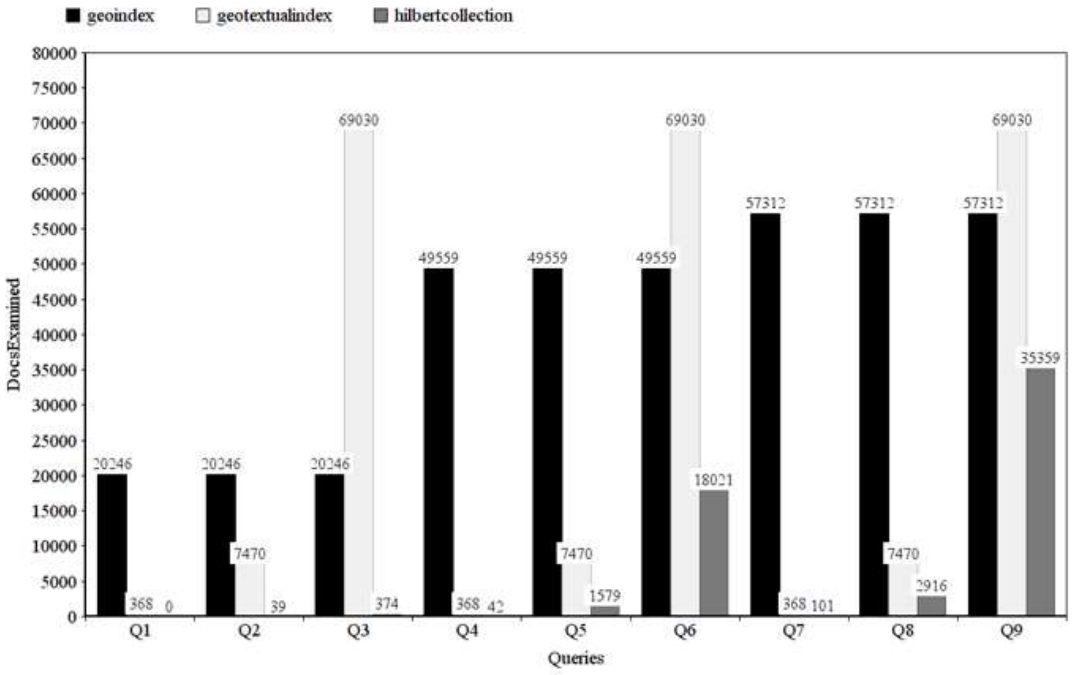
Q9 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Τα παραπάνω ερωτήματα υλοποιήθηκαν για την MongoDB χρησιμοποιώντας **aggregation pipeline**.

Για την μεθοδολογία Hilbert Curve δόθηκαν οι διαστάσεις για το x -180 μέχρι 180 και για το y -90 έως 90. Χρησιμοποιήθηκε αναπαράσταση με αναπαράσταση 10 bits.

Αποτελέσματα ερωτημάτων:

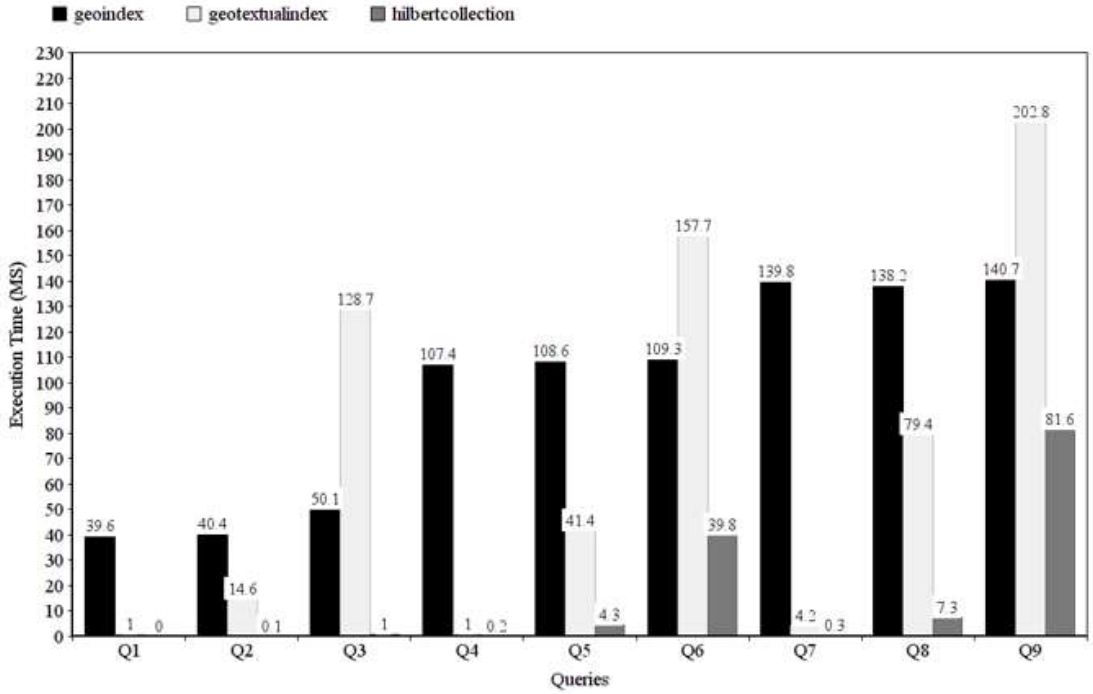
Για το collection geoindex έχει χρησιμοποιηθεί μόνο χωρική ευρετηρίαση, για το collection geotextualindex έχει χρησιμοποιηθεί η χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB και στο hilbertcollection έχει εφαρμοστεί η χωρο-κειμενική ευρετηρίαση που παρουσιάστηκε στο κεφάλαιο 4.



Εικόνα 5.15 MongoDB κεντρικοποιημένη υποδομή - σύγκριση εγγράφων που εξετάζονται

Στο γράφημα της εικόνας 5.15 συγκρίνονται τα ερωτήματα Q1-Q9 για τα collections geoindex, geotextualindex και hilbertindex, σχετικά με τα documents που εξετάστηκαν κατά την εκτέλεση των ερωτημάτων. Για το ερώτημα Q1 που περιέχει το μικρό box και την λίστα λέξεων με την μικρή συχνότητα, στο hilbertcollection παρουσιάζεται με μεγάλη διαφορά η καλύτερη απόδοση, εξετάζοντας 0 documents κατά την εκτέλεση του Q1. Αμέσως μετά στο geotextualindex εξετάζονται 368 documents και με μεγάλη διαφορά η χειρότερη απόδοση, παρουσιάζεται στο geoindex εξετάζοντας 20246. Για το ερώτημα Q2, η κατάσταση παραμένει ίδια. Το Q2 περιέχει το μικρό box και την λίστα λέξεων κανονικής συχνότητας. Στο hilbertcollection παρουσιάζεται συντριπτικά η καλύτερη απόδοση, εξετάζοντας 39 documents. Στο geotextualindex έχει μειωθεί η διαφορά στα documents που εξετάζονται με το geoindex συγκινώντας το με το Q1. Ωστόσο, η διαφορά παραμένει μεγάλη. Στο geoindex μακράν εμφανίζεται η χειρότερη απόδοση. Για το Q3 που περιέχει το μικρό box και τις λέξεις με μεγάλη συχνότητα, παρουσιάζονται κάποιες αλλαγές σε σχέση με το Q1 και Q2. Στο hilbertcollection σταθερά παρουσιάζεται η καλύτερη απόδοση εξετάζοντας 374 documents. Σε αυτήν την περίπτωση όμως στο geotextualindex παρουσιάζεται η χειρότερη απόδοση μέχρι στιγμής, εξετάζοντας 69030 documents ενώ στο geoindex εξετάζονται 20246 documents. Για το Q4 με το μεσαίο μέγεθος box και την λίστα λέξεων με την μικρή συχνότητα, πάλι στο hilbertcollection εμφανίζεται η καλύτερη απόδοση, εξετάζοντας 42 documents. Η χειρότερη απόδοση εμφανίζεται στο geoindex εξετάζοντας 49559 documents. Στο geotextualindex εξετάζονται 368 documents. Δεν αλλάζει κάτι ουσιαστικό στο Q5 με το μεσαίο μέγεθος box και την λίστα λέξεων με κανονική συχνότητα. Για το Q6 με το μεσαίο μέγεθος box και την λίστα λέξεων μεγάλης συχνότητας, σταθερά με μεγάλη διαφορά στο hilbertcollection εξετάζονται τα λιγότερα documents 18021. Αυτό που

αλλάζει είναι ότι στο geotextualindex εξετάζονται τα περισσότερα documents 69030 και εμφανίζεται η χειρότερη απόδοση. Η ίδια εικόνα επικρατεί και στο Q9. Για το Q7 και Q8. Η χειρότερη επίδοση εμφανίζεται στο geoindex και η καλύτερη με διαφορά και πάλι στο hilbertcollection. Συμπερασματικά, η χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε στο κεφάλαιο 4 εμφανίζει καλύτερα τόσο σε ερωτήματα μεγάλης γεωγραφικής περιοχής, όσο και ερωτήματα συχνών λέξεων αναζήτησης, καθώς και ο συνδυασμός τους. Η χωρο-κειμενική ευρετηρίαση που προσφέρει η MongoDB παρουσιάζει την χαμηλή απόδοση σε αναζητήσεις λέξεων μεγάλης συχνότητας. Ενώ η χρήση μόνο χωρικής ευρετηρίασης που προσφέρει η MongoDB παρουσιάζει χαμηλή απόδοση σε ερωτήματα μεγάλης γεωγραφικής περιοχής που περιέχουν πολλά αντικείμενα.



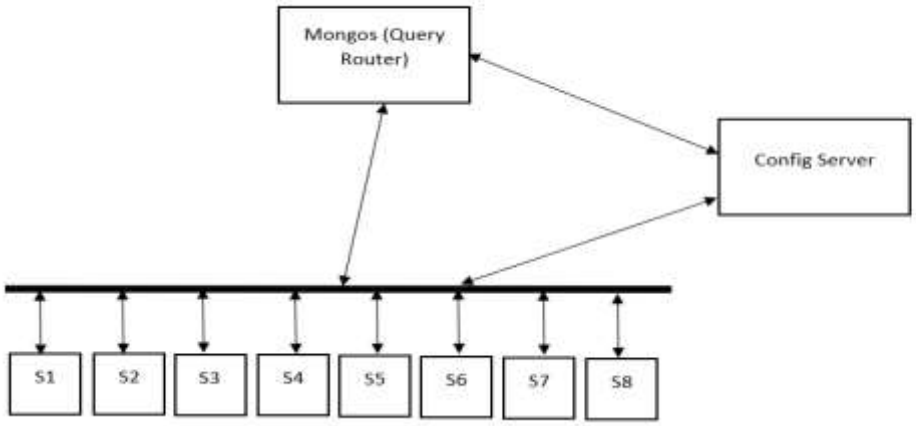
Εικόνα 5.16 MongoDB κεντρικοποιημένη υποδομή - σύγκριση χρόνου εκτέλεσης ερωτημάτων

Στο γράφημα της εικόνας 5.16 συγκρίνονται τα ερωτήματα Q1-Q9 για τα collections geoindex, geotextualindex και hilbertindex, σχετικά με των χρόνο εκτέλεσης των ερωτημάτων. Για το ερώτημα Q1, στο hilbertcollection παρουσιάζεται με μεγάλη διαφορά ο καλύτερος χρόνος εκτέλεσης με 0 ms. Αμέσως μετά στο geotextualindex ο χρόνος εκτέλεσης είναι 1 ms και με μεγάλη διαφορά η χειρότερη απόδοση, παρουσιάζεται στο geoindex με χρόνο 39,6 ms. Για το ερώτημα Q2, η κατάσταση παραμένει ίδια. Στο hilbertcollection εμφανίζεται η καλύτερη απόδοση με χρόνο 0,1. Στο geoindex μακράν εμφανίζεται η χειρότερη απόδοση. Για το Q3 στο hilbertcollection σταθερά παρουσιάζεται η καλύτερη απόδοση με χρόνο 1 ms. Σε αυτήν την περίπτωση όμως στο geotextualindex παρουσιάζεται η χειρότερη απόδοση μέχρι στιγμής, με χρόνο 128 ms. Στο geoindex ο χρόνος εκτέλεσης είναι 50,1 ms. Για το Q4 και πάλι στο hilbertcollection εμφανίζεται η καλύτερη απόδοση με χρόνο 0,2

ms. Η χειρότερη απόδοση εμφανίζεται στο geoindex με χρόνο 107,4 ms. Στο geotextualindex ο χρόνος εκτέλεσης ανέρχεται στο 1 ms. Δεν αλλάζει κάτι ουσιαστικό στο Q5. Για το Q6 και σε αυτήν την περίπτωση, στο hilbertcollection παρουσιάζεται ο καλύτερος χρόνος εκτέλεσης με 39,8 ms. Αυτό που αλλάζει είναι ότι στο geotextualindex με χρόνο 157,7 εμφανίζεται η χειρότερη απόδοση. Η ίδια εικόνα επικρατεί και στο Q9. Για το Q7 και Q8. Η χειρότερη επίδοση εμφανίζεται στο geoindex και η καλύτερη με διαφορά και πάλι στο hilbertcollection. Σύμφωνα με τους χρόνους εκτέλεσης αποδεικνύεται ότι, χωρο-κειμενική ευρετηρίαση που προσφέρει η MongoDB παρουσιάζει την χαμηλή απόδοση σε αναζητήσεις λέξεων μεγάλης συχνότητας. Ενώ η χρήση μόνο χωρικής ευρετηρίασης που προσφέρει η MongoDB παρουσιάζει χαμηλή απόδοση σε ερωτήματα μεγάλης γεωγραφικής περιοχής που περιέχουν πολλά αντικείμενα.

5.1.2 MongoDB σε κατανεμημένο περιβάλλον

Στα πλαίσια της πειραματικής διαδικασίας, δημιουργήθηκε ένα cluster MongoDB περιβάλλον. Η αρχιτεκτονική του φαίνεται στη παρακάτω Εικόνα 5.17.



Εικόνα 5.17 MongoDB κατανεμημένη υποδομή

Στον παρακάτω πίνακα αναγράφονται αναλυτικά οι πόροι που δόθηκαν στη cluster υποδομή

VM	RAM	CPU	HARD DISK
Mongos	16384 MB	8	124 GB
Config Server	4096 MB	2	100 GB
S1	8192 MB	4	100 GB
S2	8192 MB	4	100 GB
S3	8192 MB	4	100 GB
S4	8192 MB	4	100 GB
S5	8192 MB	4	100 GB
S6	8192 MB	4	100 GB
S7	8192 MB	4	100 GB
S8	4096 MB	2	100 GB

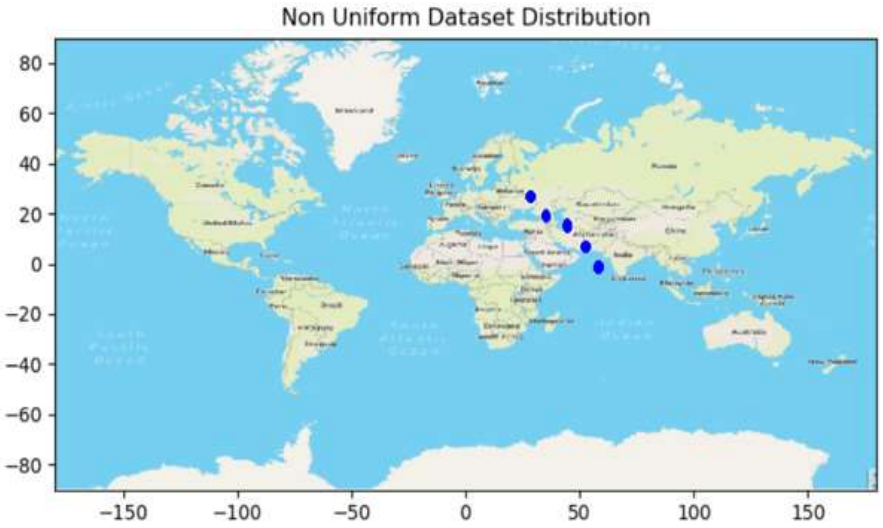
Πίνακας 5.3 Πόροι κατανεμημένης υποδομής MongoDB

Δημιουργήθηκαν 2 συνθετικά Datasets το uniform_dataset.txt και το non_uniform_dataset.txt. Το κάθε ένα περιείχε 12.000.000 εγγραφές και το μέγεθος του είναι 691 MB. Τα δεδομένα στο πρώτο Dataset ακολουθούν ομοιόμορφη κατανομή ενώ στο δεύτερο ακολουθούν μη ομοιόμορφη κατανομή. Για το δεύτερο Dataset τα δεδομένα ακολουθούν κανονική κατανομή σχηματίζοντας 5 clusters. Για κάθε αντικείμενο των 2 Datasets έχει ανατεθεί ένας τυχαίος αριθμός από keywords στο διάστημα [1,5] . Τα keywords έχουν επιλεχθεί από ένα λεξικό με 9000 εγγραφές χρησιμοποιώντας Zipf κατανομή. Έτσι, δημιουργήθηκαν 3 κατηγορίες λέξεων χαμηλής, μεσαίας και υψηλής συχνότητας επιλογής. Το συνολικό μέγεθος του λεξικού είναι 90,5 KB.

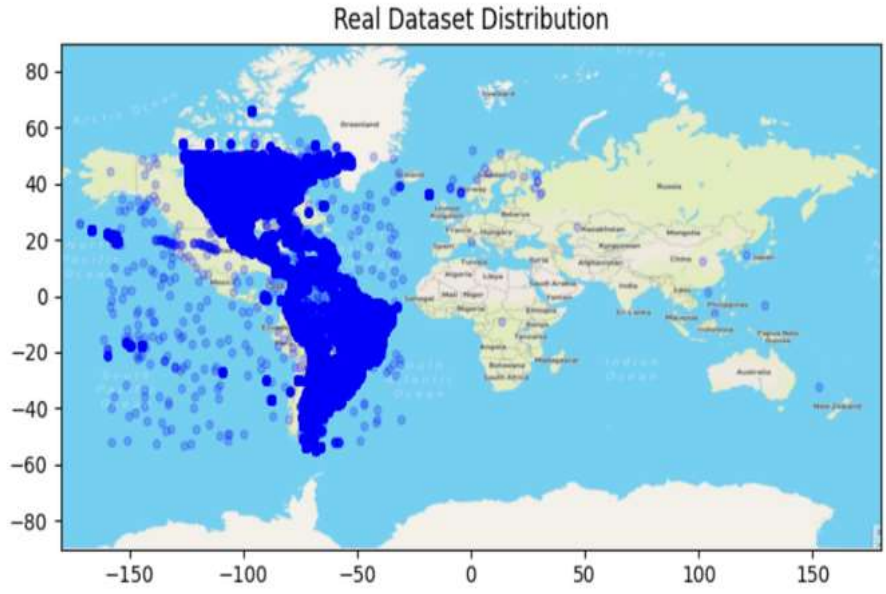
Στα πλαίσια της πειραματικής διαδικασίας χρησιμοποιήθηκε και ένα Dataset με πραγματικά δεδομένα από tweets. Το real_dataset.txt περιείχε 9490674 εγγραφές και το μέγεθος του είναι 938 MB.

Παρακάτω ακολουθεί η οπτική απεικόνιση των 3 Datasets στο χάρτη.

Εικόνα 5.18 MongoDB κατανεμημένη υποδομή - ομοιόμορφη κατανομή δεδομένων



Εικόνα 5.19 MongoDB κατανεμημένη υποδομή – μη ομοιόμορφη κατανομή δεδομένων

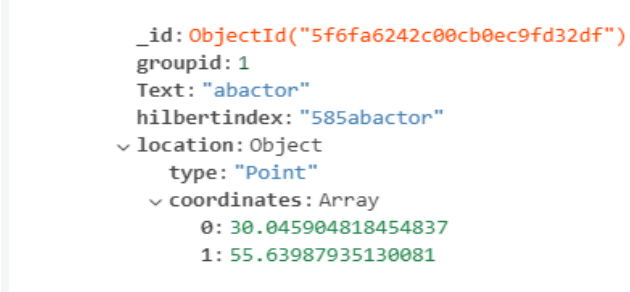


Εικόνα 5.20 MongoDB κατανεμημένη υποδομή – κατανομή πραγματικών δεδομένων

Τα 3 datasets έγιναν εισαγωγή στη MongoDB cluster υποδομή. Δημιουργήθηκαν 6 collections το uniform_collection και το uniform_default_index περιέχουν δεδομένα από το uniform_dataset.txt, το non_uniform_collection και non_uniform_default_index περιέχουν δεδομένα από το non_unoform_dataset.txt και το real_data_collection με το real_data_default_index περιέχουν δεδομένα από το real_dataset.txt . Για τα collections uniform_collection, non_uniform_collection και real_data_collection ισχύει ότι το κάθε χωρο-κειμενικό αντικείμενο έχει χωριστεί σε τόσα documents όσο είναι το πλήθος των keywords που περιέχει η κάθε εγγραφή είτε του πρώτου είτε του δεύτερου Dataset. Συνολικά το uniform_collection περιέχει 29,998,598 documents το non_uniform_collection και το real_data_collection

περιέχει 29998278 documents και το real_data_collection περιέχει 54577876 documents. Γι' αυτά τα collections χρησιμοποιήθηκε το geotextualindex με χρήση της τεχνικές Hilbert.

Παρακάτω ακολουθεί η μορφή που έχουν τα documents για τα collections uniform_collection, non_uniform_collection και real_data_collection.

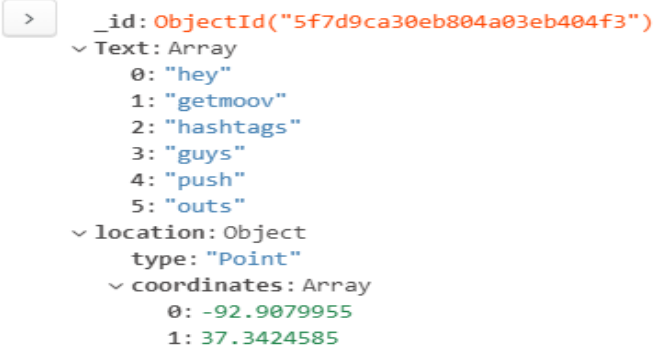


Εικόνα 5.21 MongoDB κατανεμημένη υποδομή – πρώτη μορφή εγγράφων

Σαν Sharding Strategy και για τα collections uniform_collection, non_uniform_collection και real_data_collection, χρησιμοποιήθηκε το πεδίο hilbertindex όπου χρησιμοποιήθηκε και σαν index.

Για τα collections uniform_default_index, non_uniform_default_index, real_data_default_index ισχύει ότι κάθε χωρο-κειμενικό αντικείμενο είναι ένα document, οι συντεταγμένες βρίσκονται στο πεδίο coordinates και τα keywords στο πεδίο Text. Συνολικά το uniform_default_index περιέχει 12000000 documents το non_uniform_default_index περιέχει 12000000 documents και το real_data_default_index περιέχει 9490674 documents. Γι' αυτά τα collections χρησιμοποιήθηκε το geotextualindex που προσφέρει η MongoDB.

Παρακάτω ακολουθεί η μορφή που έχουν τα documents για τα collections uniform_default_index, non_uniform_default_index, real_data_default_index.



Εικόνα 5.22 MongoDB κατανεμημένη υποδομή – δεύτερη μορφή εγγράφων

Σαν Sharding Strategy για τα collections uniform_default_index, non_uniform_default_index, real_data_default_index, χρησιμοποιήθηκε το πεδίο coordinates όπου χρησιμοποιήθηκε και σαν index μαζί με το πεδίο Text.

Στη συνέχεια της πειραματικής διαδικασίας ακολουθούν 3 υπό-ενότητες μια για κάθε κατηγορία δεδομένων.

Uniform Data

Δημιουργούνται τα ακόλουθα 4 κουτιά και 3 λίστες. Παρατίθενται κατά αύξουσα σειρά ανά κατηγορία βάσει το μέγεθος τους.

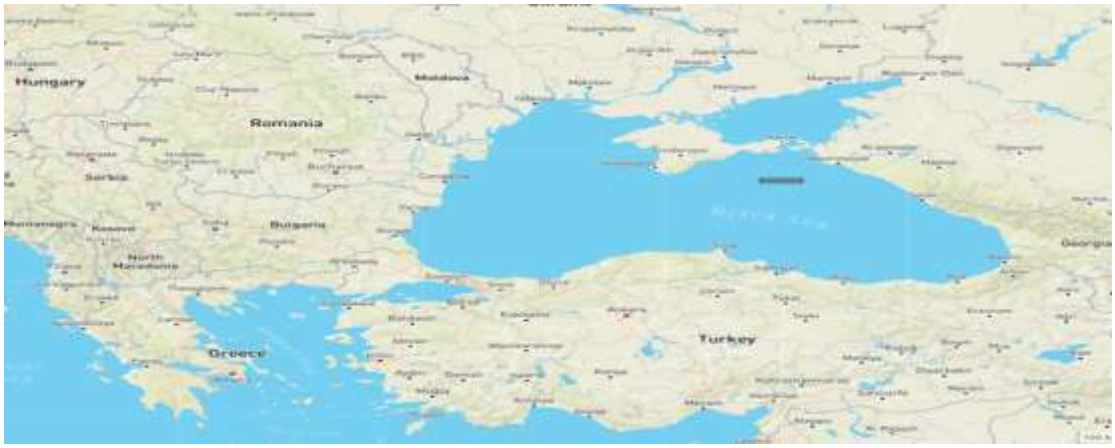
- **Box1** [35.9482421875, 44.25713822211053] [35.9380078125, 44.14029065139799]
- **Box2** [51.958203125, 44.77131167976407] [51.88906249999999, 44.93306116240612]
- **Box3** [35.955859375, 35.982170703266075] [35.1001953125, 36.1023764487364]
- **List1** { actus, accustom, accuser} (Σπάνιες λέξεις)
- **List2** { abatements, abdicated, abdicator} (Κανονική συχνότητα λέξεων)
- **List3** {abaft, abalone, abandon} (Συχνές λέξεις).

Ο παρακάτω πίνακας περιέχει τον αριθμό των documents που περιέχουν τα κουτιά και οι λίστες ανά collection.

	uniform_collection	**uniform_default_index**
Box1	109	47
Box2	1221	478
Box3	10222	4073
List1	8700	8063
List2	130598	130203
List3	965537	941451

Πίνακας 5.4 Αριθμός εγγραφών στα collections με την ομοιόμορφη κατανομή δεδομένων

Ακολουθεί η οπτική απεικόνιση των boxes.



Εικόνα 5.23 Uniform data box1

Εικόνα 5.24 Uniform data box2

Εικόνα 5.25 Uniform data box3

Παρακάτω δίνονται τα ερωτήματα της πειραματικής διαδικασίας.

Q1 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q2 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q3 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q4 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q5 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q6 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

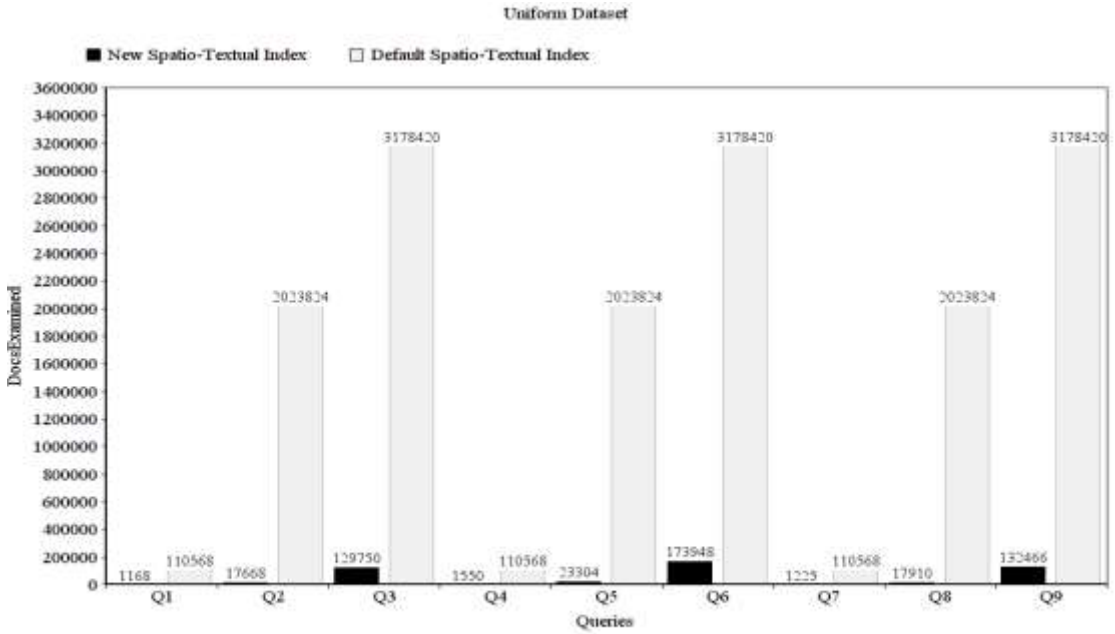
Q7 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q8 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q9 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Αποτελέσματα

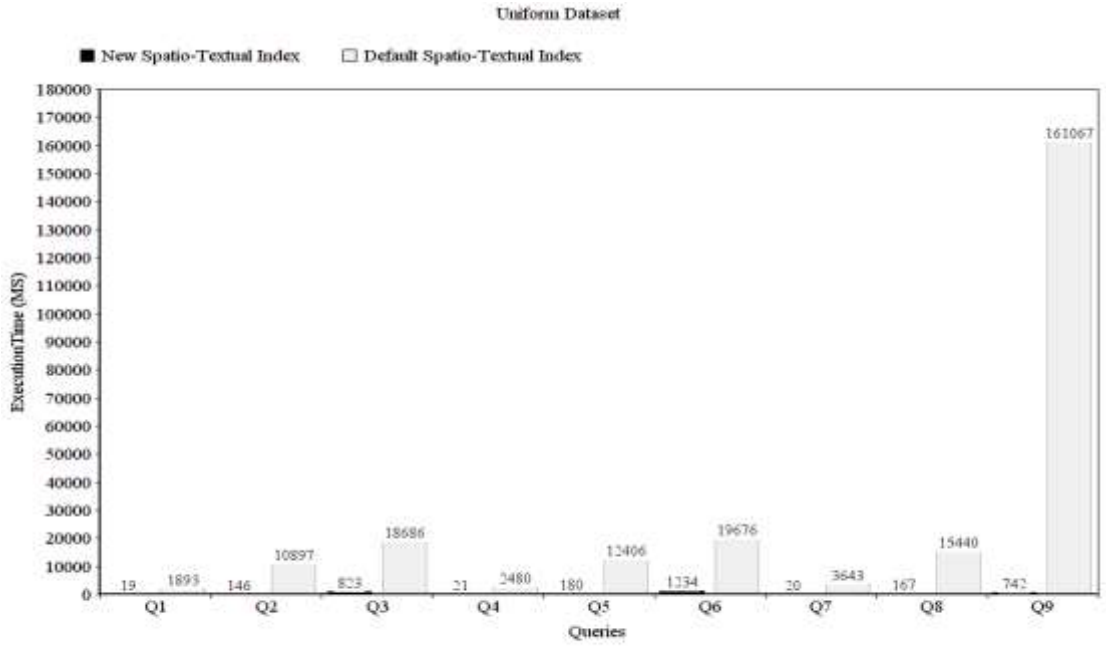
Το New Spatio-Textual index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε στο κεφάλαιο 4 και έχει εφαρμοστεί στο uniform_collection. Το Default Spatio-Textual Index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB και έχει εφαρμοστεί στο uniform_default_index collection.



Εικόνα 5.26 MongoDB κατανεμημένη υποδομή – σύγκριση εγγράφων που εξετάζονται στα δεδομένα με ομοιόμορφη κατανομή

Στο γράφημα τους εικόνας 5.26 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τα documents που εξετάστηκαν κατά την εκτέλεση των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index εμφανίζει με μεγάλη διαφορά καλύτερα αποτελέσματα από το Default Spatio-Textual index. Για το ερώτημα Q1 με το μικρό box και την λίστα μικρής συχνότητας, το New Spatio-Textual index εξέτασε 1168 documents και το Default Spatio-Textual index 110568 documents. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index εξέτασε 3178420 documents ενώ στα ίδια ερωτήματα το New Spatio-Textual index κυμάνθηκε από τα 129750 documents μέχρι 173948 documents. Η διαφορά είναι μεγάλη. Τα συγκεκριμένα ερωτήματα χρησιμοποιούσαν την λίστα λέξεων με τα περισσότερα documents. Αυτό σημαίνει ότι, η χωρο-κειμενική ευρετηρίαση που προσφέρει η MongoDB παρουσιάζει

χαμηλή απόδοση σε αναζητήσεις λέξεων μεγάλης συχνότητας. Η χωρο-κειμενική απόδοση που αναπτύχθηκε στο κεφάλαιο 4, εμφανίζει υψηλή απόδοση σε δεδομένα ομοιόμορφης κατανομής.



Εικόνα 5.27 MongoDB κατανεμημένη υποδομή – σύγκριση χρόνου εκτέλεσης ερωτημάτων στα δεδομένα με ομοιόμορφη κατανομή

Στο γράφημα τους εικόνας 5.27 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τον χρόνο εκτέλεσης των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index εμφανίζει και εδώ, με μεγάλη διαφορά καλύτερα αποτελέσματα από το Default Spatio-Textual index Για το ερώτημα Q1, το New Spatio-Textual index εκτελέστηκε σε 19 ms και το Default Spatio-Textual index εκτελέστηκε σε 1893 ms. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index συγκέντρωσε τους μεγαλύτερους χρόνους εκτελεσης που διακυμάνθηκαν από 18686 ms μέχρι 161067 ms ενώ το New Spatio-Textual index αντίστοιχα το 742 ms μέχρι 1234 ms. Το αξιοσημείωτο είναι ότι το New Spatio-Textual index παρουσιάζει υψηλές αποδόσεις τόσο σε ερωτήματα που περιέχουν μεγάλες γεωγραφικές περιοχές, όσο και σε ερωτήματα που περιέχουν συχνές λέξεις. Σύμφωνα και με τους χρόνους εκτέλεσης των ερωτημάτων, η χωρο-κειμενική ευρετηρίαση που προσφέρει η MongoDB επιβεβαιώνεται ότι παρουσιάζει χαμηλή απόδοση σε αναζητήσεις λέξεων μεγάλης συχνότητας. Σε αντίθεση, η χωρο-κειμενική απόδοση που αναπτύχθηκε στο κεφάλαιο 4, εμφανίζει υψηλή απόδοση σε δεδομένα ομοιόμορφης κατανομής.

Non-Uniform Data

Δημιουργούνται τα ακόλουθα 4 κουτιά και 3 λίστες. Παρατίθενται κατά αύξουσα σειρά ανά κατηγορία βάσει το μέγεθος τους.

- **Box1** [31.02734375, 55.97531083569679] [30.7284375, 55.968945343432936].
- **Box2** [70.12934375, 15.87831083569679] [70.02734375 , 15.87531083569679].
- **Box3** [50.22734375, 35.87531083569679] [50.0273437 , 35.8653108356967].
- **List1** {actus, accustom , accuser } (Σπάνιες λέξεις).
- **List2** {abatements , abdicated, abdicator } (Κανονική συχνότητα λέξεων).
- **List3** {abaft , abalone, abdicator} (Συχνές λέξεις).

Ο παρακάτω πίνακας περιέχει τον αριθμό των documents που περιέχουν τα κουτιά και οι λίστες ανά collection.

	non_uniform_collection	non_uniform_default_index
Box1	101	38
Box2	946	385
Box3	10189	4073
List1	8533	8529
List2	130856	130412
List3	963599	939180

Πίνακας 5.5 Αριθμός εγγραφών στα collections με την μη ομοιόμορφη κατανομή δεδομένων

Παρακάτω ακολουθεί οπτική απεικόνιση των κουτιών στο χάρτη.

Εικόνα 5.28 Non Uniform data box1

Εικόνα 5.29 Non uniform data box2

Εικόνα 5.30 Non uniform data box3

Παρακάτω δίνονται τα ερωτήματα της πειραματικής διαδικασίας.

Q1 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q2 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q3 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q4 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q5 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q6 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

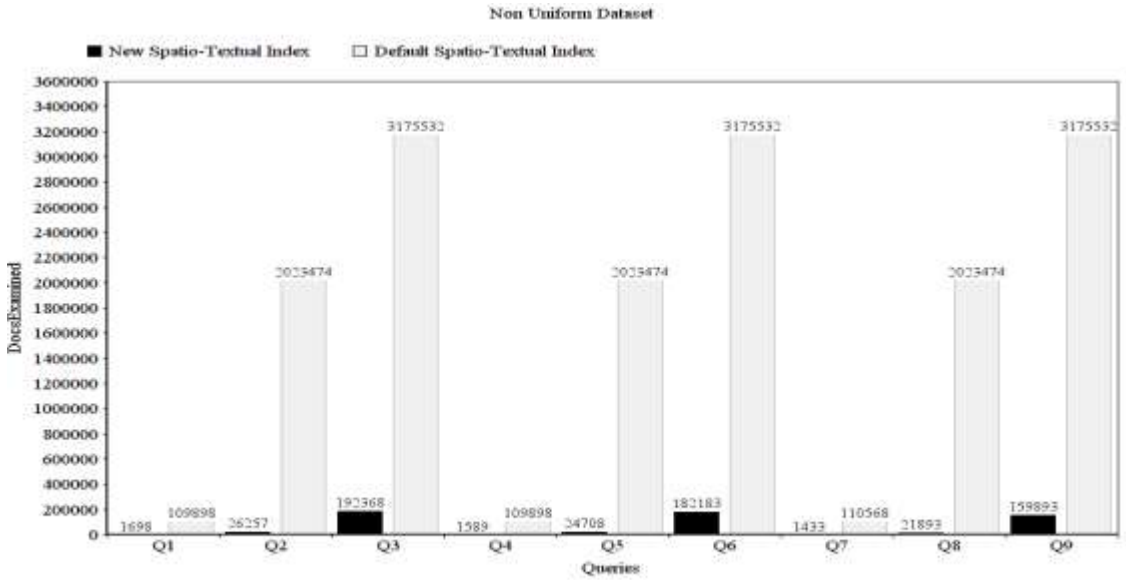
Q7 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q8 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q9 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

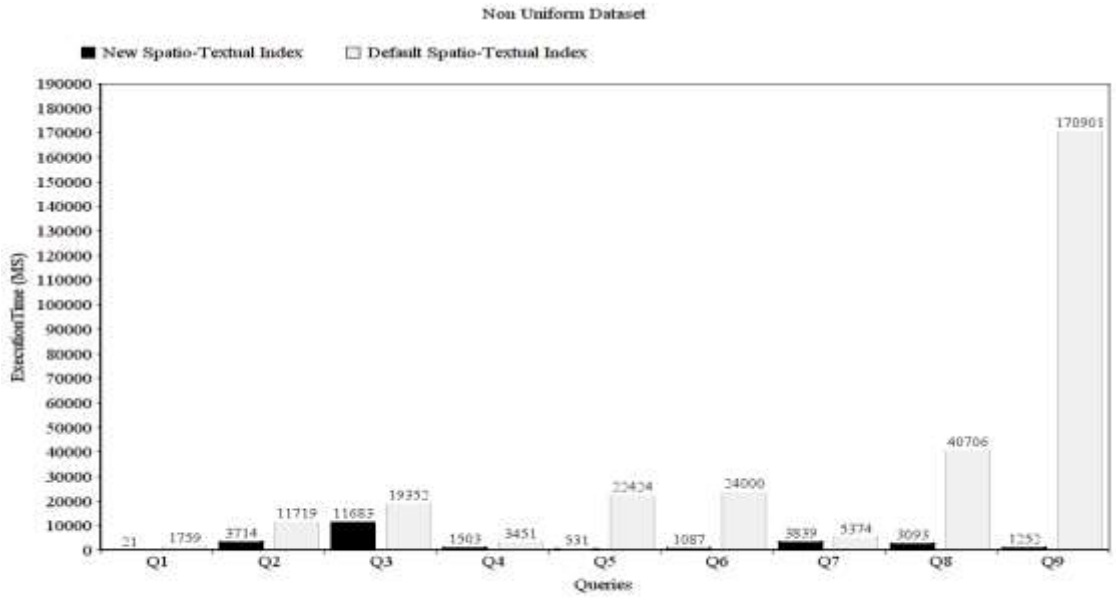
Αποτελέσματα

Το New Spatio-Textual index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε στο κεφάλαιο 4 και έχει εφαρμοστεί στο non_uniform_collection. Το Default Spatio-Textual Index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB και έχει εφαρμοστεί στο non_uniform_default_index collection.



Εικόνα 5.31 MongoDB κατανεμημένη υποδομή - σύγκριση εγγράφων που εξετάζονται στα δεδομένα μη ομοιόμορφης κατανομής

Στο γράφημα της εικόνας 5.31 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τα documents που εξετάστηκαν κατά την εκτέλεση των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index εμφανίζει με μεγάλη διαφορά, καλύτερα αποτελέσματα από το Default Spatio-Textual index. Για το ερώτημα Q1, το New Spatio-Textual index εξέτασε 1698 documents και το Default Spatio-Textual index 109898 documents. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index εξέτασε 3175532 documents ενώ στα ίδια ερωτήματα το New Spatio-Textual index κυμάνθηκε από τα 159893 documents μέχρι 192368 documents. Το New Spatio-Textual index και σε αυτήν την περίπτωση, εμφάνισε υψηλή απόδοση για την χωρο-κειμενική ευρετηρίαση. Ενώ η χωρο-κειμενική ευρετηρίαση της MongoDB παρουσίασε χαμηλή απόδοση σε ερωτήματα που περιείχαν συχνές λέξεις.



Εικόνα 5.32 MongoDB κατανεμημένη υποδομή - σύγκριση χρόνου εκτέλεσης ερωτημάτων στα δεδομένα μη ομοιόμορφης κατανομής

Στο γράφημα της εικόνας 5.32 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τον χρόνο εκτέλεσης των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index παρουσιάζει καλύτερα αποτελέσματα από το Default Spatio-Textual index. Για το ερώτημα Q1, το New Spatio-Textual index εκτελέστηκε σε 21 ms και το Default Spatio-Textual index εκτελέστηκε σε 1759 ms. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index συγκέντρωσε τους μεγαλύτερους χρόνους εκτελεσης που διακυμάνθηκαν από 19352 ms μέχρι 170901 ms ενώ το New Spatio-Textual index αντίστοιχα το 1252 ms μέχρι 11683 ms. Τα συγκεκριμένα ερωτήματα χρησιμοποιούσαν την λίστα λέξεων με τα περισσότερα documents. Συνεπώς και σε αυτήν την περίπτωση, η χωρο-κειμενική ευρετηρίαση που προσφέρει η MongoDB παρουσιάζει χαμηλή απόδοση σε αναζητήσεις λέξεων μεγάλης συχνότητας. Η χωρο-κειμενική απόδοση που αναπτύχθηκε στο κεφάλαιο 4, εμφανίζει υψηλή απόδοση και σε δεδομένα μη ομοιόμορφης κατανομής.

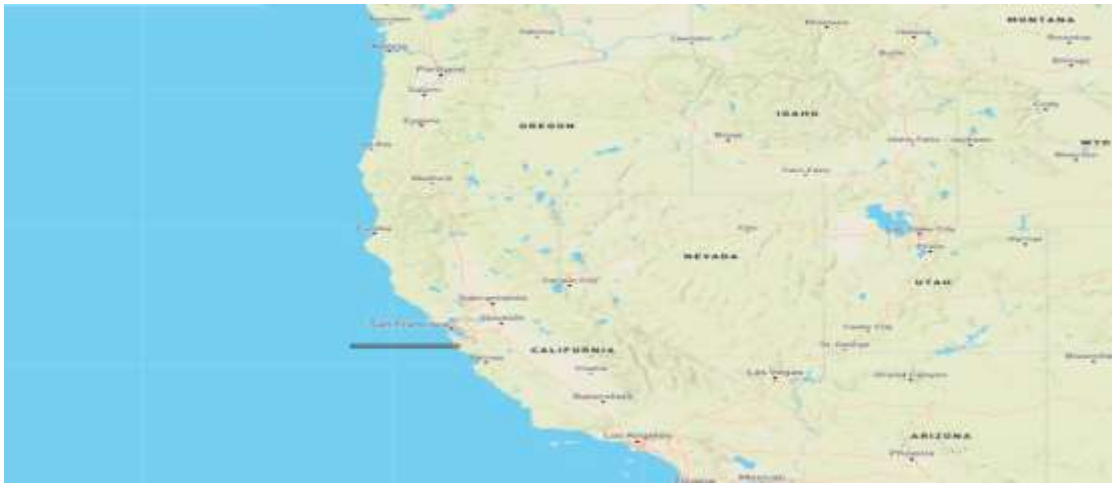
Real Data

Δημιουργούνται τα ακόλουθα 4 κουτιά και 3 λίστες. Παρατίθενται κατά αύξουσα σειρά ανά κατηγορία βάσει το μέγεθος τους.

- **Box1** [-121.640625, 33.87220829920497] [-118.125, 33.87041555094183]
- **Box2** [-124.67285156250001, 37.16031654673677] [-122.25585937500001 , 37.47530995561875]
- **Box3** [-75.00390625, 39.66827914916014] [-70.400390625 , -39.774769485295465]
- **List1** {steak, egg, plate} (Σπάνιες λέξεις) .
- **List2** {hashtags, tag, tweet } (Κανονική συχνότητα λέξεων).
- **List3** { thanks, happy, birthday} (Συχνές λέξεις).

Παρακάτω ακολουθεί οπτική απεικόνιση των κουτιών στο χάρτη.

Εικόνα 5.33 Real data box1



Εικόνα 5.34 Real data box2

Εικόνα 5.35 Real data box3

Ο παρακάτω πίνακας περιέχει τον αριθμό των documents που περιέχουν τα κουτιά και οι λίστες ανά collection.

	real_data_collection	**real_data_default_index**
Box1	99	15
Box2	1037	186
Box3	9564	2217
List1	4965	4831
List2	28938	4777
List3	215047	161578

Πίνακας 5.6 Αριθμός εγγραφών στα collections με τα πραγματικά δεδομένα

Παρακάτω δίνονται τα ερωτήματα της πειραματικής διαδικασίας.

Q1 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q2 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q3 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q4 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q5 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q6 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

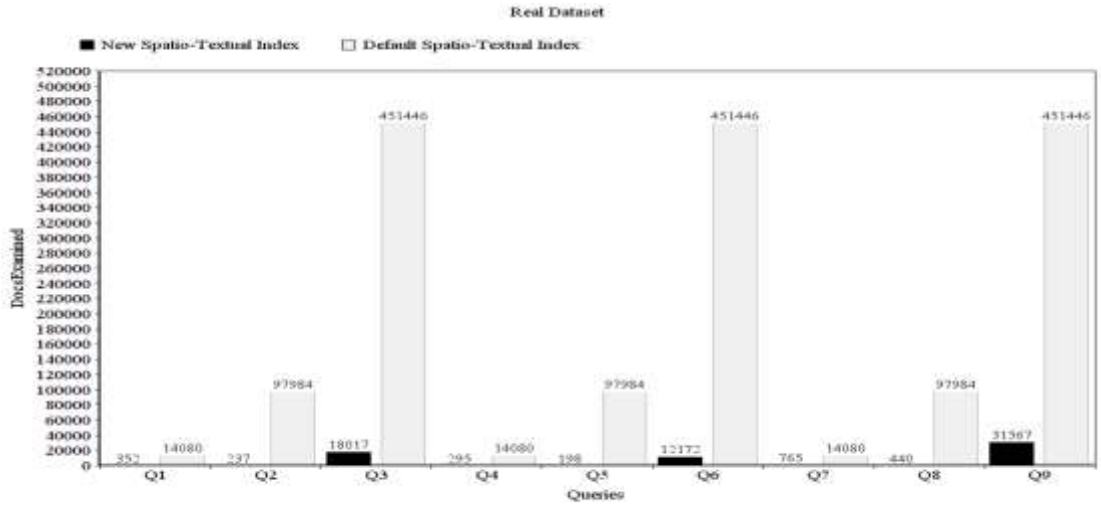
Q7 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q8 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q9 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

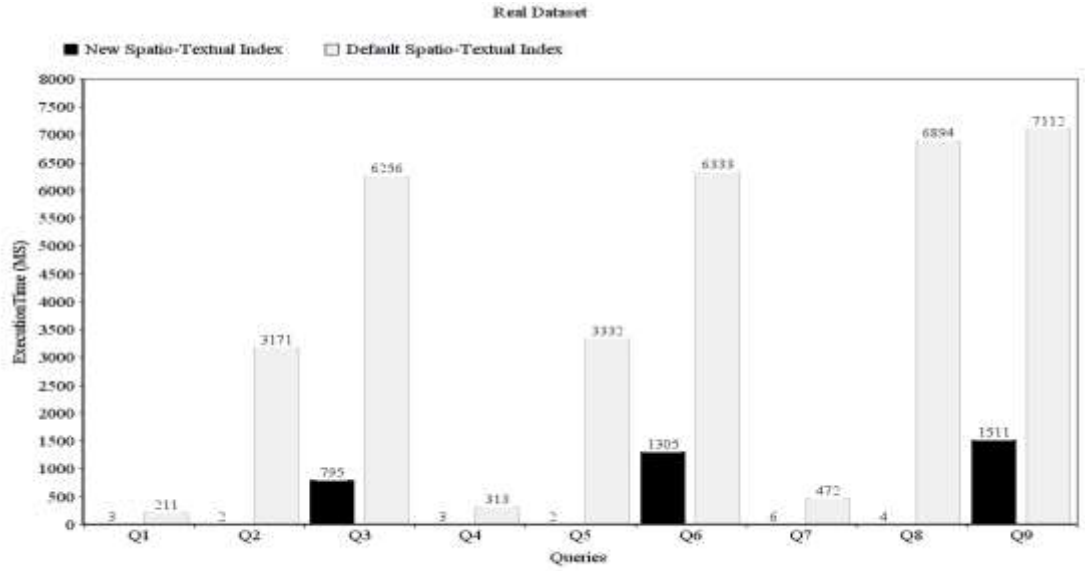
Αποτελέσματα

Το New Spatio-Textual index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε στο κεφάλαιο 4 και έχει εφαρμοστεί στο real_data_collection. Το Default Spatio-Textual Index αντικατοπτρίζει την χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB και έχει εφαρμοστεί στο real_data_default_index collection.



Εικόνα 5.36 MongoDB κατανεμημένη υποδομή - σύγκριση εγγράφων που εξετάζονται στα πραγματικά δεδομένα

Στο γράφημα της εικόνας 5.36 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τα documents που εξετάστηκαν κατά την εκτέλεση των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index και στα πραγματικά δεδομένα, παρουσιάζει εμφανώς καλύτερα αποτελέσματα από το Default Spatio-Textual index. Για το ερώτημα Q1, το New Spatio-Textual index εξέτασε 352 documents και το Default Spatio-Textual index 14080 documents. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index εξέτασε 451446 documents ενώ στα ίδια ερωτήματα το New Spatio-Textual index κυμάνθηκε από τα 12172 documents μέχρι 32567 documents. Τα συγκεκριμένα ερωτήματα χρησιμοποιούσαν την λίστα με τις συχνές λέξεις. Αυτό σημαίνει ότι, χωρο-κειμενική ευρετηρίαση που διαθέτει η MongoDB εμφανίζει χαμηλή απόδοση όταν αναζητούνται λέξεις μεγάλη συχνότητα. Αυτό ισχύει και σε κατανεμημένο περιβάλλον MongoDB υποδομής σε πραγματικά δεδομένα. Το New Spatio-Textual index και σε αυτήν την περίπτωση, εμφάνισε υψηλή απόδοση για την χωρο-κειμενική ευρετηρίαση.



Εικόνα 5.37 MongoDB κατανεμημένη υποδομή - σύγκριση χρόνου εκτέλεσης ερωτημάτων σε πραγματικά δεδομένα

Στο γράφημα της εικόνας 5.37 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τον χρόνο εκτέλεσης των ερωτημάτων. Σε όλα τα ερωτήματα το New Spatio-Textual index εμφανίζει καλύτερα αποτελέσματα από το Default Spatio-Textual index. Για το ερώτημα Q1, το New Spatio-Textual index εκτελέστηκε σε 3 ms και το Default Spatio-Textual index εκτελέστηκε σε 211 ms. Για τα Q3, Q6 και Q9 το Default Spatio-Textual index συγκέντρωσε τους μεγαλύτερους χρόνους εκτελεσης που διακυμάνθηκαν από 6256 ms μέχρι 7112 ms ενώ το New Spatio-Textual index αντίστοιχα το 795 ms μέχρι 1511 ms.

5.1.3 HBase σε κεντρικοποιημένο περιβάλλον

Για την αποτύπωση των ερωτημάτων στην Hbase, χρησιμοποιούνται Filters και FilterLists. Μέσω των Filters εκφράζονται τα ερωτήματα. Τα FilterLists μπορούν να συνδυάσουν πολλά Filters. Τα ερωτήματα εκτελέστηκαν μέσω της εντολής scan.

Ο παρακάτω πίνακας περιέχει τα τεχνικά χαρακτηρίστηκα της υπολογιστικής μονάδας που έτρεξε η πειραματική διαδικασία.

CPU	**RAM**	**HARD DISK**
4 cores (Intel Core i7)	16 GB	256 GB SSD

Πίνακας 5.7 Πόροι - HBase

Τα δεδομένα της πειραματικής διαδικασίας προήλθαν από το dataset restaurants-ver1. Το συγκεκριμένο Dataset περιείχε 78981 εγγραφές. Ακολουθήθηκαν 2 προσεγγίσεις για την εισαγωγή των δεδομένων σε πίνακες.

Σχεδιαστικά το data model της πρώτης προσέγγισης έχει την ακόλουθη δομή:

(Rowkey, data:groupid, data:lon, data:lat, data:keyword).

Το rowkey παίρνει τιμή από το hilbertvalue με τον τρόπο που αναπτύχθηκε στο Κεφάλαιο 4, σε συνδυασμό με ένα αύξοντα αριθμό. Ακολουθούν τα column families hilbertindex, groupid lon, lat και keyword. Η μεταβλητή hilbertindex περιέχει μια τιμή που προέρχεται από την γεωγραφική συντεταγμένη lon,lat που έχει κωδικοποιηθεί από την Hilbert curve μέθοδο, συνενώνοντας το keyword .Η μεταβλητή groupid αντιστοιχεί τις εγγραφές του αρχικού dataset με αυτές στην HBase. Κάθε αντικείμενο περιέχει τις παρακάτω 5 εγγραφές.

- (rowkey, groupid).
- (rowkey, hilbertindex).
- (rowkey, lon).
- (rowkey, lat).
- (rowkey, keyword).

Παράδειγμα, έστω ότι μια εγγραφή από το αρχικό dataset έχει την ακόλουθη μορφή.

lon=-118.937563, lat=34.19198813 και keywords = { Greek, American }

Στην HBase η καταχώρηση θα γίνει με τον παρακάτω τρόπο.

- **HilbertValue /1**, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- **HilbertValue/1**, column=data:hilbertindex, timestamp=**DefaultTimestampValue**, value= **HilbertValueGreek.**
- **HilbertValue/1**, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.
- **HilbertValue/1**, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.
- **HilbertValue/1**, column=data:keyword, timestamp=**DefaultTimestampValue**, value=Greek.
- **HilbertValue/2**, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- **HilbertValue/2**, column=data:hilbertindex, timestamp=**DefaultTimestampValue**, value= **HilbertValueAmerican**
- **HilbertValue/2**, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.

- **HilbertValue/2**, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.
- **HilbertValue/2**, column=data:keyword, timestamp=**DefaultTimestampValue**, value= American.

Με αυτόν τον τρόπο πραγματοποιείται χωρο-κειμενική ευρετηρίαση σε ένα dataset που χρησιμοποιεί Column oriented data model.

Σχεδιαστικά το data model της δεύτερης προσέγγισης έχει την ακόλουθη δομή:

(Rowkey, data:groupid, data:lon, data:lat, data:keyword).

Το rowkey παίρνει τιμή από το keyword σε συνδυασμό με έναν αύξοντα αριθμό. Ακολουθούν τα column families groupid lon, lat και keyword. Η μεταβλητή groupid αντιστοιχεί τις εγγραφές του αρχικού dataset με αυτές στην HBase. Κάθε αντικείμενο περιέχει τις παρακάτω 4 εγγραφές.

- (rowkey, groupid).
- (rowkey, lon).
- (rowkey, lat).
- (rowkey, keyword).

Παράδειγμα, έστω ότι μια εγγραφή από το αρχικό dataset έχει την ακόλουθη μορφή.

lon=-118.937563, lat=34.19198813 και keywords = { Greek, American }

Στην HBase η καταχώρηση θα γίνει με τον παρακάτω τρόπο.

- Greek /1, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- Greek /1, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.
- Greek /1, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.
- Greek /1, column=data:keyword, timestamp=**DefaultTimestampValue**, value=Greek.
- American /2, column=data:groupid, timestamp=**DefaultTimestampValue**, value=1.
- American/2, column=data:lon, timestamp=**DefaultTimestampValue**, value=-118.937563.
- American/2, column=data:lat, timestamp=**DefaultTimestampValue**, value=34.19198813.

- American/2, column=data:keyword, timestamp=**DefaultTimestampValue**, value= American.

Σε αυτήν την περίπτωση δεν χρησιμοποιείται κάποιο ευρετήριο.

Στην συνέχεια δημιουργήθηκαν 2 πίνακες στην HBase ο geodataindex και ο geodata. Στον πίνακα geodataindex είχε την δομή της πρώτης προσέγγισης και ο geodata είχε την δομή της δεύτερης προσέγγισης. Οι πίνακες περιείχαν 239694 objects.

Σε συνέχεια της πειραματικής διαδικασίας δημιουργήθηκαν τα ακόλουθα 3 κουτιά και 3 λίστες. Παρατίθενται κατά αύξουσα σειρά ανά κατηγορία βάσει το μέγεθος τους.

- **Box1** [-30.937563, 60.19198813] [-70.6995, 20.974964] (Μικρό Box).
- **Box2** [-82.937563, 40.19198813] [-50.6995, 20.974964] (Μεσαίο Box).
- **Box3** [-98.937563, 60.191988133] [-50.6995, 20.974964] (Μεγάλο Box).
- **List1** {Cambodian, Lebanese, Persian} (Σπάνιες λέξεις).
- **List2** {Greek, Mediterranean, Grill } (Κανονική συχνότητα λέξεων).
- **List3** {Burgers, Pizza , Sandwiches } (Συχνές λέξεις).

Ο παρακάτω πίνακας περιέχει τον αριθμό των αντικειμένων που περιέχουν τα κουτιά και οι λίστες ανά πίνακα.

	geodataindex	geodata
Box1	820	820
Box2	44327	44327
Box3	169148	169148
List1	185	185
List2	4295	4295
List3	46254	46254

Πίνακας 5.8 Κουτιά και λίστες ερωτημάτων

Παρακάτω δίνονται τα ερωτήματα της πειραματικής διαδικασίας.

Q1 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q2 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q3 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q4 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

Q5 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q6 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box2 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Q7 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List1.

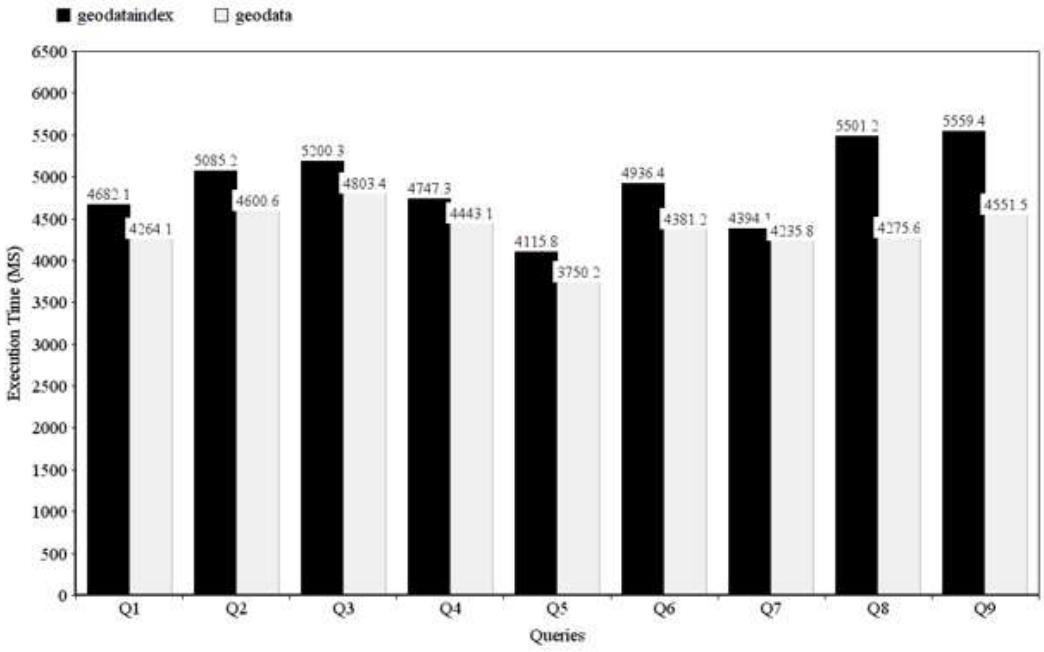
Q8 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box3 και περιέχουν μια ή περισσότερες από τις λέξεις της List2.

Q9 - να βρεθούν όλα τα χωρο-κειμενικό αντικείμενος εντός του Box1 και περιέχουν μια ή περισσότερες από τις λέξεις της List3.

Για την μεθοδολογία Hilbert curve δόθηκαν οι διαστάσεις για το x -180 μέχρι 180 και για το y -90 έως 90. Χρησιμοποιήθηκε αναπαράσταση με 10 bits.

Αποτελέσματα

Η μεθοδολογία που αναπτύχθηκε στο κεφάλαιο 4 έχει εφαρμοστεί στον πίνακα geodataindex και η στον πίνακα geodata έχει εφαρμοστεί η δεύτερη προσέγγιση χωρίς την χρήση κάποιας ευρετηρίασης.



Εικόνα 5.38 HBase- σύγκριση χρόνου εκτέλεσης ερωτημάτων

Στο γράφημα της εικόνας 5.38 συγκρίνονται τα ερωτήματα Q1-Q9, σχετικά με τον χρόνο εκτέλεσης των ερωτημάτων. Οι χρόνοι εκτέλεσης των ερωτημάτων για τις 2

προσεγγίσεις παρουσιάζουν παρόμοια απόδοση. Ωστόσο, η τεχνική χωρίς την χρήση ευρετηρίασης εμφανίζει ελαφρώς καλύτερους χρόνους εκτέλεσης για όλα τα ερωτήματα. Χαρακτηριστικά για το ερώτημα Q1 που έχει μικρό το μικρό box και την λίστα λέξεων με μικρή συχνότητα. Στον πίνακα geodataindex το ερώτημα εκτελείται στα 46821 ms και αντίστοιχα στο geoindex σε 42641 ms. Στο ερώτημα Q3 με το μικρό box και την λίστα με τις συχνές λέξεις η κατάσταση παραμένει ίδια. Στον πίνακα geodataindex το ερώτημα εκτελείται στα 52003 ms και αντίστοιχα στο geoindex σε 48034 ms. Η μεγαλύτερη διαφορά παρουσιάζεται στο ερώτημα Q8 με το μεγάλο box και την λίστα λέξεων κανονικής συχνότητας. Στον πίνακα geodataindex το ερώτημα εκτελείται στα 55012 ms και αντίστοιχα στο geoindex σε 42756 ms. Παρόμοια εικόνα είναι και στο Q9. Αυτό σημαίνει ότι όσο μεγαλώνει γεωγραφική περιοχή αναζήτησης και περιέχει περισσότερα αντικείμενα τόσο μειώνεται η απόδοση της χωρο-κειμενικής ευρετηρίασης.

6. Συμπεράσματα και μελλοντική μελέτη

Η χωρο-κειμενική ευρετηρίαση που πραγματεύεται αυτή η έρευνα. Εφαρμόστηκε σε 2 NoSQL stores MongoDB και HBase με στόχο να αυξήσει την απόδοση των χωρο-κειμενικών ερωτημάτων που εκτελούνται σε αυτά τα stores. Συγκεκριμένα τα ερωτήματα που εκτελέστηκαν ήταν τύπου boolean range queries. Αρχικά, δοκιμάστηκε σε ένα κεντρικοποιημένο περιβάλλον MongoDB συγκινώντας την με ήδη υπάρχουσες χωρικές και χωρο-κειμενικές ευρετηριάσεις που διαθέτει η MongoDB χρησιμοποιώντας μια ποικιλία ερωτημάτων που περιείχαν γεωγραφικές περιοχές με λίγα, κανονικά και πολλά χωρο-κειμενικά αντικείμενα σε συνδυασμό με λίστες λέξεων με μικρή, κανονική και μεγάλη συχνότητα. Τα αποτελέσματα ήταν εντυπωσιακά. Η χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε εμφάνισε υψηλή απόδοση σε σχέση με τις ήδη υπάρχουσες χωρικές και χωρο-κειμενικές ευρετηριάσεις που προσφέρει ήδη η MongoDB, ανεξάρτητα με το πλήθος των αντικειμένων που περιείχαν οι γεωγραφικές περιοχές στα ερωτήματα και ανεξάρτητα από την συχνότητα των λέξεων προς αναζήτηση. Για την επιβεβαίωση της υψηλής απόδοσης που εμφάνισε η χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε. Εφαρμόστηκε και σε MongoDB κατανεμημένο περιβάλλον χρησιμοποιώντας συνθετικά δεδομένα ομοιόμορφης κατανομής και μη ομοιόμορφης κατανομής καθώς και πραγματικά δεδομένα που προήλθαν από την πλατφόρμα Twitter. Η κατάσταση παρέμεινε ίδια. Εξίσου, στα συνθετικά δεδομένα ομοιόμορφης κατανομής και μη ομοιόμορφης κατανομής και στα πραγματικά δεδομένα. Τα αποτελέσματα ήταν εντυπωσιακά. Η χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε εμφάνισε υψηλή απόδοση και πάλι, τόσο στα αντικείμενα που εξετάστηκαν για την εκτέλεση του κάθε ερωτήματος, όσο και στους χρόνους εκτέλεσης των ερωτημάτων ανεξάρτητα από την κατανομή που ακολουθούσαν τα δεδομένα, σε σύγκριση με την χωρο-κειμενική ευρετηρίαση που ήδη προσφέρει η MongoDB. Η οποία παρουσίασε χαμηλή απόδοση σε ερωτήματα που περιέχουν συχνές λέξεις αναζήτησης. Σε καμία περίπτωση όμως δεν ήταν ανταγωνιστική σε κανένα ερώτημα σε σχέση με την χωρο-κειμενική ευρετηρίαση που αναπτύχθηκε. Για την αξιολόγηση της νέας χωρο-κειμενικής ευρετηρίασης και σε άλλο NoSQL store, χρησιμοποιήθηκε η HBase. Η Hbase όμως δεν υποστηρίζει απευθείας χωρική ή χωρο-κειμενική ευρετηρίαση σε αντίθεση με την MongoDB. Έτσι, σχεδιάστηκε ένα σχήμα δεδομένων βάσει της μορφής των δεδομένων που υποστηρίζει η HBase, στο οποίο δεν εφαρμόζεται κάποια χωρική ή χωρο-κειμενική ευρετηρίαση και θα μπορεί να συγκριθεί με την νέα χωρο-κειμενική ευρετηρίαση. Τα πειράματα εκτελέστηκαν μόνο σε κεντρικοποιημένο περιβάλλον. Η απόδοση των δυο τεχνικών εμφάνισαν παρόμοια αποτελέσματα. Ωστόσο, στην τεχνική που δεν χρησιμοποιήθηκε ευρετηρίαση εμφάνισε ελαφρώς καλύτερη απόδοση.

Ως συνέχεια της έρευνας, θα μπορούσε να εφαρμοστεί η νέα χωρο-κειμενική ευρετηρίαση σε περισσότερα NoSQL stores με διαφορετικό μοντέλο δεδομένων από αυτό που έχει εφαρμοστεί μέχρι στιγμής, ώστε να αξιολογηθεί η απόδοση της συνολικά στα NoSQL stores. Επίσης, σαν επέκταση της έρευνας, θα μπορούσε να εξεταστεί η απόδοση της νέας χωρο-κειμενικής ευρετηρίασης και σε επιπλέον τύπους χωρο-κειμενικών ερωτημάτων όπως boolean knn queries και top-k queries. Συγκεκριμένα, επειδή τα Boolean KNN Queries αναζητούν τον πλησιέστερο γείτονα ενός αντικειμένου σε συγκεκριμένη ακτίνα και εκτελούνται διαφορετικοί υπολογισμοί σε σχέση με τα Boolean Range Queries, είναι ζητούμενο να ελεγχθεί πως ανταποκρίνεται η νέα χωρο-κειμενική ευρετηρίαση. Το ίδιο ισχύει και για τα top-k queries, διότι στους υπολογισμούς, εξετάζεται και η κειμενική ομοιότητα των αντικειμένων πράγμα που δεν γίνεται στα boolean range queries.

7. Βιβλιογραφία

[1] Nikolaos Koutroumanis, Panagiotis Nikitopoulos, Akrivi Vlachou, Christos Doulkeridis. NoDA: Unified NoSQL Data Access Operators for Mobility Data. SSTD '19: Proceedings of the 16th International Symposium on Spatial and Temporal Databases, Pages 174–177, 2019. DOI: https://doi.org/10.1145/3340964.3340981

[2] Ahmed R.Mahmood, Sri Punni, Walid G.Aref. Spatio-temporal access methods: a survey (2010 - 2017). Geoinformatica, Pages 1-36, 2019. DOI: https://doi.org/10.1007/s10707-018-0329-2.

[3] Amit Singh, Hakan Ferhatosmanoglu, Ali Şaman Tosun. High dimensional reverse nearest neighbor queries. CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management, Pages 91–98. DOI: https://doi.org/10.1145/956863.956882

[4] Amélie Marian, Nicolas Bruno, Luis Gravano. Evaluating top-k queries over web-accessible databases. ACM Transactions on Database Systems, Pages:319-362 DOI: https://doi.org/10.1145/1005566.1005569

[5] Suprio Ray, Bogdan Simion, Angela Demke Brown, Ryan Johnson. Skew-Resistant Parallel In-memory Spatial Join. SSDBM '14: Proceedings of the 26th International Conference on Scientific and Statistical Database Management. Pages 1–12. DOI: https://doi.org/10.1145/2618243.2618262

[6] L. Chen, G. Cong, C. S. Jensen, and D. Wu. Spatial keyword query processing: An experimental evaluation. PVLDB, Pages:217–228, 2013. DOI: https://doi.org/10.14778/2535569.2448955.

[7] J. Jiang, H. Lu, B. Yang, and B. Cui. Finding top-k local users in geo-tagged social media data. 31st International Conference on Data Engineering, Pages 267–278, 2015. DOI: 10.1109/ICDE.2015.7113290.

[8] Y.Ma, Y.Zhang, and X.Meng. St-hbase: A scalable data management system for massive geo-tagged objects Proceedings of the 14th international conference on Web-Age Information Management. Pages 155–166, 2013. DOI: https://doi.org/10.1007/978-3-642-38562-9_16.

[9] Rick Cattell. Scalable SQL and NoSQL data stores. ACM SIGMOD Record, Pages:1–26, 2011. DOI: https://doi.org/10.1145/1978915.1978919.

[10] Cornelia Gyrödi, Robert Gyrödi, George Pecherle, Andrada Olah. A Comparative Study: MongoDB vs. MySQL. 13th International Conference on Engineering of Modern Electric Systems (EMES). DOI: 10.1109/EMES.2015.7158433

[11] Chang, F., Dean J., Ghemawat S., Hsieh W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A, and Gruber, R. E. Bigtable: A distributed storage system for

structured data. ACM Transactions on Computer Systems, 2008, Pages:1–26, DOI: https://doi.org/10.1145/1365815.1365816.