ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

"Συγκριτική μελέτη των κατανεμημένων συστημάτων διαχείρισης κειμένων MongoDB, Elastic Search και Lucene"

Δημήτρης Γκέκας,

Γιώργος Σαμαράς

Επιβλέπων : Ταμπακάς Βασίλειος

Πάτρα , Απρίλιος 2022

Περίληψη

Η παρούσα εργασία ασχολείται με τις βάσεις δεδομένων προσανατολισμένες σε έγγραφα και συγκεκριμένα με τις Elasticsearch MongoDB και Lucene.

Ουσιαστικά, αποτελεί αφενός μια ανάλυση των δυνατοτήτων και των χαρακτηριστικών τους αλλά αφετέρου και μια συγκριτική μελέτη αυτών.

Η ανάλυση γίνεται από την κάλυψη της αρχιτεκτονικής, των υπηρεσιών, των δυνατοτήτων και των αδύναμων σημείων της κάθε μιας τεχνολογίας. Οι συγκρίσεις προκύπτουν μέσα από την εξέταση της συμπεριφοράς, της αρχιτεκτονικής και των δυνατοτήτων των τριών τεχνολογιών σε συγκεκριμένες συνιστώσες.

Τέλος, εξάγονται συμπεράσματα σχετικά με την επιλογή της κατάλληλης τεχνολογίας, ανάλογα με την εφαρμογή και προκύπτουν ιδέες για περαιτέρω μελλοντική μελέτη.

Περιγραφή του Προβλήματος

Τα τελευταία χρόνια παρατηρείται μια συνεχής αύξηση των εγκατεστημένων εφαρμογών λογισμικού και των ρυθμών ανταλλαγής δεδομένων μέσω διαδικτύου.

Η αγορά του λογισμικού παρουσιάζει πρωτοφανή ανάπτυξη και η ανάγκη για συστήματα ικανά να διαχειρίζονται μεγάλο όγκο δεδομένων έχει γίνει επιτακτική, σχεδόν σε κάθε κλάδο της βιομηχανίας.

Η τεχνολογική εξέλιξη, όσων αφορά στο υλικό των υπολογιστών, έχει μειώσει ρυθμούς σε σχέση με τα προηγούμενα έτη και το βάρος φαίνεται πως πέφτει στην ανάπτυξη πιο αποδοτικού λογισμικού και πιο αποτελεσματικών τεχνολογιών για την καλύτερη αξιοποίηση των πόρων που υπάρχουν.

Η δε αύξηση των διαδικτυακών εφαρμογών καθιστά διαθέσιμους τεράστιους όγκους δεδομένων, κάθε είδους και η σωστή αξιοποίηση αυτών μπορεί να δώσει πολύτιμες καινοτομίες και λύσεις, τόσο σε βιομηχανικό, όσο και σε ερευνητικό επίπεδο.

Η ανάγκη για μεταφορά δεδομένων μέσω διαδικτύου και η διάδοση των διαδικτυακών εφαρμογών οδήγησε στην δημιουργία και την ευρεία χρήση της τεχνολογίας Μεταφοράς αναπαραστατικής κατάστασης (Rest API).

Αυτή η τεχνολογία επιτρέπει την μεταφορά δεδομένων με την μορφή αιτήματος και απάντησης, όπου κάθε διαδικτυακή εφαρμογή είναι ικανή να παράγει και να δέχεται αιτήματα προς και από άλλες εφαρμογές, δημιουργώντας έτσι μια διασύνδεση μέσω διαδικτύου.

Ολοένα και περισσότερες γίνονται οι εφαρμογές που χρησιμοποιούν αυτήν την τεχνολογία, οδηγώντας αφενός στην ανάγκη για ασφαλή, γρήγορη και αξιόπιστη μεταφορά δεδομένων και αφετέρου στην ανάπτυξη αποδοτικών τεχνολογιών για την διαχείριση, την ανάκτηση και την αποθήκευση αυτών.

Η μεγαλύτερη πρόσβαση σε πληροφορία οδήγησε με αυτών τον τρόπο σε αυξημένες ανάγκες διαχείρισης μεγάλου όγκου δεδομένων. [1]

Επιπρόσθετα, το Διαδίκτυο των Πραγμάτων (Internet of Things) είναι μια από τις μεγαλύτερες τάσεις της επιστήμης των ηλεκτρονικών υπολογιστών στις μέρες μας. Οι εφαρμογές του διαδικτύου των πραγμάτων αποτελούν, ουσιαστικά, συστοιχίες πολλών συσκευών, συνδεδεμένων στο διαδίκτυο ώστε να ανταλλάζουν πληροφορία και να μπορούν να διαχειριστούν κεντρικά. Οι εγκατεστημένες εφαρμογές αυτού του είδους αυξάνονται ραγδαία, ενώ είναι άρρηκτα συνδεδεμένες με την συλλογή και την μεταφορά δεδομένων. Η έκρηξη του όγκου των δεδομένων, η πολυπλοκότητά τους και η σημασία που έχουν αυτά για πολλές εφαρμογές έχει οδηγήσει στην ανάπτυξη της επιστήμης των Big D a t a ^ Big Data είναι συλλογές από μεγάλου όγκου και πολυπλοκότητας δεδομένα, που συλλέγονται με μεγάλες ταχύτητες από διάφορες συσκευές και εφαρμογές (πχ αισθητήρες, καταγραφείς συμπεριφορών χρηστών κτλ). Στις μέρες μας, τα Big Data χρησιμοποιούνται για πληθώρα εφαρμογών, ικανών να βελτιώσουν το επίπεδο ζωής των ανθρώπων, να βοηθήσουν σε πολύπλοκες αποφάσεις και άλλα. [2]

Όλα τα παραπάνω συνηγορούν στην ανάγκη για τεχνολογίες ικανές να αποθηκεύσουν, ανακτήσουν και να διαχειριστούν πολύπλοκα και μεγάλου όγκου δεδομένα.

Αυτές οι τεχνολογίες θα πρέπει να λειτουργούν αποδοτικά, γρήγορα και για μεγάλη ποικιλία δεδομένων και εφαρμογών.

Πρόκειται για Βάσεις Δεδομένων που θα μπορούν να λειτουργήσουν με καινοτόμους τρόπους και με πλεονεκτήματα που δεν υπάρχουν σε παλαιότερες τεχνολογίες.

Αναπόφευκτα, θα πρέπει να διαφέρουν από τις παραδοσιακές βάσεις δεδομένων σε πολλά χαρακτηριστικά τους και να εστιάζουν σε συγκεκριμένες εφαρμογές.

Αυτές οι τεχνολογίες έχουν κεντρίσει το ενδιαφέρον πολλών ερευνών, ενώ ήδη έχουν εισχωρήσει και στην βιομηχανία.

Ευχαριστίες

Θα θέλαμε να ευχαριστήσουμε τον επιβλέποντα καθηγητή μας Ταμπακά Βασίλειο, για την καθοδήγηση που μας προσέφερε και το χρόνο που διέθεσε δίνοντάς μας χρήσιμες συμβουλές και οδηγίες για την ολοκλήρωση της πτυχιακής μας εργασίας.

Στο ίδιο πλαίσιο ευγνωμοσύνης, θα θέλαμε να ευχαριστήσουμε όλους τους καθηγητές του Τμήματος Μήχανικων Πληροφοριής για τη συμβολή τους στην επιστημονική και τεχνολογική μας συγκρότηση στα χρόνια της φοίτησής μου στο Τμήμα.

ΠΕΡΙΕΧΟΜΕΝΑ

Περιεχόμενα Σχημάτων

Περιεχόμενα Πινάκων

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1. Σκοπός και Συνεισφορά της Εργασίας

Η παρούσα εργασία, αποτελεί, αφενός, μια θεωρητική ανάλυση των νέων τεχνολογιών αποθήκευσης Elastic Search, MongoDB και Luciene και αφετέρου μια συγκριτική μελέτη αυτών μέσα από πραγματικές εφαρμογές τους.

Στα πλαίσια της εργασίας αυτής θα καταγραφεί το θεωρητικό υπόβαθρο των τεχνολογιών[1] αυτών, οι δυνατότητές τους και η διαφορές τους από τις παραδοσιακές σχεσιακές βάσεις δεδομένων. Ακόμη, θα παρατεθούν οι περιορισμοί αυτών, όπως και οι μελλοντικές επεκτάσεις που μπορούν να εισαχθούν για την περεταίρω αξιοποίησή τους. Επιπλέον, θα παρουσιαστούν πρακτικές εφαρμογές της καθεμιάς τεχνολογίας. Αυτό θα βοηθήσει στην αξιολόγηση των δυνατοτήτων των τεχνολογιών αυτών, στην εξοικείωση και στην ποσοτικοποίηση των χαρακτηριστικών τους.

1.2 Διάρθρωση της Αναφοράς

Η διάρθρωση της παρούσας εργασίας είναι η εξής:

- Κεφάλαιο 2: Παρουσιάζεται το θεωρητικό υπόβαθρο της ερευνητικής περιοχής και αναλύονται σχεσιακές και μη σχεσιακές βάσεις δεδομένων.

- Κεφάλαιο 3: Παρουσιάζονται και αναλύονται οι δυνατότητες, οι εφαρμογές και τα βασικά χαρακτηριστικά της Elasticsearch.

- Κεφάλαιο 4: Παρουσιάζονται και αναλύονται οι δυνατότητες, οι εφαρμογές και τα βασικά χαρακτηριστικά της MongoDB.

- Κεφάλαιο 5: Παρουσιάζονται και αναλύονται οι δυνατότητες, οι εφαρμογές και τα βασικά χαρακτηριστικά της Lucene

- Κεφάλαιο 6: Παρουσιάζεται πρακτικό κομμάτι για την ανάλυση τον MongoDB, Elasticsearch και Lucene.

- Κεφάλαιο 7: Παρουσιάζονται συγκρίσεις των τριών παραπάνω εργαλείων με βάση διάφορες συνιστώσες.

- Κεφάλαιο 8: Παρουσιάζονται τα τελικά συμπεράσματα, τα προβλήματα που προέκυψαν και προτείνονται θέματα για μελλοντική μελέτη, αλλαγές και επεκτάσεις.

ΚΕΦΑΛΑΙΟ 2

Θεωρητικό Υπόβαθρο

2.1 Σχεσιακές βάσεις δεδομένων

Οι σχεσιακές βάσεις δεδομένων είναι οι παραδοσιακές μορφές αποθήκευσης των δεδομένων μιας εφαρμογής στον σκληρό δίσκο. Τα δεδομένα αποθηκεύονται σε πίνακες, οι στήλες των οποίων ονομάζονται πεδία και οι γραμμές εγγραφές. Κάθε εγγραφή, είναι ένα σύνολο τιμών στα πεδία που αποθηκεύει ο πίνακας. Κάθε πίνακας πρέπει να περιλαμβάνει και ένα πεδίο που να μπορεί να καθορίσει με μοναδικό τρόπο κάθε εγγραφή από τις άλλες και αυτό το πεδίο ονομάζεται κλειδί. Ένα παράδειγμα ενός τμήματος ενός πίνακα φαίνεται στην Εικόνα 2.1.

10	Tide	ISBN	Author	Publishing...	isAva ilable
1	1984	2343454895456	George Orwell	04/12/1979	13
2	AnnaKarenina	1234548485843	Leo ToJstcy	07/11/1998	P
4	The Adventures of I	3450345345443	Mark Twain	08/11/1999	O
5	Ulysses	9944933003232	James Joyce	06/05/2010	Q
8	War and Peace	0944344903312	Leo Tolstoy	08/11/2001	●
11	The Brothers Karan	9003940397271	Doso	04/07/2012	B 1
12	On the Road	0459450444310	JackKerouac	30/12/2005	Π=Π
15	The Metamorphosi	2003948930545	Franz Kafka	09/03/1976	o
16	The Illiad	9449039333923	Homer	05/07/1998	o
17	The Odyssey	8409404850139	Homer	06/08/1999	o I

Εικόνα 2.1 Παράδειγμα ενός πίνακα που αποθηκεύει βιβλία σε μια εφαρμογή μιας δανειστικής βιβλιοθήκης.

Οι πίνακες μπορούν να συσχετίζονται μεταξύ τους με συσχετίσεις ένα προς πολλά (ένας Συγγραφέας μπορεί να έχει γράψει πολλά βιβλία) και πολλά προς πολλά (καθένα από πολλά Βιβλία μπορεί να ανήκει σε πολλές Κατηγορίες). Οι συσχετίσεις δείχνουν στα προς συσχέτιση δεδομένα, χρησιμοποιώντας τα κλειδιά άλλων πινάκων. Κλειδιά άλλων πινάκων, που καθορίζουν συσχετίσεις σε έναν πίνακα ονομάζονται ξένα κλειδιά.

Για την σχεδίαση μια σχεσιακής βάσης δεδομένων, συνήθως ακολουθούνται τα παρακάτω βασικά βήματα:

- Η συνολική πληροφορία που πρόκειται να αποθηκεύει κάθε στιγμή η εφαρμογή χωρίζεται σε επιμέρους οντότητες
- Για κάθε οντότητα δημιουργείται ένας πίνακας στην βάση δεδομένων
- Για κάθε οντότητα ξεκαθαρίζεται το πλήθος των χαρακτηριστικών που χρειάζεται να αποθηκεύσει η εφαρμογή
- Για κάθε χαρακτηριστικό μιας οντότητας δημιουργείται και ένα πεδίο στον αντίστοιχο πίνακα
- Καταγράφονται οι συσχετίσεις των οντοτήτων μεταξύ τους
- Για κάθε συσχέτιση ένα προς πολλά, ένα επιπλέον πεδίο προστίθεται στον έναν πίνακα ώστε να καθορίζει μοναδικά την εγγραφή του άλλου πίνακα που υπάρχει η συσχέτιση (ξένο κλειδί)
- Για κάθε συσχέτιση πολλά προς πολλά ένας νέος πίνακας δημιουργείται και περιλαμβάνει πεδία ώστε να καθορίζει μοναδικά τις εγγραφές των δύο άλλων πινάκων που συσχετίζονται (ξένα κλειδιά)

Για παράδειγμα, ας υποθέσουμε ότι δημιουργείται μια εφαρμογή για κάποιο κατάστημα ενοικίασης ταινιών. Οι απαιτήσεις του συστήματος είναι να καταγράφει τα στοιχεία των πελατών, τις διαθέσιμες ή μη διαθέσιμες ταινίες και τις οικονομικές συναλλαγές. Οι οντότητες και άρα οι πίνακες μιας σχεσιακής βάσης θα μπορούσαν να είναι οι Πελάτες, Ταινίες, Συναλλαγές. Τα πιθανά πεδία για τους Πελάτες θα μπορούσαν να είναι Κωδικός Πελάτη, Όνομα, Επίθετο, Τηλέφωνο, Διεύθυνση. Για τις Ταινίες Κωδικός Ταινίας, Σκηνοθέτης, Τίτλος, Έτος, Είδος, Διαθεσιμότητα. Για τις Συναλλαγές Ποσό, Κωδικός Πελάτη, Κωδικός Ταινίας, Ημερομηνία. Ο πίνακας των συναλλαγών προσφέρει μια συσχέτιση πολλά προς πολλά μεταξύ πελατών και ταινιών και για αυτό περιλαμβάνει ως ξένα κλειδιά τα πεδία Κωδικός Πελάτη και Κωδικός Ταινίας.

Οι σχεσιακές βάσεις δεδομένων παρέχουν μηχανισμούς για την ανάγνωση, την εγγραφή, την ανανέωση και την διαγραφή δεδομένων. Αυτού του είδους οι λειτουργίες διενεργούνται συνήθως μέσω της διαδεδομένης γλώσσας προγραμματισμού SQL (Structured Query Language). Οι εντολές στην SQL ονομάζονται ερωτήματα και καλύπτουν τις τέσσερις

παραπάνω βασικές λειτουργίες στα δεδομένα (εγγραφή, ανάγνωση, διαγραφή, ανανέωση). Οι βασικές εντολές μαζί με παραδείγματα χρήσης τους φαίνονται στον πίνακα 2.1.

Λειτουργία	Μέθοδος	Σύνταξη	Παράδειγμα
Εγγραφή	INSERT	INSERT INTO όνομα πίνακα (πεδίο1,πεδίο2,...) VALUES (ΐίμή1,ΐίμή2,...);	INSERT INTO books(author,title,isbn,publishing,isAvailable) VALUES('George Orwell','1984',2343,'4/12/1979',True);
Ανάγωνηση	SELECT	SELECT πεδίο1,πεδίο2,... FROM όνομα πίνακα WHERE συνθήκη;	SELECT author FROM books WHERE title = '1984'
Ανανέωση	UPDATE	UPDATE όνομα πίνακα SET πεδίοΐ = ιμή1,πεδίο2 = ιμή2,... WHERE συνθήκη;	UPDATE books SET isbn 234345 WHERE title = '1984
Διαγραφή	DELETE	DELETE FROM όνομα πίνακα WHERE συνθήκη;	DELETE FROM books WHERE title = '1984';

Πίνακας 2.1 Βασικές εντολές ερωτημάτων σε SQL.

Τα Συστήματα Διαχείρισης Βάσεων Δεδομένων είναι λογισμικά που επιτρέπουν την δημιουργία την ρύθμιση, την διαχείριση και την συντήρηση μιας βάσης δεδομένων. Τα πιο γνωστά τέτοια συστήματα είναι οι PostgreSQL, MySQL, Microsoft SQL Server, Oracle και άλλα.

Τα συστήματα αυτά δίνουν την δυνατότητα διαχείρισης των δεδομένων, εκτέλεσης ερωτημάτων, εξαγωγής αντιγράφων ασφαλείας, ρύθμισης παραμέτρων ασφαλείας,

καθορισμού χρηστών, καθορισμού ρόλων και άλλα. Παρουσιάζουν διαφορές μεταξύ και το καθένα έχει τα δικά του πλεονεκτήματα και μειονεκτήματα, αλλά αυτό ξεφεύγει από τους σκοπούς της παρούσας έρευνας.

Κάθε σύστημα διαχείρισης ορίζει τους δικούς του τύπους δεδομένων τους οποίους μπορεί να ακολουθεί κάθε πεδίο της βάσης, όμως σε γενικές γραμμές, οι βασικοί τύποι που φαίνονται στον πίνακα 2.2 περιλαμβάνονται στα περισσότερα σύχρονα συστήματα διαχείρισης βάσεων δεδομένων. [16]

Τύπος	**Περιγραφή**	**Παράδειγμα**
Numeric	Αριθμός	35
Char	Συμβολοσειρά σταθερού μήκους	"Anna Karenina"
Text	Συμβολοσειρά μεταβλητού μήκους	"Anna Karenina"
Date	Ημερομηνία	"08/07/1994"
Boolean	Λογική τιμή	True
Null	Κενό	NULL

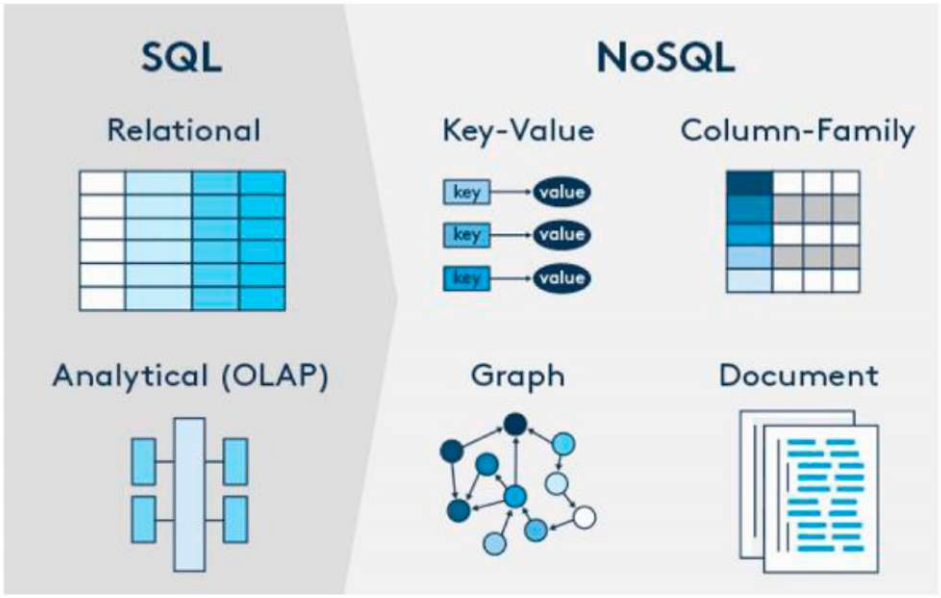
Πίνακας 2.2 Βασικοί τύποι δεδομένων σε σχεσιακές βάσεις δεδομένων.

2.2 Μη σχεσιακές βάσεις δεδομένων

Οι Μη Σχεσιακές Βάσεις Δεδομένων χρησιμοποιούν διαφορετικού είδους μοντέλα δεδομένων από αυτά των σχεσιακών βάσεων. Για τον λόγο αυτό ονομάζονται μη σχεσικές ή NoSQL Βάσεις καθώς ξεφεύγουν από την αρχιτεκτονική και την φιλοσοφία των σχεσιακών βάσεων, ενώ δεν χρησιμοποιούν SQL για την ανάπτυξη ερωτημάτων. Τέτοιου είδους βάσεις αναπτύχθηκαν με στόχο την διαχείριση δεδομένων μεγάλου όγκου, εκεί που οι σχεσιακές βάσεις υστερούσαν σε απόδοση. Κάποιες από τις πιο γνωστές βάσεις αυτού του είδους είναι οι MongoDB, Redis, Apache Cassandra, Apache HBase, Apache CouchDB.

Οι μη σχεσιακές βάσεις έχουν πιο ευέλικτη δομή και θέτουν λιγότερους περιορισμούς στα δεδομένα, γεγονός που τις καθιστά ταχύτερες και πιο αποδοτικές σε μεγάλο όγκο δεδομένων. Επίσης, η αρχιτεκτονική τους, συνήθως επιτρέπει την κατανεμημένη λειτουργία, την λειτουργία, δηλαδή, της βάσης σε μια συστάδα υπολογιστών αντί ενός. Αυτό εκτοξεύει τους διαθέσιμους υπολογιστικούς πόρους, δίνει την δυνατότητα παραλληλοποίησης διαδικασιών και παρέχει καλύτερη κάλυψη λειτουργίας βελτιώνοντας τόσο την απόδοση όσο και την αξιοπιστία της βάσης. Οι NoSQL βάσεις χρησιμοποιούν, επιπλέον, σύγχρονα συστήματα διερεύνησης και αυτόματης ανάκαμψης, ενώ διαχειρίζονται πολύ πιο αποδοτικά τους υπολογιστικούς πόρους του διακομιστή τους. Αυτό τις καθιστά πιο οικονομικές σε κόστος λειτουργίας και συντήρησης.

Οι μη σχεσιακές βάσεις, εκτός από την απόδοση και την κλιμάκωση που προσφέρουν είναι σχεδιασμένες και υλοποιημένες αρκετά χρόνια αργότερα από τις αντίστοιχες σχεσιακές. Έχουν αναπτυχθεί με πολύ πιο σύγχρονα μέσα, στοχεύουν σε πιο σύγχρονους στόχους και χρησιμοποιούν βέλτιστες πρακτικές της εποχής των διαδικτυακών εφαρμογών και των Big Data. Ακόμη, η ευέλικτη δομή αποθήκευσης των δεδομένων τους, εκτός από την απόδοση βελτιώνει και τις ικανότητες αποθήκευσης. Δίνει μεγαλύτερη ελευθερία στον σχεδιαστή και επιτρέπει την μοντελοποίηση πολύ πιο σύνθετων συσχετίσεων.



Εικόνα 2.2 Γραφική αναπαράσταση σχεσιακών και μη σχεσιακών βάσεων ανά τύπο.

Οι βασικοί τύποι μη-σχεσιακών βάσεων είναι οι Βάσεις Τιμών Κλειδιών, οι Βάσεις Στηλών, οι Βάσεις Γραφημάτων και οι Βάσεις Δεδομένων που Προσανατολίζονται σε Έγγραφα. Οι βάσεις τιμών κλειδιών χρησιμοποιούν λεξικά για να αντιστοιχίσουν κάποιο μοναδικό κλειδί στον δείκτη που οδηγεί σε μια τιμή. Οι βάσεις στηλών χρησιμοποιούν επίσης κλειδιά, τα οποία όμως οδηγούν σε πολλαπλές τιμές, τις οικογένειες στηλών. Οι βάσεις γραφημάτων βασίζονται σε ένα γραφικό μοντέλο θεωρίας γράφων, με κόμβους, σχέσεις μεταξύ αυτών και ιδιότητες αυτών. Οι βάσεις που χρησιμοποιούν έγγραφα οργανώνουν τα δεδομένα τους σε συλλογές εγγράφων, ενώ καθένα από αυτά αποτελείται από ζεύγη κλειδιών-τιμών. Οι τελευταίες είναι και αυτές στις οποίες εστιάζει η παρούσα εργασία, καθώς θα αναλυθούν εκτενώς στα επόμενα κεφάλαια, παράλληλα με την μελέτη εργαλείων τέτοιου είδους. Οι τύποι των μη σχεσιακών βάσεων φαίνονται σε σχηματική αναπαράσταση στην Εικόνα 2.2.

Επεκτείνοντας το παράδειγμα της βάσης δεδομένων του καταστήματος ενοικίασης ταινιών, από το παραπάνω κεφάλαιο, θα μπορούσαμε να προσθέσουμε την οντότητα των ηθοποιών,

ώστε να αποθηκεύσουμε πληροφορίες για αυτούς στην βάση. Οι πιθανές συλλογές της νέας βάσης προσανατολισμένης στα έγγραφα θα μπορούσαν πλέον να είναι οι Ταινίες, Πελάτες,

Συναλλαγές, Ηθοποιοί. Κάθε ταινία θα αποθηκεύεται σε ένα έγγραφο με τα πεδία Κωδικός Ταινίας, Σκηνοθέτης, Τίτλος, Έτος, Είδος, Διαθεσιμότητα, Ηθοποιοί να αποτελούν πεδία κάθε εγγράφου. Η διαφορά στην παρούσα βάση είναι ότι στην συσχέτιση ταινιών και ηθοποιών δεν χρειάζεται καμία δήλωση ξένου κλειδιού, ενώ η συσχέτιση μια ταινίας μπορεί να γίνει μέσα στο ίδιο έγγραφο για πολλούς ηθοποιούς ταυτόχρονα. Αυτού του είδους οι συσχετίσεις δεν μπορούν να υπάρξουν σε σχεσιακές βάσεις.

Οι λειτουργίες που υποστηρίζουν οι NoSQL βάσεις είναι παρόμοιες με αυτές που υποστηρίζουν οι σχεσιακές και περιλαμβάνουν εγγραφή, ανάγνωση, ανανέωση και διαγραφή δεδομένων. Οι λειτουργίες αυτές γίνονται χάρη σε συγκεκριμένες εντολές που ορίζει το εκάστοτε σύστημα διαχείρισης της μη σχεσιακής βάσης και δεν είναι τυποποιημένες όπως στις σχεσιακές, αφού δεν χρησιμοποιούν SQL ερωτήματα. Ανάλογα με το είδος τους, οι περισσότερες μη σχεσιακές βάσεις προσφέρουν εργαλεία γραμμής εντολών, υποστήριξη γλωσσών προγραμματισμού και διεπαφές εντολών (APIs) για την λειτουργία, την διασύνδεση, την συντήρηση και την χρήση της βάσης.

Οι μη σχεσιακές βάσεις δεν έχουν αντικαταστήσει τις σχεσιακές και δεν δείχνουν να στοχεύουν σε κάτι τέτοιο. Παρέχουν υψηλότερη απόδοση, κλιμάκωση και πιο ευέλικτη δομή δεδομένων. Αυτό τις καθιστά πιο καλή επιλογή σε δεδομένα τεράστιου όγκου, αλλά και σε ημιδομημένα ή αδόμητα δεδομένα. Επίσης, μπορούν να μοντελοποιήσουν πιο σύνθετες συσχετίσεις δεδομένων. Παρόλα αυτά, η απουσία περιορισμών και αυστηρής δομής οδηγεί σε απουσία συνέπειας και συνοχής. Για χάρη της απόδοσης, οι μη σχεσιακές βάσεις κάνουν συμβιβασμούς στον έλεγχο ολοκλήρωσης διαδικασιών και είναι ευάλωτες σε διαδικασίες που θα πρέπει να γίνονται σύγχρονα. Επίσης, η χρήση τους απαιτεί την γνώση συγκεκριμένων διεπαφών, αλλά και της αρχιτεκτονικής τους. Οι σχεσιακές βάσεις χρησιμοποιούν αρχιτεκτονικές και εργαλεία που είναι εγκατεστημένα πολλά χρόνια και ευρέως διαδεδομένα, ενώ οι μη σχεσιακές πιο εξειδικευμένα και μη τυποποιημένα εργαλεία (πχ διεπαφές API) και αρχιτεκτονικές. Εν κατακλείδι, οι μη σχεσιακές βάσεις μπορούν να συμπληρώσουν τις σχεσιακές και να δώσουν

λύσεις σε ορισμένες εφαρμογές, αλλά δεν μπορούν να τις αντικαταστήσουν εξ' ολοκλήρου. [17]

ΚΕΦΑΛΑΙΟ 3

ELASTIC SEARCH

3.1 Βασικά Χαρακτηριστικά

Η Elasticsearch είναι μια κατανεμημένη μηχανή αναζήτησης κειμένου πραγματικού χρόνου που βασίζεται στην βιβλιοθήκη μηχανών αναζήτησης Apache Lucene. Γεννήθηκε το 2010 και ουσιαστικά, αποτελεί μια NoSQL βάση δεδομένων, που αποθηκεύει έγγραφα μαζί με την μηχανή αναζήτησης σε αυτά. Έχει αναπτυχθεί σε Java, αλλά μπορεί να χρησιμοποιηθεί μέσα από μια μεγάλη ποικιλία γλωσσών προγραμματισμού. Τα περισσότερα τμήματα της Elasticsearch είναι ανοιχτού κώδικα, αλλά κάποια από αυτά έχουν διαφορετικές άδειες. Θεωρείται μια από τις πιο δημοφιλείς μηχανές αναζήτησης.

elasticsearch

Εικόνα 3.1 Elasticsearch

Χρησιμοποιεί δεδομένα JSON και υποστηρίζει διασύνδεση με το πρωτόκολλο HTTP για χρήση σε διαδικτυακές εφαρμογές. Αποτελεί μια κατανεμημένη μηχανή αναζήτησης, δηλαδή μπορεί να χωριστεί σε επιμέρους τμήματα και υποστηρίζει αντίγραφα για κάθε τμήμα. Δίνει την δυνατότητα σύνταξης ερωτημάτων σε μια ειδική γλώσσα πεδίου. Τεράστιες εταιρίες και κατά συνέπεια τεράστιες εφαρμογές, ειδικά διαδικτυακές, χρησιμοποιούν Elasticsearch για τις μηχανές αναζήτησής τους. Ενδεικτικά, μερικές από αυτές είναι το Wikipedia, το Netflix, το GitHub, το Facebook και η Adobe. [6][7][11]

3.1.1 Αναζήτηση πλήρους κειμένου

Η Elasticsearch είναι μια μηχανή αναζήτησης πλήρους κειμένου και αυτό σημαίνει ότι υποστηρίζει αναζήτηση με ακριβείς τιμές αλλά και πλήρες κείμενο. Η αναζήτηση με ακριβείς τιμές αναζητά ένα απόσπασμα κειμένου επακριβώς μέσα σε ένα άλλο κείμενο. Ακόμα και ένα κεφαλαίο γράμμα, ή ένας επιπλέον χαρακτήρας κάνει την διαφορά σε αυτήν την περίπτωση. Για παράδειγμα, οι τιμές "String1" και "string1" δεν ταιριάζουν σύμφωνα με αναζήτηση με ακριβείς τιμές. Το ίδιο και οι τιμές "String1" και "String12". Αυτού του είδους η αναζήτηση είναι χρήσιμη σε περιπτώσεις αναζήτησης ενός κλειδιού, μιας συγκεκριμένης τιμής που είναι κομβικής σημασίας κλπ.

Σε πολλές περιπτώσεις, τα έγγραφα στα οποία θέλουμε να κάνουμε αναζήτηση περιλαμβάνουν γραπτό λόγο και ο χρήστης δεν γνωρίζει ή δεν θυμάται κατά γράμμα την διατύπωση του κειμένου που αναζητά. Τότε, η αναζήτηση πλήρους κειμένου είναι ιδιαίτερα χρήσιμη. Η αναζήτηση πλήρους κειμένου δεν συγκρίνει τιμές, αλλά το κατά πόσο μια τιμή είναι σχετική με τα δεδομένα που είναι αποθηκευμένα. Για παράδειγμα, μια αναζήτηση με βάση το κείμενο "Γιάννενα" θα πρέπει να δίνει και αποτελέσματα που περιλαμβάνουν το "Ιωάννινα", τα κεφαλαία και τα μικρά γράμματα θα πρέπει να μην επηρεάζουν κλπ.

Η Elasticsearch πετυχαίνει την αναζήτηση πλήρους κειμένου, χάρη σε αντεστραμμένα ευρετήρια τα οποία και κατασκευάζει. Η Elasticsearch αναλύει κείμενα, δημιουργώντας λίστες με τις λέξεις που αυτά περιλαμβάνουν, αλλά και λίστες με τις λέξεις και με όλα τα κείμενα στα οποία αυτές εμφανίζονται. Επίσης, άλλου είδους ευρετήρια μοντελοποιούν συσχετίσεις μεταξύ λέξεων που έχουν κοινή σημασία. Τελικά, τα αντεστραμμένα λεξικά κατασκευάζονται έτσι ώστε να διευκολύνουν την γρήγορη αναζήτηση σε πλήρες κείμενο. [8]

3.1.2 Δομή

Το θεμελιώδες δομικό συστατικό της Elasticsearch είναι το έγγραφο. Το έγγραφο αποτελεί ένα σύνολο από ζεύγη κλειδιών-τιμών, αποθηκευμένα σε μορφή JSON. Τα κλειδιά αποτελούν μια συμβολοσειρά που χαρακτηρίζει την πληροφορία, ενώ οι τιμές αποτελούν την πραγματική πληροφορία και μπορεί να είναι οποιουδήποτε αποδεκτού τύπου JSON δεδομένα (ακέραιος, δεκαδικός, συμβολοσειρά κλπ). Ένα έγγραφο μπορεί να περιέχει, επίσης, ένα άλλο έγγραφο, ή ένα μέρος άλλου εγγράφου. Τα έγγραφα, μεταξύ τους, δεν απαιτείται να ακολουθούν κάποια προκαθορισμένη δομή, δηλαδή μπορεί το κάθε ένα να έχει διαφορετικό μήκος ή είδος ζευγών. Ένα παράδειγμα εγγράφου φαίνεται στην Εικόνα 3.2, ενώ αναλυτικά οι τύποι δεδομένων JSON παρουσιάζονται στον Πίνακα 3.1. [6][11]

```
"jobdescription" : "Systems administrator and Linux specialit"
```

Εικόνα 3.2 Παράδειγμα εγγράφου Elasticsearch

Ο αμέσως επόμενος διαχωρισμός της πληροφορίας στην Elasticsearch είναι ο τύπος. Ένας τύπος αποτελεί ένα σύνολο δεδομένων που έχει εννοιολογικά κοινά χαρακτηριστικά και κάποια συνοχή. Ο σχεδιαστής του συστήματος επιλέγει πόσοι και ποιο θα είναι οι τύποι της μηχανής αναζήτησης. Για παράδειγμα, σε μια εφαρμογή διαδικτυακού καταστήματος, 3 τύποι θα μπορούσαν να είναι οι χρήστες, τα προϊόντα και οι κριτικές προϊόντων. [6][9]

Τύπος	**Περιγραφή**	**Παράδειγμα**
Number	Αριθμός	{"age":30}

String	Συμβολοσειρά	{"name": "John"}

Object	Αντικείμενο JSON	{
		"employee": {
		"name": "John",
		"age":30,
		"city":"New York"
		}
		}

Array	Πίνακας	{
		"employees":["john","Anna","Peter"]
		}

Boolean	Λογική Τιμή	{ "sale": true }
Null	Κενό	{"middlename": null}

Πίνακας 3.1 Τύποι δεδομένων JSON.

Συλλογές εγγράφων που διαθέτουν παρόμοια χαρακτηριστικά ομαδοποιούνται σε ευρετήρια στην Elasticsearch. Τα ευρετήρια αποτελούνται από επιμέρους τύπους και κατ' επέκταση από επιμέρους έγγραφα. Χάρη στα ευρετήρια η αναζήτηση γίνεται πολύ πιο αποδοτικά, αφού μέσα από αυτά δρομολογείται η αναζήτηση. Η δημιουργία των ευρετηρίων είναι κάτι που αφορά τον σχεδιαστή και συνδέεται ξεκάθαρα με την απόδοση της μηχανής αναζήτησης. Τα ευρετήρια, φαινομενικά, δίνουν την δυνατότητα κατακόρυφης κλιμάκωση του συστήματος, αλλά όπως θα αναλύσουμε παρακάτω δίνουν την δυνατότητα και της οριζόντιας κλιμάκωσης αυτού.

Τα δεδομένα της μηχανής αναζήτησης αποθηκεύονται σε φυσικά μηχανήματα, που δρουν ως εξυπηρετητές. Διάφορα έγγραφα, τύποι και ευρετήρια αποθηκεύονται σε κόμβους και ο κάθε κόμβος μπορεί να αποθηκεύσει ένα ή περισσότερα ευρετήρια. Οι κόμβοι είναι υπεύθυνοι να αλληλεπιδρούν με τα δεδομένα τα οποία αποθηκεύουν ως διακομιστές.

Το σύνολο των κόμβων της μηχανής αναζήτησης αποτελεί μια συστάδα. Η συστάδα είναι το ανώτερο επίπεδο της οργάνωσης του συστήματος, καθώς οι κόμβοι μιας συστάδας περιλαμβάνουν όλα τα δεδομένα της εφαρμογής. Η οργάνωση του συστήματος με αυτόν τον τρόπο δίνει την δυνατότητα της κεντρικής διαχείρισης όλων των κόμβων και διευκολύνει την αναζήτηση. Η συστάδα, εκτός από τους κόμβους, περιλαμβάνει και ένα ευρετήριο για τους κόμβους ώστε να παρέχει την δυνατότητα αναζήτησης σε αυτούς.

Παρακάτω, στον Πίνακα 3.2, παρουσιάζεται μια σχετική αντιστοιχία των δομικών στοιχείων της Elasticsearch, σε σχέση με τις κλασσικές σχεσιακές βάσεις δεδομένων. Η αντιστοιχία αυτή βοηθάει στην κατανόηση της αρχιτεκτονικής της Elasticsearch, παραλληλίζοντας τα δομικά της χαρακτηριστικά με αυτά των σχεσιακών βάσεων, που είναι ευρύτερα κατανοητά και πιο σαφώς διαχωρισμένα. [6][8][11]

Επίπεδο διαχωρισμού	**Αντιστοιχία σχεσιακών βάσεων**	**Περιγραφή**
Ευρετήριο	Βάση δεδομένων	Μια ολοκληρωμένη δομή αποθήκευσης και διαχείρισης δεδομένων
Τύπος	Πίνακας	Ένα σύνολο αρχείων που περιλαμβάνουν δεδομένα που συσχετίζονται μεταξύ τους
Έγγραφο	Εγγραφή(ή Γραμμή)	Ένα σύνολο από χαρακτηριστικά που αναφέρονται στην ίδια οντότητα
Ζεύγος κλειδιού τιμής	Πεδίο (ή Στήλη)	Ένα χαρακτηριστικό μιας οντότητας

Πίνακας 3.2 Διαχωρισμός Δεδομένων στην Elasticsearch. Σύγκριση με σχεσιακές βάσεις.

3.1.3 Εισαγωγή, επεξεργασία και ανάκτηση δεδομένων

Η Elasticsearch δίνει πλήρως την δυνατότητα στην εισαγωγή, την ανάγνωση, την επεξεργασία και την διαγραφή δεδομένων. Οι λειτουργίες αυτές είναι γνωστές και ως λειτουργίες CRUD (Create, Read, Update, Delete). Η Elasticsearch υποστηρίζει την αρχιτεκτονική REST, κάτι που σημαίνει ότι παρέχει διεπαφή (API) για κάθε μια από τις λειτουργίες CRUD σε ένα προς ένα αντιστοιχία με τις μεθόδους που ορίζει το πρωτόκολλο HTTP (POST, GET, PUT, DELETE). [9][10]

Παρότι η Elasticsearch υποστηρίζει διεπαφή με πολλές γλώσσες προγραμματισμού, ώστε οι βασικές λειτουργίες στα δεδομένα να γίνονται εύκολα, ουσιαστικά, οι λειτουργίες αυτές υλοποιούνται μέσα από μεθόδους HTTP. Τα ερωτήματα υλοποιούνται μέσα από την μέθοδο GET, η δημιουργία εγγράφων μέσα από την μέθοδο POST, η ενημέρωση μέσα από την μέθοδο PUT και η διαγραφή μέσα από την μέθοδο DELETE. Αυτά συνοψίζονται στον Πίνακα 3.3.

Εκτός από την μέθοδο, για να ολοκληρωθεί μια λειτουργία, θα πρέπει να οριστούν και οι κατάλληλες παράμετροι που θα λάβει η μέθοδος. Για παράδειγμα η πληροφορία που θα αποθηκεύσει το νέο έγγραφο που δημιουργεί μια POST κλήση, ή τα κριτήρια με βάση τα οποία θα επιλεχθούν τα έγγραφα που θα διαγράψει μια DELETE κλήση. Αυτές οι παράμετροι ορίζονται σε μορφή JSON. Η χρήση JSON και HTTP μεθόδων καθιστά την Elasticsearch ιδιαίτερα βολική για διαδικτυακές εφαρμογές. Εκτός από τις απλές μεθόδους που ενεργοποιούν βασικές λειτουργίες, η Elasticsearch υποστηρίζει και μαζικές λειτουργίες. Χάρη στο BULK_API, πολλές λειτουργίες μπορούν ομαδοποιηθούν και να εκτελεστούν αυτόματα για πολλαπλά έγγραφα, επιταχύνοντας τις διαδικασίες και διευκολύνοντας τον χρήστη. [9][11]

Λειτουργία	**Μέθοδος**	**Σύνταξη**	**Παράδειγμα**
Create	POST	POST /ευρετήριο/τύπος { Ζεύγη κλειδιών-τιμών σε JSON	POST/catalog/product { "title": "Mastering Elasticsearch", "description":"Mastering Elasticsearch",

		}	"author": "bhavi Dixit", "price": 54.99 }
Read	GET	GET /ευρετήριο/τύποςM	Get /catalog/product/AVrASKqgaBGmnAMj1SBe

Update	PUT	PUT /ευρετήριο/τύποςM
		{
		Ζεύγη κλειδιών-τιμών σε JSON
		}
			PUT /catalog/product/1
			{
			"title":"Elasticsearch for Hadoop",
			"author":"Vishal Shukla",
			"ISBN":"1785288997",
			"price":26.99
			}

Delete	DELETE	DELETE /index/type/id	DELETE /catalog/product/AVrASKqgaBGmnAMj1SBe

Πίνακας 3.3 Μέθοδοι HTTP.

3.1.4 Επιπλέον δυνατότητες

Εκτός από τις παραπάνω βασικές λειτουργίες, η Elasticsearch διαθέτει και κάποια επιπλέον χαρακτηριστικά. Χάρη σε αυτά, η μηχανή αναζήτησης γίνεται πολύ πιο αποδοτική, πιο πλήρης, και πιο εύχρηστη, ενώ διευκολύνει τον προγραμματιστή, αναλαμβάνοντας αρμοδιότητες. Πιο συγκεκριμένα:

Κλιμάκωση

Η αποθήκευση μεγάλου όγκου δεδομένων σε ένα υπολογιστικό κόμβο, αφενός εξαντλεί τον αποθηκευτικό χώρο του κόμβου και αφετέρου απαιτεί μεγάλη υπολογιστική ισχύ από τον κόμβο για λειτουργίες ανάκτησης και εγγραφής. Για να αντισταθμίσει αυτό το πρόβλημα, η Elasticsearch δίνει την δυνατότητα τόσο της κατανομής του συστήματος σε επιμέρους συστήματα προσφέροντας οριζόντια κλιμάκωση, όσο και του καταμερισμού του φόρτου σε επιμέρους εξυπηρετητές. Ένα ευρετήριο μπορεί να χωριστεί σε επιμέρους ευρετήρια, τα οποία

ονομάζονται θραύσματα, τα οποία μοιράζονται σε διαφόρους κόμβους. Χάρη στα θραύσματα, η Elasticsearch αποτελεί μια κατανεμημένη μηχανή αναζήτησης.

Τα θραύσματα μπορούν να χρησιμοποιούνται παράλληλα και από διαφορετικούς κόμβους, ώστε η πληροφορία να εγγράφεται και να ανακτάται πιο αποδοτικά. Αυτό σημαίνει ότι τα ερωτήματα εκτελούνται ταυτόχρονα σε διάφορα θραύσματα, σε διάφορους κόμβους, ώστε το συνολικό έργο που απαιτείται να ολοκληρώνεται ως ένα σύνολο από επιμέρους μικρότερα έργα. Αυτό επιφέρει ταχύτερη εκτέλεση σε ερωτήματα, αλλά και καλύτερη κατανομή του φόρτου και άρα πιο ευέλικτη κατανομή των υπολογιστικών πόρων σε κάθε κόμβο. Η ικανότητα κατανομής της υπολογιστικής ισχύος σε περισσότερους κόμβους μειώνει το κόστος και κάνει πιο σταθερή την μηχανή αναζήτησης. Φυσικά, η οριζόντια κλιμάκωση δεν επηρεάζει τον τελικό χρήστη παρά μόνο με την αύξηση της απόδοσης. Κάθε λειτουργία που λαμβάνει χώρα στην μηχανή αναζήτησης κατανέμεται στους κόμβους και συνενώνεται από αυτούς αυτόματα, αφού αυτό το αναλαμβάνει η Elasticsearch. Για παράδειγμα, κατά την αναζήτηση δεδομένων, η αναζήτηση κατανέμεται ταυτόχρονα σε αρκετούς κόμβους, σε διάφορα θραύσματα, αλλά μόλις ο κάθε ένας δώσει αποτέλεσμα, τα τελικά αποτελέσματα συνενώνονται και παραδίδονται στον χρήστη. [6][8]

Αντίγραφα

Η Elasticsearch παρέχει την δυνατότητα διατήρησης αντιγράφων των δεδομένων σε επίπεδο θραύσματος. Μάλιστα, υπάρχει η δυνατότητα παραμετροποίησης της λειτουργίας αυτής. Ο σχεδιαστής της μηχανής αναζήτησης μπορεί να ορίσει πόσα αντίγραφα θα δημιουργεί η Elasticsearch, για κάθε θραύσμα και σε ποιους κόμβους αυτά θα αποθηκεύονται. Τα αντίγραφα παρέχουν πολλαπλά οφέλη στο σύστημα. Αρχικά αυξάνουν την διαθεσιμότητα των δεδομένων, αφού σε περίπτωση σφάλματος σε έναν κόμβο ή σε ένα θραύσμα, τα δεδομένα μπορούν να ανακτηθούν από διαφορετικές πηγές. Επίσης, προσδίδουν ένα επιπλέον επίπεδο κλιμάκωσης, αφού η Elasticsearch δίνει την δυνατότητα παραλληλοποίησης ερωτημάτων, χρησιμοποιώντας ταυτόχρονα διάφορα αντίγραφα των δεδομένων σε διάφορους κόμβους. Το τελικό αποτέλεσμα συντίθεται από την Elasticsearch πριν προωθηθεί. Τέλος, τα αντίγραφα μπορούν να χρησιμοποιηθούν για την ανάκτηση δεδομένων, σε περίπτωση απώλειας αυτών, λόγω κάποιου σοβαρού σφάλματος στον κόμβο. [6][8][11]

Υποστήριξη γλωσσών προγραμματισμού

Η Elasticsearch παρέχει βιβλιοθήκες που καλύπτουν την διασύνδεση ενός μεγάλου φάσματος γλωσσών προγραμματισμού με την μηχανή αναζήτησης. Οι βιβλιοθήκες αυτές είναι έτοιμος κώδικας που ολοκληρώνει την διασύνδεση εφαρμογής και διακομιστή της μηχανής αναζήτησης, μέσα από εντολές της εκάστοτε γλώσσας. Οι περισσότερες σύγχρονες γλώσσες προγραμματισμού (όπως C, C++, C#, Python, Ruby, Java, Node.js, PHP, Scala, Rust, Swif) υποστηρίζονται από την Elasticsearch, μέσα από επίσημες βιβλιοθήκες ανανεώνονται και συντηρούνται τακτικά. Ακόμη, ένα μεγάλο σύνολο επιπλέον βιβλιοθηκών κυκλοφορεί από την κοινότητα σε μορφή λογισμικού ανοιχτού κώδικα και καλύπτει επιπλέον γλώσσες προγραμματισμού και μεσολογισμικά. Μέσα από αυτές τις βιβλιοθήκες, ο χρήστης είναι ικανός να συνδέσει την εκάστοτε εφαρμογή με μια μηχανή αναζήτησης Elasticsearch, ώστε να ανακτά και να καταγράφει δεδομένα ευκολότερα.

Επίσης, η Elasticsearch παρέχει διάφορα εργαλεία γραμμής εντολών, για την διαχείριση των διαθέσιμων μηχανών αναζήτησης. Τα εργαλεία αυτά, εγκαθίστανται στο κέλυφος του λειτουργικού συστήματος και παρέχουν διάφορες υπηρεσίες, όσον αφορά τις μηχανές αναζήτησης Elasticsearch που υπάρχουν στον υπολογιστή. Χρησιμοποιούνται συχνά για την εισαγωγή και την εξαγωγή δεδομένων, την μετατροπή δεδομένων για λόγους συμβατότητας, την παρακολούθηση της ομαλής λειτουργίας του συστήματος, την αυτοματοποίηση διαδικασιών συντήρησης (πχ αντίγραφα ασφαλείας) και άλλα. [11]

3.2 Δυνατότητες και αδυναμίες

Δυνατότητες

Η Elasticsearch αποτελεί μια κατανεμημένη μηχανή αναζήτησης που επιτρέπει την εξισορρόπηση του φόρτου σε επιμέρους διακομιστές. Αυτό επιτρέπει την παραλληλοποίηση διαδικασιών, άρα την αποδοτικότερη ανάκτηση ή αποθήκευση δεδομένων, την ελαχιστοποίηση του κόστους, αφού οι απαιτούμενοι πόροι είναι λιγότεροι αλλά και στην καλύτερη διαχείριση τεραστίου όγκου δεδομένων. Η Elasticsearch έχει σχεδιαστεί για να είναι αποδοτική σε Big Data. Παρέχει εξειδικευμένα εργαλεία, όπως τα ευρετήρια και την οριζόντια

κλιμάκωση, για να κάνει την διαχείριση δεδομένων μεγάλου όγκου όσο πιο αποδοτική γίνεται. Η διαδικασία εκτέλεσης των ερωτημάτων είναι σε μεγάλο βαθμό παράλληλη, προσφέροντας εξαιρετική απόδοση. Το πιο εντυπωσιακό χαρακτηριστικό της Elasticsearch είναι ότι όλη αυτή η παραλληλοποίηση δεν φαίνεται στο χρήστη, αφού η μηχανή αναζήτησης αναλαμβάνει εξ' ολοκλήρου να μοιράσει τα ερωτήματα στα αντίστοιχα θραύσματα και να συνενώσει τα αποτελέσματα πριν δώσει απαντήσεις.

Ακόμη, η κατανεμημένη φύση της μηχανής αναζήτησης παρέχει ασφάλεια και υψηλή διαθεσιμότητα δεδομένων. Ακόμη και σε περίπτωση σφάλματος σε κάποιο κόμβο ή σε κάποιο θραύσμα, η μηχανή αναζήτησης δύναται να χρησιμοποιήσει κάποιο αντίγραφο ώστε να μην μειωθεί η διαθεσιμότητα των δεδομένων. Και πάλι, το πιο σημαντικό πλεονέκτημα

της Elasticsearch σε αυτόν τον τομέα είναι ότι αυτή η εναλλαγή μεταξύ αντιγράφων δεν γίνεται αντιληπτή από τον χρήστη, αλλά την αναλαμβάνει αποκλειστικά η μηχανή αναζήτησης.

Επίσης, η Elasticsearch παρέχει πολλούς μηχανισμούς που συμβάλουν στην αυτόματη συντήρηση, εποπτεία και βελτιστοποίηση της μηχανής αναζήτησης, διευκολύνοντας τον χρήστη. Παρέχει αρκετά εργαλεία και δυνατότητες που εμπλουτίζουν τις υπηρεσίες που προσφέρει. Για παράδειγμα, υπάρχει η δυνατότητα χρήσης έτοιμων φίλτρων στις συμβολοσειρές αναζήτησης για τον διαχωρισμό των λέξεων, για την μετατροπή σε πεζά για την απομάκρυνση άρθρων μορίων και τετριμμένων λέξεων και άλλα. Ακόμη, η μηχανή αναζήτησης μπορεί να κάνει προτάσεις αυτόματης συμπλήρωσης του όρου αναζήτησης. Τέτοιοι αυτοματισμοί παρέχουν πολύτιμες διευκολύνσεις στον χρήστη, χωρίς να τον επιβαρύνουν στην σχεδίαση, ενώ κάνουν την μηχανή αναζήτησης να προσαρμόζεται περισσότερο στην εκάστοτε εφαρμογή.

Ακόμη, η Elasticsearch εκτός από αποδοτική είναι και ιδιαίτερα εύχρηστη. Παρέχει στον χρήστη και τον σχεδιαστή μια τεράστια ποικιλία από δυνατότητες, ώστε να χρησιμοποιήσει την μηχανή αναζήτησης σύμφωνα με τις ανάγκες της κάθε εφαρμογής. Πρώτα από όλα, στην σχεδίαση παρέχει πολλαπλά επίπεδα αφαιρετικότητας. Σε υψηλό επίπεδο ο σχεδιαστής μπορεί να ορίσει τη συστάδα, τους κόμβους και εν συνεχεία τα θραύσματα και τα αντίγραφα αυτών, θέτωντας λεπτομερώς τις παραμέτρους του υπολογιστικού συστήματος στο οποίο θα λειτουργεί η μηχανή αναζήτησης. Σε χαμηλότερα επίπεδα η Elasticsearch παρέχει πληθώρα επιλογών για την δόμηση των τύπων, των εγγράφων και των ευρετηρίων. Ειδικά χάρη στους τύπους και τα ευρετήρια, ο σχεδιαστής μπορεί να καθορίσει τον τρόπο με τον οποίο η μηχανή αναζήτησης θα ανακτά πληροφορία και άρα να παραμετροποιήσει στον μέγιστο βαθμό την συμπεριφορά της. Επιπρόσθετα, κατά την φάση της χρήσης της μηχανής αναζήτησης σε κάποια

εφαρμογή, η Elasticsearch παρέχει ισχυρές βιβλιοθήκες για τις περισσότερες σύγχρονες γλώσσες προγραμματισμού. Επίσης, παρέχει εύχρηστα εργαλεία, τα φίλτρα, τα ισχυρά ερωτήματα, η πληθώρα τύπων δεδομένων και άλλα.

Η Elasticsearch είναι μια σύγχρονη και από τις πιο δημοφιλείς μηχανές αναζήτησης, καθώς σχεδιάστηκε σε μια εποχή που οι διαδικτυακές εφαρμογές, τα Big Data και οι πιο σύγχρονες μέθοδοι ανάπτυξης εφαρμογών κυριαρχούν , με αποτέλεσμα να σχεδιαστεί ώστε να τα συμπεριλάβει. Εκτός από τις δυνατότητες της μηχανής αναζήτησης σε Big Data, η Elasticsearch επιτρέπει την ταχύτατη και την αρθρωτή ανάπτυξή εφαρμογών σε επιμέρους υπηρεσίες, χάρη στα εργαλεία που παρέχει για την οριζόντια κλιμάκωση, την σχεδίαση και την χρήση της. Πολύ σημαντικό είναι ότι η βάση έχει σχεδιαστεί ειδικά για διαδικτυακές εφαρμογές, υποστηρίζοντας μεθόδους HTTP, REST αρχιτεκτονική, δεδομένα JSON, δυναμικά δεδομένα, στατιστικά και συνεχή κλιμάκωση, καθώς μια εφαρμογή μεγαλώνει.

Σε γενικές γραμμές, η Elasticsearch μπορεί να καλύψει σχεδόν όλο το φάσμα λειτουργιών που απαιτείται να καλύπτει μια μηχανή, αλλά συνήθως αρκετά πιο αποδοτικά. Εκτός αυτού, η Elasticsearch παρέχει και επιπλέον δυνατότητες. Είναι πολύ πιο αποδοτική σε δεδομένα μεγάλου όγκου, είναι κατανεμημένη παρέχοντας οριζόντια κλιμάκωση και παρέχει τεράστιες σχεδιαστικές δυνατότητες και δυνατότητες χρήσης. Φυσικά, όλες αυτές οι δυνατότητες της μηχανής αναζήτησης οδηγούν σε κάποιους περιορισμούς και συμβιβασμούς που πρέπει να καλύπτονται. [6][11]

Περιορισμοί

Η Elasticsearch χρησιμοποιεί ισχυρά ερωτήματα αποδοτικά, χάρη στην ιδιαίτερη διαχείριση μνήμης και του σκληρού δίσκου. Αυτό σημαίνει ότι απαιτείται να έχει τον έλεγχο της μνήμης του διακομιστή, αλλά και να έχει αρκετούς πόρους.

Η Elasticsearch λειτουργεί ιδανικά σε πλήρεις αφιερωμένους διακομιστές, καθώς δεν είναι αποδοτική σε εικονικές μηχανές, σε κοινόχρηστους διακομιστές και σε συστήματα 32bit με μνήμη μικρότερη των 64GB. Σε περίπτωση που η μνήμη RAM του διακομιστή δεν επαρκεί για την ανάκτηση δεδομένων από ένα ερώτημα, η μηχανή ανζήτησης θα χρησιμοποιήσει τον σκληρό δίσκο για επιπλέον μνήμη, ρίχνοντας αρκετά την απόδοση, ή προκαλώντας ακόμα και σφάλματα.

Ακόμη η Elasticsearch απαιτεί συνήθως ταχύτερο σκληρό δίσκο, για να διατηρήσει αποδοτική προσπέλαση σε έγγραφα, γεγονός που ανεβάζει το κόστος της υποδομής.

Η Elasticsearch παρέχει υποστήριξη αποκλειστικά σε δεδομένα JSON. Μπορεί να μπορούν να χρησιμοποιηθούν διεπαφές για διάφορες γλώσσες προγραμματισμού, σε κάθε περίπτωση τα δεδομένα ερωτήματος και απάντησης θα πρέπει να είναι σε JSON, πράγμα που περιορίζει τον χρ^{ήστη}.

Η ελευθερία που παρέχει η Elasticsearch στον σχεδιαστή αυξάνει την πολυπλοκότητα της σχεδίασης και σε μικρές εφαρμογές δεν αποδίδει, ενώ κάνει τα λάθη πιο εύκολα. Ιδιαίτερα η οριζόντια κλιμάκωση και τα αντίγραφα της μηχανής αναζήτησης θα πρέπει να σχεδιαστούν σωστά, ώστε οι επιμέρους κόμβοι να διαχειρίζονται τα δεδομένα σωστά. Η υπερβολική χρήση κόμβων και αντιγράφων μπορεί να οδηγήσει σε επιπλέον κόστος συντήρησης. Επίσης, οι αλλαγές στην δομή των εγγράφων και στον σχεδιασμό του συστήματος απαιτούν πολύ προσοχή και συχνά αναθεώρηση των ευρετηρίων, γεγονός που αυξάνει το κόστος και την προσπάθεια που εμπεριέχει μια επέκταση.

Τέλος, ο διαχωρισμός των αντιγράφων από την κλιμάκωση σε κατανεμημένα συστήματα υπολογιστών παρέχει πολλά πλεονεκτήματα, αλλά προϋποθέτει την σωστή σχεδίαση, ενώ κάνει πιο πολύπλοκη την συνολική βάση.

Οι SQL βάσεις προϋπήρχαν για αρκετά χρόνια και είναι ήδη εγκατεστημένες σε πολλές εφαρμογές. Η SQL είναι μια γλώσσα εξειδικευμένη στην ανάκτηση πληροφορίας και διαχωρίζεται από την εφαρμογή που θα χρησιμοποιήσει αυτά τα δεδομένα.

Η Elasticsearch παρέχει δυνατότητες επιπλέον των παραδοσιακών SQL βάσεων και μηχανών αναζήτησης. Παρέχει ισχυρά ερωτήματα, άρρηκτα συνδεδεμένα με την εφαρμογή, πιο ευέλικτα και πιο εύκολα για χρήση μέσα στην εφαρμογή. Το αποτέλεσμα είναι οι SQL βάσεις να είναι πιο ακόμα διαδεδομένες, κάνοντας τους προγραμματιστές σκεπτικούς στην αλλαγή, ενώ οι αλλαγές που επιφέρει η μεταφορά ενός ήδη υπάρχοντος συστήματος από SQL σε Elasticsearch απαιτεί πόρους και χρόνο. Ακόμη, μια SQL βάση παρέχει πιο επαναχρησιμοποιήσιμα ερωτήματα και είναι πιο εύχρηστη σε περιπτώσεις που συχνά χρειάζεται η πληροφορία να ανακτηθεί εκτός εφαρμογής (πχ από αναλυτές). [6][7][11]

ΚΕΦΑΛΑΙΟ 4

MongoDB

4.1 Βασικά Χαρακτηριστικά

Η MongoDB είναι μια ανοικτού κώδικα βάση δεδομένων που κατατάσσεται στην κατηγορία των NoSQL βάσεων. Μπορεί να λειτουργήσει σε διανομές Linux, αλλά και σε OS X και Windows καλύπτοντας σχεδόν όλο το φάσμα των λειτουργικών συστημάτων. Διαθέτει μηχανισμούς για την αποθήκευση και την ανάκτηση δεδομένων με αρκετά διαφορετικό τρόπο από τις παραδοσιακές σχεσιακές βάσεις δεδομένων.

Εικόνα 4.1 MongoDB

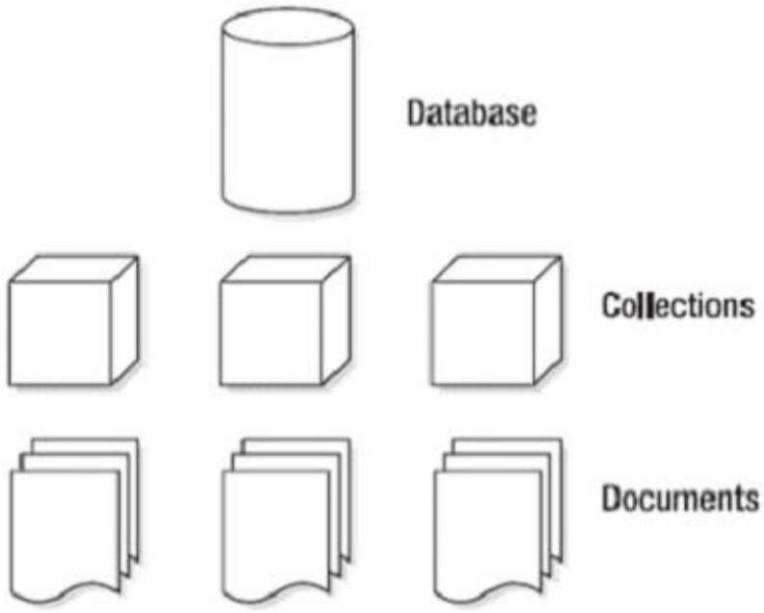
Από την μια, οι σχεσιακές βάσεις δεδομένων διαχωρίζουν τα δεδομένα σε Πίνακες (ή Σχέσεις) και ακόμα περισσότερο σε Εγγραφές (ή Γραμμές) και Πεδία (ή Στήλες). Από την άλλη, η σχεδίαση της MongoDB επιτρέπει τον διαχωρισμό των δεδομένων σε Συλλογές (Collections) και ακόμη περισσότερο σε Έγγραφα (Documents) και σε ζεύγη Κλειδιών-Τιμής (Keys-Values). Αυτή η διαφοροποίηση καθιστά την βάση πολύ πιο αποδοτική, επεκτάσιμη και εύχρηστη σε πολλές περιπτώσεις, ωστόσο αποκλείει ορισμένες εφαρμογές όπου η βάση δεν μπορεί να δώσει

λύσεις. Ο διαχωρισμός των δεδομένων στην MongoDB συνοψίζεται στον Πίνακα 4.1, όπου και περιγράφονται τα επίπεδα αυτού, σε αντιστοιχία με τις σχεσιακές βάσεις δεδομένων. Στην Εικόνα 4.2 παρουσιάζεται μια υψηλού επιπέδου απεικόνιση της σχεδίασης της MongoDB. [2][3].

4.1.1. Συλλογές Έγγραφα και Ζεύγη Κλειδιών-Τιμών

Οι συλλογές σε μια βάση MongoDB είναι σύνολα εγγράφων. Οι συλλογές περιλαμβάνουν έγγραφα που παρουσιάζουν συσχετίσεις, ώστε να μοντελοποιούν πολύπλοκες σχέσεις ανάμεσα στα δεδομένα. Κάθε συλλογή έχει κάποιο όνομα και μπορεί να περιλαμβάνει διαφορετικό πλήθος αρχείων από τις υπόλοιπες.

Τα έγγραφα στην MongoDB είναι δυναμικές μορφές αποθήκευσης δεδομένων, δηλαδή έγγραφα που ανήκουν στην ίδια συλλογή μπορούν να αποτελούνται από διαφορετικό πλήθος και τύπους από ζεύγη Κλειδιών-Τιμών. Ο μόνος περιορισμός είναι το κάθε έγγραφο να έχει μέγεθος έως 16MB. Κάθε έγγραφο έχει ένα μοναδικό αναγνωριστικό κωδικό (_id) που αποτελεί και το πρώτο ζεύγος Κλειδιού-Τιμής, Ο κωδικός αυτός μπορεί να οριστεί από τον χρήστη, ωστόσο συνήθως δημιουργείται αυτόματα από την MongoDB, αφού η βάση παρέχει αυτή την δυνατότητα. Η δομή των εγγράφων καθορίζεται από το πρότυπο BSON που αποτελεί, ουσιαστικά, μια γενίκευση του JSON. Το BSON περιλαμβάνει περισσότερους τύπους δεδομένων από το JSON και η MongoDB αναλαμβάνει να παρέχει την αντιστοιχία δεδομένων από την μορφή BSON στην μορφή που υποστηρίζει η εκάστοτε γλώσσα προγραμματισμού. Ένα παράδειγμα εγγράφου φαίνεται στην Εικόνα 4.3. [2][4]



Εικόνα 4.2 Υψηλού επιπέδου αναπαράσταση της δομής της MongoDB.

Τα ζεύγη κλειδιών-τιμών αποτελούν το δομικό συστατικό των εγγράφων. Το κλειδί δίνει το δικαίωμα στην προσπέλαση της πληροφορίας μέσα από ένα όνομα και θα πρέπει να είναι μοναδικό για κάθε έγγραφο, αφού καθορίζει ένα χαρακτηριστικό αυτού. Η τιμή αποτελεί την αυτούσια πληροφορία και μπορεί να αποθηκεύεται σε οποιοδήποτε τύπο δεδομένου BSON. Η τιμή μπορεί να είναι για παράδειγμα ένας αριθμός, μια συμβολοσειρά, ή ακόμα ένας πίνακας, ένα άλλο έγγραφο ή και ένας πίνακας εγγράφων. Αναλυτικά, οι τύποι δεδομένων του προτύπου φαίνονται στον πίνακα 4.2. Στον πίνακα αυτό, στην στήλη «Παράδειγμα» φαίνεται μια πιθανή όψη κάθε τύπου δεδομένων σε κονσόλα JavaScript. Φυσικά, αυτή η όψη θα μπορούσε να διαφέρει σε άλλη γλώσσα προγραμματισμού, ωστόσο δίνει, ενδεικτικά, μια ιδέα για τον κάθε τύπο δεδομένων.

Εικόνα 4.3 Παράδειγμα εγγράφου MongoDB.

Οι περισσότεροι τύποι δεδομένων BSON που παρουσιάζονται στον Πίνακα 4.2 είναι προφανείς, αλλά κάποιοι όχι. Έχει, λοιπόν, σημασία να σταθούμε στα παρακάτω:

- Οι πίνακες (Array) μπορεί να περιλαμβάνουν οποιουδήποτε τύπου δεδομένα BSON. Το μέγεθός τους είναι μεταβλητό και ο ίδιος πίνακας μπορεί να περιλαμβάνει στοιχεία πολλών διαφορετικών τύπων. Αυτό τους καθιστά, ουσιαστικά, καθόλα δυναμικές δομές δεδομένων.

- Οι Regex είναι συμβολοσειρές γραμμένες σε μια συγκεκριμένη γλώσσα που ονομάζεται Regular Expression. Αυτές οι αλληλουχίες συμβόλων ορίζουν πρότυπα αναζήτησης κειμένου και χρησιμοποιούνται ευρέως σε τέτοιου είδους εφαρμογές.

- Η μόνη διαφορά μεταξύ JavaScript και Scoped JavaScript είναι ότι στην δεύτερη περίπτωση, μια συνάρτηση θα μπορούσε να εκτελεστεί έχοντας πρόσβαση στις μεταβλητές που βρίσκονται στην εμβέλεια του προγράμματος κατά την

εκτέλεση.

- Ο τύπος Object αποτελεί ένα ενσωματωμένο έγγραφο, ενώ ο τύπος ObjectID αποτελεί μια αναφορά σε ένα αντικείμενο, χάρης το μοναδικό αναγνωριστικό του.

• Οι τύποι Maxkey και Minkey αποτελούν εικονικές τιμές που ορίζουν τον μέγιστο και τον ελάχιστο δείκτη του εγγράφου και χρησιμοποιούνται κατά την ταξινόμηση των εγγράφων μιας συλλογής. [3]

Επίπεδο διαχωρισμού	Αντιστοιχία σχεσιακών βάσεων	Περιγραφή
Βάση δεδομένων	Βάση δεδομένων	Μια ολοκληρωμένη δομή αποθήκευσης και διαχείρισης δεδομένων
Συλλογή	Πίνακας	Ένα σύνολο αρχείων που περιλαμβάνουν δεδομένα που συσχετίζονται μεταξύ τους
Έγγραφο	Εγγραφή (ή Γραμμή)	Ένα συνολο από χαρακτηριστικά που αναφέρονται στην ίδια οντότητα
Ζεύγος κλειδιού τιμής	Πεδίο (ή Στήλη)	Ένα χαρακτηριστικό μιας οντότητας

Πίνακας 4.1 Διαχωρισμός Δεδομένων στην MongoDB. Σύγκριση με σχεσιακές βάσεις.

Τύπος	Αριθμός	Περιγραφή	Παράδειγμα
Double	1	Δεκαδικός αριθμός	3.14
String	2	Συμβολοσειρά(UTF-8)	"Hello world"
Object	3	Αντικείμενο BSON	{name:"Tiiri",age:"iTiyob"}
Array	4	Πίνακας	[12,3.14,"Hello"]
Binary	5	Δυαδικά δεδομένα	BinData(2,"DgAAAEltlHNvbWUgYmluYXJ5")
Objectid	7	Αναφορά σε αντικείμενο	0bjectid("4e1bdda65025ea6601560b50")
Boolean	8	Λογική τιμή	True

Date	9	Ημερομηνία και ώρα	ISODate("2011-02-24T21:26:00Z")
Null	10	Κενό	Null
Regex	11	Έκφραση	/test/i
JavaScript	13	Κώδικας JavaScript	Function(){return false;}
Scoped JavaScript	15	Κώδικας JavaScript με χρήση μεταβλητών εμβέλειας	Function(){return false;}
32-bit Integer	16	Ακέραιος 32bit	18
Timestamp	17	Κωδικοποιημένη χρονική σήμανση	{"t":1371429067,"I" : 0}
64-bit Integer	18	Ακέραιος 64bit	NumberLong(18)
Maxkey	127	Μέγιστος αριθμός κλειδιού	{"$maxKey":1}
Minkey	255	Ελάχιστος αριθμός κλειδιού	{"$minKey" : 1}

Πίνακας 4.2 Τύποι Δεδομένων BSON

4.1.2. Συσχετίσεις δεδομένων στην MongoDB

Η MongoDB μπορεί να μοντελοποιήσει πολύ πολύπλοκες συσχετίσεις δεδομένων και μπορεί να το κάνει με πολλούς τρόπους. Πρώτα από όλα, η αρχιτεκτονική με βάση την οποία είναι σχεδιασμένη η βάση επιτρέπει αφενός την ταξινόμηση των δεδομένων σε έγγραφα και αφετέρου την ταξινόμηση των εγγράφων σε συλλογές. Κάθε έγγραφο αποτελεί ένα σύνολο δεδομένων που παρουσιάζουν κάποιο ισχυρό συσχετισμό. Επίσης, μια συλλογή αποτελεί ένα σύνολο εγγράφων τα οποία παρουσιάζουν σαν οντότητες έναν ισχυρό συσχετισμό. Από την άλλη, τα δεδομένα που αποθηκεύονται μέσα σε διαφορετικά αρχεία της ίδιας συλλογής παρουσιάζουν δευτερεύουσες συσχετίσεις διαφορετικού είδους. Αυτού του είδους η μοντελοποίηση των σχέσεων που υπάρχουν μέσα στα δεδομένα δεν είναι πρωτοφανής, αφού μοιάζει αρκετά με τον ιεραρχικό διαχωρισμό των δεδομένων σε πίνακες. Τα πιο ισχυρά

μοντέλα συσχετίσεων που μπορεί να υποστηρίξει η MongoDB είναι αυτά που παρουσιάζονται παρακάτω.

Μια πολύ σημαντική τεχνική μοντελοποίησης συσχετισμού μεταξύ δεδομένων είναι η δυνατότητα της MongoDB να αποθηκεύει ολόκληρα έγγραφα, ενσωματωμένα μέσα σε άλλα έγγραφα. Σε ένα έγγραφο, λοιπόν, μπορεί να συμπεριλαμβάνεται ένα ή περισσότερα έγγραφα, ή ακόμα και πίνακες εγγράφων ώστε να καταγράφονται οι σχέσεις μεταξύ των δεδομένων. Αυτού του είδους οι συσχετισμοί ονομάζονται μη κανονικοποιημένα μοντέλα δεδομένων και επιτρέπουν στα άμεση αλληλεπίδραση με τα δεδομένα που υπάρχει συσχέτιση. Αυτού του τύπου οι συσχετίσεις αποτελούν συσχετίσεις ένα προς ένα, αφού το ενσωματωμένο έγγραφο μπορεί να συσχετιστεί μόνο με το γονικό έγγραφο στο οποίο εμπεριέχεται.

Μια εξίσου σημαντική τεχνική μοντελοποίηση συσχετισμού δεδομένων είναι η δυνατότητα της MongoDB να αποθηκεύει αναφορές προς άλλα έγγραφα, μέσα σε έγγραφα. Σε ένα έγγραφο μπορεί να συμπεριλαμβάνονται, με αυτόν τον τρόπο, ένας ή περισσότεροι σύνδεσμοι προς άλλα έγγραφα ή ακόμα και πίνακες συνδέσμων προς άλλα έγγραφα. Αυτού του είδους οι συσχετισμοί ονομάζονται κανονικοποιημένα μοντέλα δεδομένων και επιτρέπουν στην έμμεση, αλλά και στην πολυεπίπεδη αλληλεπίδραση με τα δεδομένα που υπάρχει συσχέτιση. Αποτελούν, ουσιαστικά συσχετίσεις ένα προς πολλά, ή και πολλά προς πολλά, αφού οποιοδήποτε έγγραφο μπορεί να συσχετιστεί με οποιοδήποτε άλλο μέσα από μια αναφορά σε αυτό. [4][5]

4.1.3. Βασικές λειτουργίες στην MongoDB

Η MongoDB υποστηρίζει όλες τις βασικές λειτουργίες που παρέχουν οι σύγχρονες βάσεις δεδομένων. Οι λειτουργίες αυτές είναι γνωστές και ως CRUD operations (Create-Read-Update-Delete). Πιο συγκεκριμένα:

• Δημιουργία (Create): Δημιουργία ενός ή πολλαπλών εγγράφων σε μια συλλογή. Υλοποιείται μέσα από την εντολή insert().

• Ανάγνωση (Read): Ανάκτηση δεδομένων από μια συλλογή. Η επιλογή των δεδομένων προς ανάκτηση γίνεται με βάση κάποια φίλτρα που ορίζει ο χρήστης. Υλοποιείται μέσα από την εντολή find().

• Ενημέρωση (Update): Επεξεργασία των δεδομένων μιας συλλογής. Για να επιλεχθούν τα δεδομένα που θα ενημερωθούν μπορούν να οριστούν φίλτρα όπως στην περίπτωση της δημιουργίας. Υλοποιείται μέσα από την εντολή update().

• Διαγραφή (Delete): Διαγραφή ορισμένων δεδομένων από μια συλλογή. Και πάλι τα δεδομένα επιλέγονται με φίλτρα που ορίζει ο χρήστης. Υλοποιείται μέσα από την εντολή delete(). [2][3][5]

Λειτουργία	**Εντολή**	**Σύνταξη**	**Παράδειγμα**
Create	Insert	Βάση.συλλογή. insert({πεδία: τιμές})	db. cars. insert ({company:" Ford", Model:" Fiesta", Year:2010 Modifications: [1.25,1.4,1.6]})
Read	find	Βάση.συλλογη.find(κριτήρια)	db. cars. Find ({year: {$gt:2011}})
Update	update	Βάση. συλλογή .update({κριτήρια},	db. cars. update ({year: {$gt: 2008}}

		{ενέργεια}, {επιλογές})	{$set: {model:" Fiesta VII"}}, {multi: true}})
Delete	delete	Βάση.συλλογή^^({κριτήρια})	Db. cars. delete ({year:2009})

Πίνακας 4.3 Βασικά ερωτήματα ανά λειτουργία της MongoDB.

Μια εντολή ανάγνωσης, στη MongoDB, δεν επιστρέφει τα έγγραφα τα οποία πληρούν τα κριτήρια που τίθενται, αλλά έναν κέρσορα σε αυτά, Για να προσπελάσουμε κάθε έγγραφο, θα πρέπει να διατρέξουμε τον κέρσορα. Μια άλλη λειτουργία της MongoDB, χρήσιμη σε περιπτώσεις ανάγνωσης είναι η ταξινόμηση. Υλοποιείται με την εντολή sort() και λαμβάνει σαν είσοδο τα κριτήρια της ταξινόμησης. Τα κριτήρια ανάγνωσης τίθενται σε ένα ερώτημα, χάρη στους τελεστές σύγκρισης που υποστηρίζει η MongoDB, με βαση τους οποίους προκύπτει ένα ταίριασμα εγγράφων. Οι τελεστές σύγκρισης υποστηρίζουν την ακριβή ισότητα με μια ή περισσότερες τιμές ή την εφαρμογή ενός εύρους τιμών και παρουσιάζονται στον Πίνακα 4.4. Πολλαπλές συγκρίσεις μπορούν να συνδυαστούν μέσα από τους τελεστές AND (οι επιμέρους συγκρίσεις ενώνονται με σύζευξη, δηλαδή θα πρέπει να ισχύουν και οι 2 επιμέρους συγκρίσεις) και OR (οι επιμέρους συγκρίσεις ενώνονται με διάζευξη, δηλαδή θα πρέπει να ισχύει τουλάχιστον μία από τις 2 επιμέρους συγκρίσεις). [5]

Λειτουργία	**Εντολή**	**Σύνταξη**
Εισαγωγή ενός	insertOne	db. inventory.insertOne({ item: "canvas", qty: 100, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm"} })

Εισαγωγή πολλαπλών	insertMany	db.inventory.insertMany([{ item: "journal", qty: 25, tags: ["blank", "red"], size:{ h: 14, w: 21, uom: "cm"}}, { item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9,w: 35.5, uom: "cm"}}, { item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm"}},])
Ενημέρωση ενός	updateOne	db.inventory.updateOne({ item: "paper" }, { $set: {"size.uom": "cm", status: "P"}, $currentDate: { lastModified: true } })
Ενημέρωση πολλαπλών	updateMany	db.inventory.updateMany({ "qty": {$it: 50 } }, { $set: {"size.uom": "in", status: "P"}, $currentDate: { lastModified: true } })
Διαγραφή ενός	deleteOne	db.inventory.deleteOne({ status: "D" })
Διαγραφή πολλαπλών	deleteMany	db.inventory.deleteMany({ status: "A" })

Πίνακας 4.4 Τελεστές σύγκρισης της MongoDB

4.1.4. Επιπλέον χαρακτηριστικά στην MongoDB

Εκτός από τις παραπάνω βασικές λειτουργίες, η MongoDB διαθέτει και κάποια επιπλέον χαρακτηριστικά. Χάρη σε αυτά, η βάση γίνεται πολύ πιο αποδοτική, πιο πλήρης, και πιο εύχρηστη, ενώ διευκολύνει τον προγραμματιστή, αναλαμβάνοντας αρμοδιότητες. Πιο συγκεκριμένα:

Ερωτήματα Ad Hoc

Τα ερωτήματα στην MongoDB μπορεί να αποτελούνται από κανόνες σε πεδία όπως ήδη αναφέρθηκε. Ωστόσο, η βάση υποστηρίζει επιπλέον και ερωτήματα εκφρασμένα σε κανονικές εκφράσεις (regular expressions), ερωτήματα που περιλαμβάνουν τυχαιότητα (τυχαίους αριθμούς) αλλά και συναρτήσεις γλωσσών προγραμματισμού (πχ JavaScript) που στέλνονται για εκτέλεση στην βάση. Με αυτό τον τρόπο η MongoDB δίνει αρκετές δυνατότητες στον χρήστη, για κάθε εφαρμογή.

Ευρετήρια

Τα πεδία των εγγράφων της βάσης μπορούν να μετατραπούν σε ευρετήρια, ώστε τα ερωτήματα να εκτελούνται πολύ πιο αποδοτικά από την βάση. Τα ευρετήρια είναι δομές δεδομένων (δέντρα) που αποθηκεύουν επιμέρους δεδομένα από πολλά αρχεία σε ταξινόμηση, ώστε η ανάκτηση πληροφορίας να γίνεται πολύ πιο γρήγορα. Χωρίς αυτά η βάση θα πρέπει να αναζητήσει σε όλα τα έγγραφα μιας συλλογής. Με την χρήση των ευρετηρίων η βάση ανακτά πληροφορία περνώντας από πολύ λιγότερα έγγραφα, αφού τα χρησιμοποιεί σαν πίνακα περιεχομένων, ενώ τα δεδομένα είναι ήδη ταξινομημένα και συχνά δεν χρειάζεται να ταξινομηθούν μετά την ανάκτηση. Η βάση έχει την δυνατότητα να διατηρεί και να συντηρεί τα ευρετήρια βελτιστοποιώντας τα μέσα από τους εσωτερικούς μηχανισμούς της.

Τα ευρετήρια μπορεί να βασίζονται στο _id των εγγράφων, να είναι μονού πεδίου (ένα πεδίο από κάθε έγγραφο) ή να είναι σύνθετα (συνδυασμός περισσότερων πεδίων από κάθε έγγραφο). Στα σύνθετα ευρετήρια υπάρχουν διάφορες επιλογές ταξινόμησης και η επιλογή γίνεται σύμφωνα με την σειρά που θέλουμε η βάση να διατρέξει το ευρετήριο. Τέλος, υπάρχουν και ευρετήρια που καταχωρούν πίνακες τιμών ή ολόκληρα έγγραφα σε κάθε εγγραφή τους. [3][5]

Ο σχεδιαστής της βάσης είναι αυτός που δημιουργεί τα ευρετήρια, ορίζοντας το είδος τους και την ταξινόμησή τους. Στην συνέχεια, η βάση χρησιμοποιεί αυτόματα τα ευρετήρια, ώστε να

εκτελέσει πιο αποδοτικά τα ερωτήματα που θα της τεθούν. Γίνεται λοιπόν σαφές πως η σωστή σχεδίαση των ευρετηρίων παίζει πολύ σημαντικό ρόλο στην απόδοση της βάσης. Για παράδειγμα, τα ευρετήρια μονού πεδίου είναι πιο ευέλικτα στην χρήση, αφού η βάση μπορεί να συνδυάσει διάφορα απλά ευρετήρια για ένα σύνθετο ερώτημα. Από την άλλη, τα σύνθετα ευρετήρια μπορούν να είναι πιο αποδοτικά σε περιπτώσεις που το ερώτημα που θα τεθεί περιλαμβάνει συνδυασμό πεδίων που υπάρχει στο ευρετήριο. Άρα, η σχεδίαση των ευρετηρίων είναι άμεσα συνδεδεμένη με τα ερωτήματα που πρόκειται τεθούν στην βάση, αλλά και με τις συσχετίσεις που έχουν τα δεδομένα που αποθηκεύονται.

Αντίγραφα

Η MongoDB παρέχει την δυνατότητα διατήρησης αντιγράφων της βάσης. Ενώ οι βασικές λειτουργίες λαμβάνουν χώρα στην κύρια βάση, τα δεδομένα αντιγράφονται και σε αντίγραφα αυτής. Τα αντίγραφα μπορούν χρησιμοποιηθούν σε περίπτωση σφάλματος στην κύρια βάση (αυτόματα από την MongoDB) ή ακόμα και να χρησιμοποιηθούν σε περιπτώσεις που δεν θέλουμε να υπερφορτώσουμε την κύρια βάση, ή που θέλουμε να έχουμε διαθέσιμα τα δεδομένα σε περισσότερους διακομιστές. Τα αντίγραφα ονομάζονται δευτερεύουσες βάσεις και ο σχεδιαστής μπορεί να ορίσει τον τρόπο με τον οποίο θα λειτουργούν. Στην συνέχεια η βάση αναλαμβάνει τον συγχρονισμό των δεδομένων. Για παράδειγμα μια δευτερεύουσα βάση μπορεί να οριστεί μόνο για ανάγνωση, ή να περιλαμβάνει μόνο τις αλλαγές που πραγματοποιήθηκαν στην κύρια βάση σαν ιστορικό καταγραφής κ.α.

Κλιμάκωση

Η λειτουργία μια βάσης απαιτεί υπολογιστικούς πόρους από τον διακομιστή στον οποίο λειτουργεί. Η ανάκτηση και η καταχώρηση δεδομένων απαιτεί πόρους τόσο στον σκληρό δίσκο, όσο και στην μνήμη και τον επεξεργαστή. Παραδοσιακά, οι διακομιστές είναι εξοπλισμένοι με ισχυρούς επεξεργαστές, μεγάλη μνήμη και ανθεκτικούς σκληρούς δίσκους. Ωστόσο, η MongoDB προσφέρει την δυνατότητα μιας διαφορετικού είδους κλιμάκωσης, κατακερματίζοντας μια βάση σε επιμέρους μικρότερες βάσεις. Έτσι, κάθε επιμέρους βάση μπορεί να λειτουργεί σε διαφορετικό διακομιστή, κάνοντας την διαχείριση τεράστιου όγκου δεδομένων πιο εύκολη. Ουσιαστικά, μια συλλογή μπορεί να μοιραστεί σε επιμέρους βάσεις-συλλογές που αποτελούν ολοκληρωμένες βάσεις, μειώνοντας δραστικά τον όγκο της πληροφορίας που αποθηκεύει η κάθε μια. Η κατανομή των ερωτημάτων στις εκάστοτε

επιμέρους βάσεις είναι αρμοδιότητα το δρομολογητή ερωτημάτων, ενός μηχανισμού της βάσης που χρησιμοποιεί ειδικά ευρετήρια για να επιλέξει σε ποιες επιμέρους συλλογές-βάσεις θα προωθήσει τα ερωτήματα. Τελικά, η λειτουργία της βάσης σε πολλούς διακομιστές παράλληλα εξισορροπεί το φορτίο και κάνει την βάση πιο αποδοτική και σταθερή. Αυτού του είδους η κλιμάκωση ονομάζεται οριζόντια κλιμάκωση. [2][3][4]

Συγκέντρωση

Η συγκέντρωση είναι μια λειτουργία της MongoDB, η οποία έχει πολλά κοινά με συναρτήσεις aggregation στις SQL βάσεις. Μέσα από αυτήν την λειτουργία, η βάση επιλέγει και ομαδοποιεί δεδομένα από πολλά έγγραφα. Η συγκέντρωση χαρακτηρίζεται από μια συνθήκη επιλογής, με βάση την οποία επιλέγονται τα δεδομένα προς ομαδοποίηση και από μια μέθοδο ομαδοποίησης, με βάση την οποία προκύπτει το τελικό αποτέλεσμα από τα επιλεγμένα δεδομένα. Για παράδειγμα, τέτοιου είδους λειτουργία είναι ο υπολογισμός του αθροίσματος ή του μέσου όρου συγκεκριμένων τιμών. [1]

Υποστήριξη γλωσσών προγραμματισμού

Η MongoDB παρέχει βιβλιοθήκες που καλύπτουν την διασύνδεση ενός μεγάλου φάσματος γλωσσών προγραμματισμού με την βάση. Οι βιβλιοθήκες αυτές είναι έτοιμα προγράμματα που διεκπεραιώνουν την διασύνδεση εφαρμογής και διακομιστή της βάσης, μέσα από εντολές της εκάστοτε γλώσσας. Οι περισσότερες σύγχρονες γλώσσες προγραμματισμού (όπως C, C++, C#, Python, Ruby, Java, Node.js, PHP, Scala, Rust, Swif) υποστηρίζονται από την MongoDB, μέσα από επίσημες βιβλιοθήκες ανανεώνονται και συντηρούνται τακτικά. Ακόμη, ένα μεγάλο σύνολο επιπλέον βιβλιοθηκών κυκλοφορεί από την κοινότητα σε μορφή λογισμικού ανοιχτού κώδικα και καλύπτει επιπλέον γλώσσες προγραμματισμού και μεσολογισμικά. Μέσα από αυτές τις βιβλιοθήκες, ο χρήστης είναι ικανός να συνδέσει την εκάστοτε εφαρμογή με μια βάση MongoDB, ώστε να ανακτά και να καταγράφει δεδομένα ευκολότερα.

Επίσης, η MongoDB παρέχει διάφορα εργαλεία γραμμής εντολών, για την διαχείριση των διαθέσιμων βάσεων. Τα εργαλεία αυτά, εγκαθίστανται στο κέλυφος του λειτουργικού συστήματος και παρέχουν διάφορες υπηρεσίες, όσον αφορά τις βάσεις MongoDB που υπάρχουν στον υπολογιστή. Χρησιμοποιούνται συχνά για την εισαγωγή και την εξαγωγή δεδομένων από την βάση, την μετατροπή δεδομένων για λόγους συμβατότητας, την

παρακολούθηση της ομαλής λειτουργίας της βάσης, την αυτοματοποίηση διαδικασιών συντήρησης (πχ αντίγραφα ασφαλείας) και άλλα. [3][5]

MongoDB Atlas

Το MongoDB Atlas είναι μια πλατφόρμα που δίνει το δικαίωμα εγκατάστασης και χρήσης μια βάσης MongoDB στο υπολογιστικό νέφος (Cloud), σε μια συστάδα υπολογιστών (Cluster). Το MongoDB Atlas παρέχει μάλιστα και δωρεάν πλάνο, το οποίο δίνει την δυνατότητα χρήσης μιας συστάδας υπολογιστών, με έως και 3 κόμβους, με περιορισμό στον όγκο των δεδομένων και με περιορισμένη τεχνική υποστήριξη. Παρόλα αυτά, το δωρεάν πλάνο δίνει τεράστιες δυνατότητες ρύθμισης της βάσης, αρκετό χώρο για τα δεδομένα που απαιτούνται για δοκιμαστική χρήση ή για μικρές εφαρμογές, αυτόματη εισαγωγή δοκιμαστικών δεδομένων μεγάλου όγκου για δοκιμές και λεπτομερείς ρυθμίσεις ασφαλείας. Τέλος, κατά την δημιουργία μιας συστάδας, η MongoDB Atlas δίνει την δυνατότητα επιλογής του παρόχου του υπολογιστικού συστήματος (μεταξύ Amazon Web Services, Google Cloud και Microsoft Azure) και της περιοχής που αυτό είναι εγκατεστημένο.

Το μεγάλο πλεονέκτημα του MongoDB Atlas είναι ότι κάποιος σχεδιαστής μπορεί να δημιουργήσει, να συντηρεί και να χειρίζεται μια βάση χωρίς να χρειαστεί να εγκαταστήσει κανένα λογισμικό στον υπολογιστή του, ενώ διατηρεί την απόλυτη συμβατότητα με κάθε εφαρμογή, μέσω του διαδικτύου. Αυτό το πλεονέκτημα απαλλάσσει από τον περιορισμό του υλικού και από τα κόστη συντήρησης αυτού, όπως και από προβλήματα συμβατότητας. Επίσης, η δυνατότητα εισαγωγής δεδομένων μεγάλου όγκου δίνει την ευκαιρία για την ταχύτατη ανάπτυξη ενός πειραματικού περιβάλλοντος για δοκιμές, χωρίς να χρειάζεται ο σχεδιαστής να εισάγει αυτά τα δεδομένα χειροκίνητα, εξοικονομώντας πολύ χρόνο και ενέργεια στον σχεδιαστή. Ιδιαίτερα σε ερευνητικές και εκπαιδευτικές εφαρμογές το MongoDB Atlas είναι ένα πάρα πολύ χρήσιμο εργαλείο χωρίς κόστος.

4.2 Δυνατότητες και αδυναμίες

Δυνατότητες

Η MongoDB αποτελεί μια κατανεμημένη βάση δεδομένων που επιτρέπει την εξισορρόπηση του φόρτου σε επιμέρους βάσεις. Αυτό συμβάλει στην παραλληλοποίηση διαδικασιών, άρα στην αποδοτικότερη ανάκτηση και αποθήκευση δεδομένων, στην ελαχιστοποίηση του κόστους, αφού οι απαιτούμενοι πόροι είναι λιγότεροι αλλά και στην καλύτερη διαχείριση τεραστίου όγκου δεδομένων. Η MongoDB έχει σχεδιαστεί για να είναι αποδοτική σε Big Data. Παρέχει εξειδικευμένα εργαλεία, όπως τα ενσωματωμένα έγγραφα, τα ευρετήρια και την οριζόντια κλιμάκωση, για να κάνει την διαχείριση δεδομένων μεγάλου όγκου όσο πιο αποδοτική γίνεται.

Ένα από τα μεγαλύτερα προσόντα της βάσης, είναι η ικανότητά της να μοντελοποιεί με πολύ αποτελεσματικό τρόπο συσχετίσεις μέσα σε δεδομένα τεράστιου όγκου. Οι συσχετίσεις μπορεί να υλοποιούνται μέσα από αναφορές σε έγγραφα, μέσα από ενσωματωμένα έγγραφα αλλά και μέσα από ευρετήρια. Εκτός από τις σχεδιαστικές επιλογές, αξιοσημείωτο είναι και το γεγονός ότι τα δεδομένα στην MongoDB αποθηκεύονται σε δυναμικά έγγραφα, τα οποία παρά την διαφοροποίησή τους διατηρούν συσχετίσεις μεταξύ τους.

Επίσης, η MongoDB παρέχει πολλούς μηχανισμούς που συμβάλουν στην αυτόματη συντήρηση και βελτιστοποίηση της βάσης, διευκολύνοντας τον χρήστη. Για παράδειγμα, υπάρχει η δυνατότητα ορισμού ευρετηρίων καθορισμένου χρόνου, ώστε να διαγράφονται αυτόματα προσωρινά αρχεία, ή σταθερού μεγέθους ώστε να κατακερματίζεται αυτόματα η βάση. Αυτοί οι αυτοματισμοί παρέχουν πολύτιμες υπηρεσίες στον χρήστη, χωρίς να τον επιβαρύνουν στην σχεδίαση, ενώ κάνουν την βάση να προσαρμόζεται περισσότερο στην εκάστοτε εφαρμογή.

Ακόμη, η MongoDB εκτός από αποδοτική είναι και ιδιαίτερα εύχρηστη. Παρέχει στον χρήστη και τον σχεδιαστή μια τεράστια ποικιλία από δυνατότητες, ώστε να χρησιμοποιήσει την βάση σύμφωνα με τις ανάγκες της κάθε εφαρμογής. Πρώτα από όλα, στην σχεδίαση παρέχει πολλαπλά επίπεδα αφαιρετικότητας. Σε υψηλό επίπεδο ο σχεδιαστής μπορεί να ορίσει την οριζόντια κλιμάκωση και τα αντίγραφα της βάσης ορίζοντας λεπτομερώς τις παραμέτρους του υπολογιστικού συστήματος στο οποίο θα λειτουργεί η βάση. Σε χαμηλότερα επίπεδα η βάση

παρέχει πληθώρα επιλογών για την δόμηση των συλλογών, των εγγράφων και των ευρετηρίων. Ειδικά χάρη στα ευρετήρια, ο σχεδιαστής μπορεί να καθορίσει τον τρόπο με τον οποίο η βάση θα ανακτά πληροφορία και άρα να παραμετροποιήσει στον μέγιστο βαθμό την συμπεριφορά της. Επιπρόσθετα, κατά την φάση της χρήσης της βάσης σε κάποια εφαρμογή, η MongoDB παρέχει ισχυρές βιβλιοθήκες για τις περισσότερες σύγχρονες γλώσσες προγραμματισμού. Επίσης, παρέχει εύχρηστα εργαλεία, όπως οι συναρτήσεις συγκέντρωσης, τα ισχυρά ερωτήματα, η πληθώρα τύπων δεδομένων και άλλα.

Η MongoDB είναι μια από τις πιο σύγχρονες βάσεις, καθώς σχεδιάστηκε σε μια εποχή που οι διαδικτυακές εφαρμογές, τα Big Data και οι πιο σύγχρονες μέθοδοι ανάπτυξης εφαρμογών υπήρχαν ήδη, με αποτέλεσμα να σχεδιαστεί ώστε να τα συμπεριλάβει. Εκτός από τις δυνατότητες της βάσης σε Big Data που αναφέρθηκαν παραπάνω, η βάση επιτρέπει την ταχύτατη και την αρθρωτή ανάπτυξή εφαρμογών σε επιμέρους υπηρεσίες, χάρη στα εργαλεία που παρέχει για την οριζόντια κλιμάκωση, την σχεδίαση και την χρήση της. Πολύ σημαντικό είναι ότι η βάση έχει σχεδιαστεί ειδικά για διαδικτυακές εφαρμογές, υποστηρίζοντας τέτοιου είδους γλώσσες, δεδομένα BSON, δυναμικά δεδομένα, καταγράφοντας στατιστικά και προσφέροντας συνεχή κλιμάκωση, καθώς μια εφαρμογή μεγαλώνει.

Σε γενικές γραμμές, η MongoDB μπορεί να καλύψει σχεδόν όλο το φάσμα λειτουργιών που παρέχουν οι παραδοσιακές, SQL βάσεις, αλλά συνήθως αρκετά πιο αποδοτικά. Εκτός αυτού, η MongoDB παρέχει και επιπλέον δυνατότητες, πέρα από αυτές που θα μπορούσαν να φτάσουν οι παραδοσιακές, SQL βάσεις. Είναι πολύ πιο αποδοτική σε δεδομένα μεγάλου όγκου, είναι κατανεμημένη βάση παρέχοντας οριζόντια κλιμάκωση και παρέχει τεράστιες σχεδιαστικές δυνατότητες και δυνατότητες χρήσης. Φυσικά, όλες αυτές οι δυνατότητες της βάσης οδηγούν σε κάποιους περιορισμούς και συμβιβασμούς που πρέπει να καλύπτονται.

Περιορισμοί

Η MongoDB χρησιμοποιεί ισχυρά ερωτήματα αποδοτικά, χάρη στην ιδιαίτερη διαχείριση μνήμης. Αυτό σημαίνει ότι απαιτείται να έχει τον έλεγχο της μνήμης του διακομιστή, αλλά και να έχει αρκετούς πόρους. Η MongoDB λειτουργεί ιδανικά σε πλήρους αφιερωμένους διακομιστές, καθώς δεν είναι αποδοτική σε εικονικές μηχανές, σε κοινόχρηστους διακομιστές και σε συστήματα 32bit. Σε περίπτωση που η μνήμη RAM του διακομιστή δεν επαρκεί για την

ανάκτηση δεδομένων από ένα ερώτημα, η βάση θα χρησιμοποιήσει τον σκληρό δίσκο για επιπλέον μνήμη, κάνοντας ρίχνοντας αρκετά την απόδοση. Ακόμη, μια βάση MongoDB απαιτεί συνήθως περισσότερο χώρο στον σκληρό δίσκο, αφού αποθηκεύει έγγραφα αντί πινάκων.

Σε περίπτωση που οι συσχετίσεις των δεδομένων που θα αποθηκευτούν στην βάση δεν είναι ισχυρές, ή που δεν σχεδιαστούν σωστά η βάση παύει να είναι εύχρηστη και αποδοτική. Η συνένωση συλλογών στην MongoDB είναι μια απαιτητική διαδικασία και πολλές φορές είναι ανέφικτο ή δύσκολο να γίνουν πολύπλοκες συσχετίσεις, όταν δεν έχουν σχεδιαστεί κατάλληλα τα ευρετήρια και τα έγγραφα της βάσης.

Η ελευθερία που παρέχει η MongoDB στον σχεδιαστή αυξάνει την πολυπλοκότητα της σχεδίασης και πολλές φορές κάνει τα λάθη πιο εύκολα. Ιδιαίτερα η οριζόντια κλιμάκωση και τα αντίγραφα της βάσης θα πρέπει να σχεδιαστούν σωστά, ώστε οι επιμέρους βάσεις να διαχειρίζονται τα δεδομένα σωστά. Η υπερβολική χρήση κλιμάκωσης και αντιγράφων μπορεί να οδηγήσει σε επιπλέον κόστος συντήρησης της βάσης. Τέλος, ο διαχωρισμός των αντιγράφων από την κλιμάκωση σε κατανεμημένα συστήματα υπολογιστών παρέχει πολλά πλεονεκτήματα, αλλά προϋποθέτει την σωστή σχεδίαση, ενώ κάνει πιο πολύπλοκη την συνολική βάση.

Οι SQL βάσεις προϋπήρχαν για αρκετά χρόνια και είναι ήδη εγκατεστημένες σε πολλές εφαρμογές. Η SQL είναι μια γλώσσα εξειδικευμένη στην ανάκτηση πληροφορίας και διαχωρίζεται από την εφαρμογή που θα χρησιμοποιήσει αυτά τα δεδομένα. Η MongoDB παρέχει δυνατότητες επιπλέον των παραδοσιακών SQL βάσεων. Παρέχει ισχυρά ερωτήματα, άρρηκτα συνδεδεμένα με την εφαρμογή, πιο ευέλικτα και πιο εύκολα για χρήση μέσα στην εφαρμογή. Το αποτέλεσμα είναι οι SQL βάσεις να είναι πιο διαδεδομένες, κάνοντας τους προγραμματιστές σκεπτικούς στην αλλαγή, ενώ οι αλλαγές που επιφέρει η μεταφορά ενός ήδη υπάρχοντος συστήματος από SQL σε MongoDB απαιτεί πόρους και χρόνο. Ακόμη, μια SQL βάση παρέχει πιο επαναχρησιμοποιήσιμα ερωτήματα και είναι πιο εύχρηστη σε περιπτώσεις που συχνά χρειάζεται η πληροφορία να ανακτηθεί εκτός εφαρμογής (πχ από αναλυτές). [5]

ΚΕΦΑΛΑΙΟ 5

LUCENE

5.1 Βασικά Χαρακτηριστικά

Η Apache Lucene είναι μια δωρεάν και ανοιχτού κώδικα βιβλιοθήκη λογισμικού μηχανών αναζήτησης. Αρχικά γράφτηκε σε Java από τον Doug Cutting το 1999 και υποστηρίζεται από το Apache Software Foundation, στην οποία και προσχώρησε το 2001. Η βιβλιοθήκη κυκλοφορεί με την άδεια λογισμικού Apache, ενώ λειτουργεί και συντηρείται για περισσότερα από 20 χρόνια. Το Lucene χρησιμοποιείται ευρέως ως τυπική βάση για μη ερευνητικές εφαρμογές αναζήτησης. Μέχρι σήμερα εξακολουθεί να είναι ένα από τα βασικά και πιο ενεργά έργα στην οικογένεια του ιδρύματος Apache.

Η βιβλιοθήκη αυτή παρέχει αναζήτηση πλήρους κειμένου, Χρησιμοποιείται ευρέως, σε πληθώρα εφαρμογών, ουσιαστικά ως μια βιβλιοθήκη λογισμικού μηχανών αναζήτησης πλήρους κειμένου που παρέχει μια πλατφόρμα αναζήτησης και ευρετηρίασης.

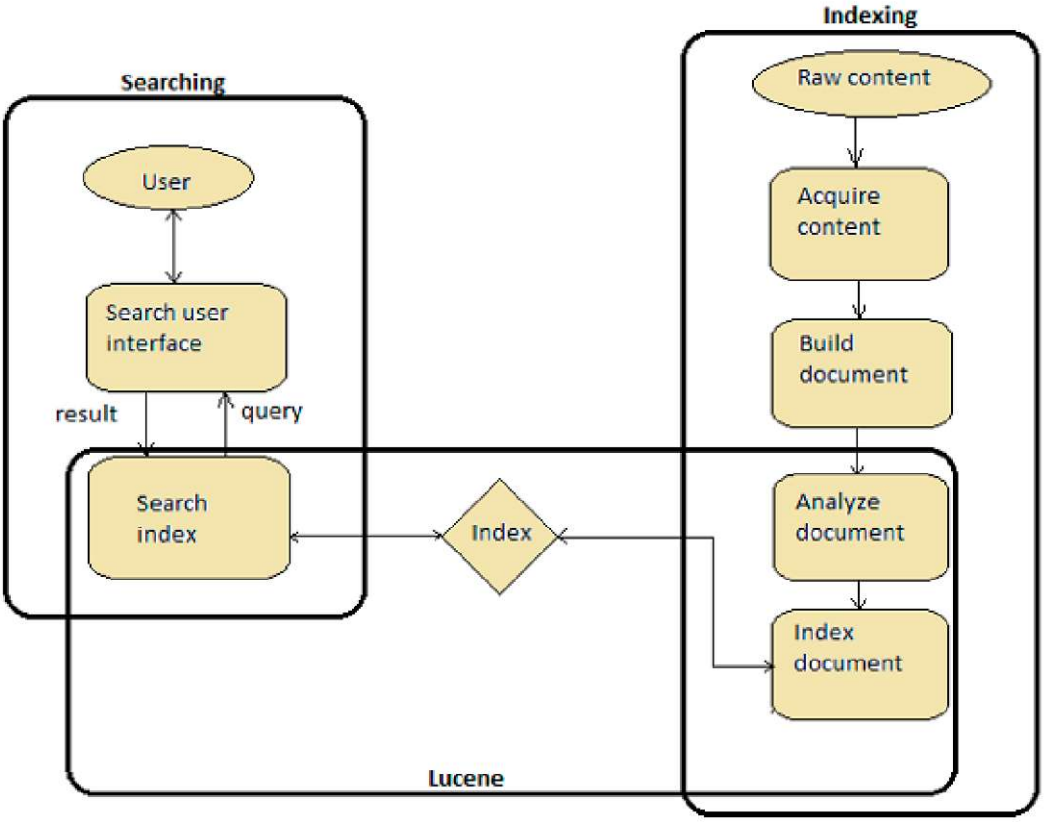
-LUCENE

Εικόνα 5.1 Apache Lucene.

Η Lucene επιτρέπει την προσθήκη δυνατοτήτων αναζήτησης σε ιστότοπους και εφαρμογές. Λαμβάνει περιεχόμενο και το προσθέτει σε ένα ευρετήριο πλήρους κειμένου το οποίο μπορεί στη συνέχεια να χρησιμοποιηθεί για την εκτέλεση ερωτημάτων. Το περιεχόμενο που προστίθεται στο ευρετήριο μπορεί να απορροφηθεί από οποιονδήποτε αριθμό πηγών, όπως βάσεις δεδομένων SQL/NoSQL ή ακόμη και από τον ίδιο τον ιστότοπο. [12][13]

Αρχιτεκτονική

Η Lucene λειτουργεί ως το βασικό συστατικό μιας εφαρμογής αναζήτησης, παρόλα αυτά δεν παρέχει όλη τη λειτουργικότητα τέτοιων εφαρμογών. Η Εικόνα 5.2 μας δείχνει ένα σχήμα τυπικής εφαρμογής αναζήτησης. Λαμβάνοντας υπόψη ότι μόνο ορισμένες διαδικασίες διεκπεραιώνονται από τη Lucene οι υπόλοιπες εργασίες πρέπει να αντιμετωπιστούν ξεχωριστά και είναι ευθύνη της εφαρμογής.



Εικόνα 5.2 Αρχιτεκτονική τυπικής εφαρμογής αναζήτησης και μερίδιο της Lucene.

Η Lucene είναι υπεύθυνη για την ανάλυση του κειμένου, για την ευρετηρίαση και για την αναζήτηση. Ότι αφορά την δομή των εγγράφων, το γραφικό περιβάλλον, τις επαληθεύσεις δεδομένων ή οτιδήποτε άλλο μέσα σε μια εφαρμογή είναι εκτός Lucene και ο σχεδιαστής θα πρέπει να αναπτύξει μια δική του ανεξάρτητη υλοποίηση για αυτά.

Έγγραφα

Η Lucene διαχειρίζεται απλό κείμενο. Οι εφαρμογές που χτίζουν τις δυνατότητες αναζήτησής τους πάνω στη Lucene ενδέχεται να υποστηρίζουν έγγραφα σε διάφορες μορφές, όπως HTML, XML, PDF κλπ, αλλά συνήθως χρησιμοποιούν δομές αρχείων που περιγράφηκαν παραπάνω, τα οποία αποθηκεύουν δεδομένα μέσα από σύνολα πεδίων και τιμών. Η Lucene δεν εμπλέκεται στην ανάλυση των μορφών των εγγράφων καθώς αυτό είναι ευθύνη της εκάστοτε εφαρμογής. Η εφαρμογή που αξιοποιεί τη Lucene θα πρέπει να χρησιμοποιήσει ένα κατάλληλο σύστημα για να μετατρέψει την αρχική μορφή σε απλό κείμενο πριν μεταβιβάσει αυτό το απλό κείμενο στη Lucene. Επίσης, η εφαρμογή είναι υπεύθυνη για την δημιουργία και την διαχείριση του συνόλου των αρχείων καθώς η Lucene δεν εμπλέκεται σε αυτό.

Ανάλυση και διακριτικά

Το απλό κείμενο που μεταβιβάζεται στη Lucene περνά από μια διαδικασία που ονομάζεται ανάλυση με σκοπό την διακριτοποίηση της πληροφορίας (Tokenization) και που αποτελεί το πρώτο βήμα σε για την δημιουργία ενός ευρετηρίου. Αυτή είναι η διαδικασία διάσπασης του κειμένου εισόδου σε μικρά στοιχεία ευρετηρίασης (διακριτικά). Ο τρόπος με τον οποίο το αρχικό κείμενο χωρίζεται σε διακριτικά καθορίζει σε μεγάλο βαθμό τον τρόπο με τον οποίο οι χρήστες θα μπορούν στη συνέχεια να αναζητήσουν αυτό το κείμενο. Για παράδειγμα, η αρχή και το τέλος των προτάσεων μπορούν να προσδιοριστούν για να παρέχουν πιο ακριβείς αναζητήσεις φράσεων και εγγύτητας (αν και η αναγνώριση προτάσεων δεν παρέχεται από τη Lucene).

Σε ορισμένες περιπτώσεις δεν αρκεί απλώς η διάσπαση του αρχικού κειμένου σε διακριτικά αλλά μπορεί αυτά να χρειαστούν και μια επιπλέον επεξεργασία. Η Lucene περιλαμβάνει εγκαταστάσεις ανάλυσης-επεξεργασίας τόσο πριν όσο και μετά τη δημιουργία των διακριτικών. Η ανάλυση πριν από τη δημιουργία διακριτικών μπορεί να περιλαμβάνει (αλλά δεν περιορίζεται σε) την αφαίρεση σήμανσης HTML και τον μετασχηματισμό ή την αφαίρεση κειμένου που ταιριάζει με αυθαίρετα μοτίβα ή σύνολα σταθερών συμβολοσειρών.

Η επεξεργασία των διακριτικών μετά την δημιουργία τους αποτελείται από διάφορα βήματα. Τα βήματα αυτά περιλαμβάνουν (αλλά δεν περιορίζονται σε):

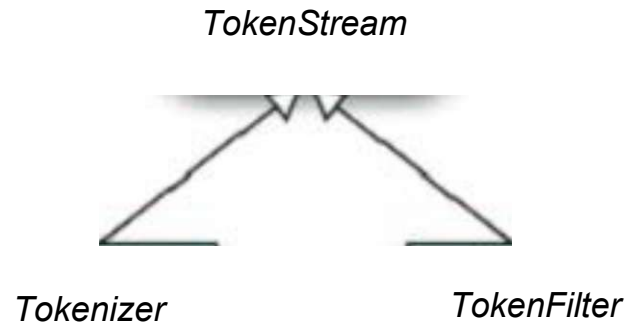
• Στέλεχος - Αντικατάσταση λέξεων με τους μίσχους τους. Κατά την διαδικασία αυτή η κάθε λέξη αντικαθίσταται από το μικρότερο δυνατό υποσύνολο γραμμάτων της που την εκφράζει. Για παράδειγμα, η αγγλική λέξη "bikes" αντικαθίσταται από το "bike". Έτσι, το ερώτημα "bike" μπορεί να βρει τόσο έγγραφα που περιέχουν το "bike" όσο και αυτά που περιέχουν το "bikes".

• Διακοπή φιλτραρίσματος λέξεων - Κατά την διαδικασία αυτή, κοινές βοηθητικές λέξεις (άρθρα, μόρια, σύνδεσμοι κλπ) όπως "το", "και" και "α" αφαιρούνται. Τέτοιου είδους λέξεις σπάνια προσθέτουν αξία σε μια αναζήτηση, ενώ η αφαίρεσή τους συρρικνώνει το μέγεθος του ευρετηρίου και αυξάνει την απόδοση. Μπορεί επίσης να μειώσει κάποιο "θόρυβο" και να βελτιώσει αισθητά την ποιότητα αναζήτησης.

• Κανονικοποίηση κειμένου - Κατά την διαδικασία αυτή αφαιρούνται ειδικοί χαρακτήρες και σύμβολα. Η απογύμνωση του τονισμού και λοιπών συμβόλων μπορεί να συμβάλει στην καλύτερη αναζήτηση, καθώς κάνει πιο ευρύ και γενικό το ευρετήριο.

• Επέκταση συνωνύμων - Κατά την διαδικασία αυτή προστίθενται συνώνυμα και αντιστοιχίζονται με κάθε λέξη του ευρετηρίου. Η προσθήκη συνωνύμων στην ίδια θέση ένδειξης με κάποια λέξη μπορεί να οδηγήσει σε καλύτερη αντιστοίχιση όταν οι χρήστες αναζητούν λέξεις στο σύνολο των συνωνύμων. [12][13]

Το πακέτο ανάλυσης παρέχει τον μηχανισμό μετατροπής συμβολοσειρών και αναγνωστών σε διακριτικά που μπορούν να ευρετηριαστούν από τη Lucene. Η Lucene δίνει την δυνατότητα δόμησης της διαδικασίας ανάλυσης, ώστε να παραχθούν αποδοτικά τα διακριτικά και τα ευρετήρια της εκάστοτε εφαρμογής. Υπάρχουν δύο κύριες κατηγορίες αναλυτών και από αυτούς προκύπτουν όλοι οι επιμέρους. Η Lucene παρέχει, μάλιστα και έτοιμους αναλυτές που μπορούν να χρησιμοποιηθούν ή να συνδυαστούν αυτούσιοι. Φυσικά, υπάρχει η δυνατότητα δημιουργίας προσαρμοσμένων αναλυτών. Η συνολική διαδικασία την ανάλυσης είναι η τελική αλληλουχία επιμέρους αναλυτών που θα χρησιμοποιηθούν.

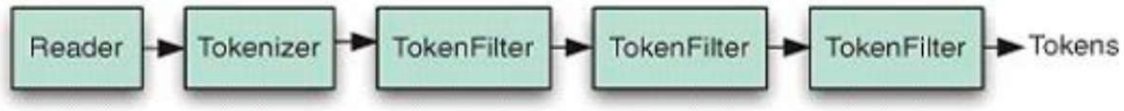


Εικόνα 5.3 βασικές κατηγορίες αναλυτών και κληρονομικότητα

Οι Tokenizers είναι αναλυτές που είναι υπεύθυνοι για τον κατακερματισμό ενός αρχικού κειμένου σε διακριτικά (Tokens). Συνήθως αποτελούν ένα από τα πρώτα βήματα της ανάλυσης. Χαρακτηρίζονται από τον τρόπο με τον οποίο διασπούν το κείμενο σε διακριτικά.

Τα TokenFilters είναι αναλυτές που είναι υπεύθυνοι για την επεξεργασία ενός αρχικού συνόλου από διακριτικά (Tokens), ώστε να παραχθούν νέα διακριτικά. Συνήθως ακολουθούν μετά από έναν ή περισσότερους Tokenizers, λαμβάνοντας είσοδο τα διακριτικά που αυτοί παράγουν ως έξοδο. Χαρακτηρίζονται από τον τρόπο με τον οποίο φιλτράρουν και τροποποιούν τα διακριτικά.

Η Lucene παρέχει ένα σύνολο από αναλυτές έτοιμους για χρήση. Αυτοί καλύπτουν ένα ευρύ φάσμα εφαρμογών και αποτελούν τόσο Tokenizers, όσο και TokenFilters. Οι έτοιμοι αναλυτές είναι μέρος του πυρήνα της Lucene και είναι παραμετροποιημένοι, ώστε να είναι εύχρηστοι και να ταιριάζουν σε κάθε εφαρμογή. Οι πιο βασικοί αναλυτές της βιβλιοθήκης παρουσιάζονται στον Πίνακα 5.1, καθένας με μια σύντομη περιγραφή. Στην Εικόνα 5.3 φαίνονται οι βασικές κατηγορίες αναλυτών και οι σχέσεις κληρονομικότητας αυτών, ενώ στην Εικόνα 5.4 η βασική αφηρημένη δομή μιας πλήρους ανάλυσης κειμένου σε Lucene.



Εικόνα 5.4 Παράδειγμα πλήρους ανάλυσης κειμένου σε Lucene

Όνομα	**Περιγραφή**	**Είδος**
TokenStream	Αφηρημένη κλάση την οποία κληρονομούν Tokenizers και TokenFilters	Αφηρημένο
Tokenizer	Αναλυτές που κατακερματίζουν κείμενο σε διακριτά	Tokenizer
CharTokenizer	Γενική αφηρημένη κλάση Tokenizer που δίνει δυνατοτήτες για υλοποίηση διάσπασης σε διακριτικά και κανονικοποίησης	Tokenizer
WhitespaceTokenizer	Tokenizer που διαχωρίζει διακριτικά με βάση ολόκληρες συμβολοσειρές	Tokenizer
KeywordTokenizer	Tokenizer που παράγει διακριτικά με βάση ολόκληρες συμβολοσειρές	Tokenizer
LetterTokenizer	Tokenizer που διαχωρίζει διακριτικά ανά γράμμα	Tokenizer
LowerCaseTokenizer	Tokenizer που κανονικοποιεί το κείμενο κάνοντας όλους τους χαρακτήρες μικρά γράμματα	Tokenizer

SinkTokenizer	Tokenizer που διαχωρίζει το κείμενο σε διακριτικά με βάση τα προηγούμενα διακριτά που εφτιαξε νοριτερα	Tokenizer
StandardTokenizer	Tokenizer βασισμένος σε μια γραμματική, σύμφωνα με την οποία αναγωνρλιζει πρότυπα ως διακριτικά	Tokenizer
TokenFilter	Αναλυτές που επεξεργάζοντε σύνολα διακριτικών	TokenFilters
LowerCaseFilter	Φίλτρο που κανονικοποιεί τα διακριτικά σε μικρά γράμματα	TokenFilters
StopFilter	Φίλτρο που αφαιρεί απο διακριτικά κοινές λέξεις (βασιζόμενο σε ένα λεξικό)	TokenFilters
PorterStemFilter	Φίλτρο που κρατάει το μικρότερο δυνατό υποσύνολο γραμμάτων από διακριτικά ώστε να εκφράζει περισσότερες έννοιες	TokenFilters
TeeTokenFilter	Φίλτρο που περνάει κάθε διακριτικό απο ενα SinkTokenizer επίπεδο ανάλυσης	TokenFilters
ASCIFoldingFilter	Φίλτρο που εγκαθιστά χαρακτήρες εκτός ASCII αυτών	TokenFilters
CachingTokenFilter	Αποθηκεύει τα διακριτικά	TokenFilters
LengthFilter	Φίλτρο που κρατάει τα διακριτικά εντός ενός ορισμένου εύρους μεγεθών	TokenFilters
StandardFitler	Φιλτρο που αφαιρεί σύμβολα και τονισμούς από διακριτικά	TokenFilters

Πίνακας 5.1 Κυριότεροι έτοιμοι αναλυτές της Lucene.

Φυσικά, όσο πιο πολύπλοκες είναι οι εφαρμογές τόσο πιο πολύπλοκη μπορεί να γίνει και η αλληλουχία αναλυτών σε μια εφαρμογή Lucene. Όσο πιο σύνθετα και προηγμένα είναι τα πρότυπα αναζήτησης τόσο πιο εύστοχη θα πρέπει να είναι η σχεδίαση της ανάλυσης. Σε αρκετές περιπτώσεις, οι κύριοι αναλυτές που παρέχει η Lucene είναι αρκετοί, αλλά από την άλλη, σε εξειδικευμένες περιπτώσεις χρειάζονται ακόμα και αυτοσχέδιοι αναλυτές για να καλύψουν τις ανάγκες του χρήστη. Σε κάθε περίπτωση, η Lucene είναι μια βιβλιοθήκη με μεγάλες δυνατότητες παραμετροποίησης και ελευθερίας στην σχεδίαση της ανάλυσης. [12][15]

Η πολυπλοκότητα της ανάλυσης που θα εφαρμοστεί σε μια εφαρμογή Lucene είναι αντιστρόφως ανάλογη της ταχύτητας του ευρετηρίου. Αυτό σημαίνει ότι μπορεί μια πολύπλοκη ανάλυση να αναγνωρίζει πιο σύνθετες συσχετίσεις λέξεων με έννοιες ή να δίνει πιο ακριβή αποτελέσματα, αλλά είναι και πιο αργή, καθώς περιλαμβάνει περισσότερους υπολογισμούς. Αυτό οδηγεί στον σχεδιαστικό συμβιβασμό μεταξύ πολυπλοκότητας ανάλυσης και ταχύτητας ευρετηρίου, ώστε να βρεθεί μια χρυσή τομή. [12] [13]

Ευρετηρίαση και αναζήτηση

Η Lucene χρησιμοποιεί τα διακριτικά που παράγονται από την ανάλυση για να χτίσει ευρετήρια και να κατατάξει τα έγγραφα σε αυτά. Αυτό βελτιώνει και την απόδοση της αναζήτησης καθώς χωρίς ευρετήρια μια αναζήτηση θα έπρεπε να περάσει μέσα από κάθε έγγραφο. Χάρη στα ευρετήρια η αναζήτηση γίνεται αποδοτικά, κάνοντας μόνο τα απαραίτητα βήματα και λαμβάνει χώρα στο ευρετήριο αντί για τα ίδια τα έγγραφα.

Τα ευρετήρια στη Lucene χτίζονται χρησιμοποιώντας λίστες που καταγράφουν τις θέσεις των διακριτικών στα αρχεία. Τα ευρετήρια αποθηκεύονται, στον σκληρό δίσκο, και κάθε φορά τα απαραίτητα φορτώνονται στην μνήμη RAM. Η Lucene χτίζει ευρετήρια αντεστραμμένης σειράς, και τα χρησιμοποιεί χάρη σε ειδικούς χάρτες που βασίζονται στην θεωρία των γράφων. Ένας χάρτης υπολογίζει την θέση στο ευρετήριο που δείχνει ένας όρος, μέσα από εναλλαγές καταστάσεων, αναλύοντας γράμμα-γράμμα τον όρο αυτό.

Η αναζήτηση λαμβάνει χώρα μέσα στο ευρετήριο, το οποίο θα πρέπει να έχει δημιουργηθεί εκ των προτέρων. Αρχικά, η Lucene βασιζόμενη στο κείμενο αναζήτησης παράγει ένα ερώτημα,

μέσα από ανάλυση και αυτό το ερώτημα καταλήγει στο ευρετήριο. Μέσα από την αναζήτηση του ερωτήματος στο ευρετήριο, ανακτώνται τα κατάλληλα έγγραφα που σχετίζονται τελικά με την αρχική αναζήτηση. [13][14]

Άλλες δυνατότητες

Η Lucene παρέχει και κάποιες επιπλέον δυνατότητες, εκτός των παραπάνω, πολύ χρήσιμες για τον σχεδιασμό μιας μηχανής αναζήτησης. Αρχικά, η αναζήτηση στα έγγραφα μπορεί να περιοριστεί σε συγκεκριμένα πεδία του ευρετηρίου, έτσι ώστε να μεγιστοποιηθεί η συνάφεια των αποτελεσμάτων και να ελαχιστοποιηθεί ο χρόνος αναζήτησης.

Επίσης, η Lucene παρέχει και μια σειρά από δυνατότητες κατάταξης των αποτελεσμάτων μιας αναζήτησης. Πρώτα από όλα, παρέχει την δυνατότητα βαθμονόμησης των αποτελεσμάτων με βάση την συνάφειά τους με τους όρους της αναζήτησης. Έτσι, δίνει την δυνατότητα κατάταξης των αποτελεσμάτων βάση της σχετικότητάς τους, πράγμα πολύ σημαντικό για την παρουσίαση και το φιλτράρισμα των αποτελεσμάτων. Ακόμη, δίνει την δυνατότητα για εφαρμογή επιπλέον ταξινόμησης στα αποτελέσματα, με βάση τα πεδία των εγγράφων (πχ με βάση το έτος, τον τίτλο κλπ).

Τέλος, η Lucene δίνει την δυνατότητα εφαρμογής όλων των απαραίτητων πράξεων συνόλων, όπως ένωση, τομή και ομαδοποίηση. Αυτή η δυνατότητα επιτρέπει στην ανάκτηση πληροφοριών που αφορούν σε ομάδες εγγράφων. Για παράδειγμα μια τέτοιου είδους πληροφορία είναι ο αριθμός των παραγγελιών που έχει κάνει ο κάθε πελάτης. Θεωρώντας ότι κάθε παραγγελία αποθηκεύεται σε ένα αρχείο και αυτό περιλαμβάνει τον κωδικό πελάτη, η παραπάνω αναζήτηση αποτελεί την αναζήτηση παραγγελιών με ομαδοποίηση αυτών ανά πελάτη. [12]

Υποστήριξη γλωσσών προγραμματισμού

Η αρχική έκδοση της Lucene γράφτηκε εξ ολοκλήρου σε Java. Όμως, η βιβλιοθήκη έχει πλέον μεταφερθεί σε διάφορες γλώσσες προγραμματισμού όπως C++, C#, Delphi, Perl, PHP, Python και Ruby. Οι δυνατότητες προηγμένης αναζήτησης που υποστηρίζονται από τη Lucene είναι τελεστές boolean, αναζητήσεις πεδίου αναζητήσεις με χαρακτήρες μπαλαντέρ και αναζητήσεις εύρους . Μερικοί από τους στόχους της Lucene περιλαμβάνουν απλή, καλά τεκμηριωμένη διεπαφή διασύνδεσης (API), υποστήριξη για κοινές λειτουργίες μηχανών αναζήτησης και

επεκτασιμότητας. Ορισμένα έργα έχουν βασιστεί στη Lucene, προσθέτοντας επιπλέον χαρακτηριστικά και επεκτείνοντας τα υπάρχοντα. [12]

5.2 Δυνατότητες και αδυναμίες

Δυνατότητες

Η Lucene αποτελεί μια δωρεάν και ανοιχτού κώδικα βιβλιοθήκη, που μπορεί να δώσει υψηλού επιπέδου λύσεις χωρίς κόστος. Παρότι παρέχεται δωρεάν, είναι μία από τις κορυφαίες διαθέσιμες λύσεις για μηχανές αναζήτησης και μπορεί να καλύψει τις ανάγκες τόσο μικρών όσο και μεγάλων εταιριών που αναπτύσσουν λογισμικό. Επιπλέον, ως λογισμικό ανοιχτού κώδικα, δίνει την δυνατότητα στους χρήστες να γίνουν και ιδιοκτήτες της, αναπτύσσοντάς την, κάνοντας προτάσεις και προσαρμόζοντας το λογισμικό στα μέτρα τους.

Η βιβλιοθήκη, με αυτό τον τρόπο, διατηρεί μια τεράστια κοινότητα που την συντηρεί, την αναπτύσσει και την υποστηρίζει, προσθέτοντας συνεχώς νέες δυνατότητες, και διορθώσεις. Τέλος, αποτελεί μια πολύ δημοφιλή βιβλιοθήκη, με αποτέλεσμα η κοινότητά της να συμβάλει στον διαμοιρασμό σχεδίων, ιδεών και υλοποιήσεων, γεγονός που δίνει στέρεο πάτημα σε έναν νέο σχεδιαστή που χρειάζεται υποστήριξη.

Μελετώντας την Lucene από άποψη απόδοσης, τα χαρακτηριστικά της παρουσιάζονται εξαιρετικά. Η ευρετηρίαση που αποδίδει είναι ταχύτατη, ενώ η ανάλυση ακριβής και η παρουσίαση των αποτελεσμάτων μπορεί να γίνει με πολλούς τρόπους. Η αποτελεσματικότητά της, σε συνδυασμό με την ταχύτητά της την έχει κάνει τόσο δημοφιλή και σίγουρα δεν είναι τυχαίο το γεγονός πως κορυφαίες μηχανές αναζήτησης έχουν βασιστεί επάνω της (όπως Elasticsearch και Solr). Το ακόμη πιο εντυπωσιακό προσόν της είναι η καταπληκτική διαχείριση υπολογιστικών πόρων που παρέχει.

Η μνήμη RAM που απαιτεί είναι ελάχιστη και ο φόρτος που επιβάλει στον επεξεργαστή μικρός. Αυτό οδηγεί στην εξοικονόμηση σημαντικών ποσών που θα απαιτούταν για την λειτουργία εφαρμογών ενώ δίνει χώρο στον σχεδιαστή για την ανάπτυξη πιο πολύπλοκων εφαρμογών. Ακόμη, η Lucene λειτουργεί σε όλα τα δημοφιλή λειτουργικά συστήματα, είναι συμβατή με

όλα τα σύγχρονα συστήματα και παρέχει εύκολη διασύνδεση με web crawlers, ώστε να παρέχει αναζήτηση μέσω διαδικτύου.

Η Lucene παρέχει στον σχεδιαστή πολλές λύσεις, ώστε πάνω τους να χτιστεί μια ισχυρή εφαρμογή. Τα έτοιμα κομμάτια, που παρέχει, είναι εύχρηστα και αποδοτικά, ενώ χρησιμοποιούνται σε αφαιρετικά επίπεδα, ώστε η σχεδίαση να διευκολύνεται. Η βιβλιοθήκη μπορεί να αποδώσει σε πληθώρα εφαρμογών και μάλιστα με πολύ καλά αποτελέσματα.

Η δυνατότητες παραμετροποίησής που παρέχει της δίνουν τεράστιες σχεδιαστικές ελευθερίες και πολύ μεγάλο εύρος εφαρμογών. Για παράδειγμα σε επίπεδο ανάλυσης, οι επιλογές πολλές, και οι αναλυτές αποδοτικοί και παραμετροποιημένοι. Εκτός αυτού, υπάρχει και η δυνατότητα εξ' ολοκλήρου προσαρμογής της εφαρμογής για πιο εξειδικευμένες εφαρμογές.

Σε γενικές γραμμές, η Lucene μπορεί να καλύψει σχεδόν όλο το φάσμα λειτουργιών που απαιτείται να καλύπτει μια μηχανή αναζήτησης, αποδοτικά με σχεδιαστικές ελευθερίες, δωρεάν και με ελάχιστες υπολογιστικές απαιτήσεις. Φυσικά, όλες αυτές οι δυνατότητες της μηχανής αναζήτησης οδηγούν σε κάποιους περιορισμούς και συμβιβασμούς που πρέπει να καλύπτονται. [12][15]

Περιορισμοί

Η Lucene από την μία δίνει βαθμούς ελευθερίας και εφόδια στον σχεδιαστή, αλλά από την άλλη απαιτεί, συνήθως, μια πολύπλοκη σχεδίαση, πολλούς κύκλους δοκιμών, εμπειρία και χρόνο ώστε να αποδώσει.

Η ανάπτυξη μιας εφαρμογής με βάση την βιβλιοθήκη χρειάζεται χρόνο και πόρους, ίσως περισσότερο από κάθε άλλη εναλλακτική μηχανή αναζήτησης που θεωρείται κορυφαία στις μέρες μας.

Η βιβλιοθήκη δεν μπορεί σε καμία περίπτωση να θεωρηθεί ως έτοιμο σύστημα που μπορεί να λειτουργήσει άμεσα σε μια νέα εφαρμογή. Αντ' αυτού, οι δυνατότητες που παρέχει έρχονται σε αντιδιαστολή με την πολυπλοκότητα σχεδίασης που επιφέρει.

Οι δυνατότητες που παρέχει η Lucene σε επίπεδο ανάλυσης κειμένου είναι εμφανές ότι είναι πολλές. Παρόλα αυτά, η πολλές επιλογές, σε συνδυασμό με την πολυπλοκότητά της στην σχεδίαση μπορούν εύκολα να οδηγήσουν σε σχεδιαστικά λάθη, δίνοντας μηχανές αναζήτησης που δεν αποδίδουν ούτε στον χρόνο ούτε στα αποτελέσματα που θα έπρεπε. Ουσιαστικά, οι

δυνατότητες της Lucene είναι τεράστιες, αλλά σε καμία περίπτωση δεν είναι εγγυημένες, ειδικά σε περίπτωση που δεν αξιοποιηθεί σωστά η βιβλιοθήκη.

Η Lucene αντιμετωπίζει δυσκολίες στην κλιμάκωση, σε περίπτωση που ο όγκος των δεδομένων σε μια εφαρμογή ανέβει τάξεις μεγέθους. Παρότι η βιβλιοθήκη δίνει πολύ αποδοτικές λύσεις, οι προσαρμογές που απαιτούνται ύστερα από την αλλαγή του όγκου των δεδομένων σε μια εφαρμογή είναι πολλές και σύνθετες.

Επιπλέον, η εγκατάσταση μιας εφαρμογής βασισμένης σε Lucene σε μια συστάδα υπολογιστών (cluster) στο διαδίκτυο είναι αρκετά δύσκολη και απαιτεί ειδική σχεδίαση. Οι συστάδες χρησιμοποιούνται ευρέως για την κλιμάκωση εφαρμογών, για την διατήρηση αντιγράφων και για την παραλληλοποίηση διαδικασιών, καθιστώντας αυτόν τον περιορισμό την Lucene πολύ σημαντικό.

Κάνοντας τον απολογισμό, η Lucene είναι ένα πολύ ισχυρό εργαλείο, που απαιτεί ωστόσο εμπειρία, πόρους και επιπλέον λύσεις για να αποδώσει. Αποτελεί μια εξαιρετική βιβλιοθήκη, που παρόλα αυτά δεν δίνει έτοιμες λύσεις. Αυτός είναι κι ο λόγος που άλλες μηχανές αναζήτησης που παρέχουν επί πληρωμή υπηρεσίες παραμένουν δημοφιλείς. [12][13][15]

ΚΕΦΑΛΑΙΟ 6

ΠΡΑΚΤΙΚΟ ΚΟΜΜΑΤΙ

6.1 Δημιουργία προγράμματος για Mongodb

Προκειμένου να εξετάσουμε την mongodDB δημιουργήθηκε ένα πρόγραμμα μέσω python και χρησιμοποιήσαμε το MongoDB Compass για να μπορούμε να έχουμε μια άλλη εικόνα του τι γίνεται μέσα στην βάση δεδομένων της mongoDB κατά την διάρκεια των δοκιμών. Πριν προχωρήσουμε στην εκτέλεση του κώδικα θα αναφέρουμε με λίγα λόγια τι κάνει αυτός ο κώδικας. Ο κώδικας δημιουργήθηκε για την μελέτη της mongoDB και αναφέρετε στο πως θα μπορούσε μια εταιρεία να κρατάει αρχείο και πληροφορίες για τους εργαζόμενους της αλλά και να διαχειρίζεται τα αρχεία αυτά. Πιο συγκεκριμένα ο κώδικας συνδέεται με την βάση δεδομένων της mongoDB και μας εμφανίζει έξι επιλογές. Πρώτη επιλογή αυτη της δημιουργίας όπου ο χρήστης θα μπορεί να εισάγει μέσα στην βάση έναν ή πολλούς εργαζόμενους . Επόμενη επιλογή ειναι η αναζήτηση, σ αυτό το σημείο ο χρήστης θα μπορεί να ψάξει μέσα στην βάση δεδομένων για κάποιον εργαζόμενο και να θέσει κάποια κριτήρια αναζήτησης. Έπειτα έχουμε την επιλογή της επεξεργασίας αρχείου, όπου με την χρήση του ID του εργαζομένου μπορεί να κάνει μια τροποποίηση στα στοιχεία του. Μετά έχουμε την επιλογή αυτόματης συμπλήρωσης βάσης με αληθοφανή αρχεία, σε περίπτωση που ο χρήστης θέλει να γεμίσει την βάση με έναν μεγάλο αριθμό εργαζομένων γρήγορα. Τέλος την επιλογή έξοδος για να τερματήσει η λειτουργεία του προγράμματος (όπου κατά την έξοδο το πρόγραμμα αδειάζει την βάση δεδομένων). Ο σχεδιασμός της βάσης έγινε μέσω του MongoDB Compass. Οι υπηρεσίες-λειτουργίες που παρέχει η βάση είναι η αποθήκευση, ανάκτηση των αρχείων καθώς και η διαμόρφωση αυτών. Η δομή των δεδομένων μέσα στην βάση θα γίνεται με βάση το ID που τους δίνει η MongoDB στα έγγραφα και τα έγγραφα θα έχουν την δομή με την οποία θα συμπληρώνουμε κατά την διαδικασία της δημιουργίας στο πρόγραμμα (πχ. ID, όνομα, ηλικία, αριθμό ταυτότητας και μισθός).

Στην συνέχεια θα προχωρήσουμε στην ανάλυση του κώδικα αρχίζοντας από τις βιβλιοθήκες που θα χρειαστούμε για την δημιουργία του κώδικα οι οποίες είναι :

```
from bson import ObjectId
from pymongo import MongoClient
from random import *
import random
import string
import names
```

Μέτα θα δούμε το κομμάτι της συνδεσιμότητας κώδικα με την βάση για να μπορεί το προγραμμά μας να επικοινωνεί με αυτή (το οποίο θα δούμε και παρακάτω κατα την διαδικασία της εκτέλεσης) :

```
cluster = MongoClient("mongodb://localhost:27017/employees")
db = cluster["employees"]
collection = db["employees"]
```

Έπειτα έχουμε το κομμάτι που ανατρέχει όταν ο χρήστης κάνει δημιουργία κάποιου εργαζόμενου :

```
def create_employeei ():
    ask1 = input("Insert a name:\ ).title()
    ask2 = int(input("Insert an age:\t"))
    ask3 = input("Insert ID Number:\t").upper()
    ask4 = int(input("Insert Salary:\t"))
    post = {"name": ask1, "age": ask2, "ID Number": ask3, "Salary": ask4}
    collection.insert_one(post)
    print("\nRegistration Success!\n")
```

Έδω έχουμε την δημιουργία τυχαίων ID για την εκτέλεση της επιλογή της αυτόματης δημιουργίας :

```
c (size=6, chars=string.ascii_uppercase + string.digits):
return ''.join(random.choice(chars) for _ n range(size))
```

Στην συνέχεια έχουμε την συνάρτηση που τρέχει όταν η εφαρμογή λάβει την επιλογή αυτόματης δημιουργίας :

```
def auto_create_employeei ():
    askI = names.get_full_name()
    ask2 = randrange(20, 65)
    ask3 = id_generator()
    ask4 = randrange(400, 2000)
    post = {"name": askI, "age": ask2, "ID Number": ask3, "Salary": ask4}
    collection.insert_one(post)
```

Σε αυτό το σημείο βλέπουμε τη συνάρτηση που τρέχει το πρόγραμμα όταν ο χρήστης επιλέξει να αναζήτησει κάποιον ή κάποιους εργαζόμενους :

```
def search_employee ():
    ask5 = input("Do you like to search base on name,age,ID,income or the whole company?\t").lower()
    if ask5 == "name":
        ask6 = input("Insert a name:\t")
```

```
    ask6_1 = ask6.title()
    for x in collection.find({ 'name": ask6_1}):
        print(x)
elif ask5 == "age":
    ask6 = int(input("Insert age:\t"))
    for x in collection.find({"age": ask6}):
        print(x)

elif ask5 == "income":
    ask6 = int(input("Insert income:\t"))
    for x in collection.find({"Salary": ask6}):
        print(x)

elif ask5 == "id":
    ask6 = input("Insert id:\t")
    ask6_1 = ask6.upper()
    for x in collection.find({ 'ID Number": ask6_1}):
        print(x)

elif ask5 == "whole company":
    for x in collection.find():
        print(x)

else:
    pass
```

Έδώ συναντάμε τη συνάρτηση που εκτελεί το πρόγραμμα όταν πρόκειτε να διαγράψουμε κάποιον εργαζόμενο :

```
                oyee ():
    ask8 = input("Insert the id you want to be deleted:\t").title()
    collection.delete_one({'_id": ObjectId(ask8)})
    print("Deleted!")
```

Τελευταίο από τις επιλογές βλέπουμε την συνάρτηση της επεξεργασίας που τρέχει το πρόγραμμα :

```
def edit_employee():
    ask7 = input("Insert the object_id of the person you want to edit:\t ).lower()
    ask7_2 = input( 'What you want to change? \t ")
    ask7_3 = input( 'To what? \t  )
    doc = collection.find_one_and_update(
        { _id": ObjectId(ask7)}, { $set": {ask7_2: ask7_3}}, upsert=True)
    print("Edit Done!")
```

Και τέλος βλέπουμε το κύριο σημείο του προγράμματος όπου τρέχει μόλις αρχίσουμε να εκτελούμε την εφαρμογή:

```
print("Welcome to MongoDB Test:\n")
menu_active = True
while menu_active:
    print("1: Create\n"
```

```
      "2: Search\n"
      "3: Edit\n"
      "4: Delete\n"
      "5: Auto-fill db\n"
      "6: Exit")
option = int(input("Choose an option:\t "))
if option == 1:
    ask = input("You want to create one or more element?\t").lower()
    if ask == "one":
        create_employees()
    elif ask == "more":
        ask4 = int(input("How many?\t"))
        for somany in range(0, ask4):
            create_employees()
    else:
        pass
elif option == 2:
    search_employee()
elif option == 3:
    edit_employee()
elif option == 4:
    delete_employee()
elif option == 5:
    ask10 = int(input("How many employees should I insert?\t"))
    for employ in range(0, ask10):
        auto_create_employees()
```

```
elif option == 6:
    askl 1 = input("If you exit the db will get deleted are you sure?\t").lower()
    if ask11 == "yes":
        collection.delete_many({})
        print("GoodBye!")
        break
    else:
        pass
elif option == 5:
    ask10 = int(input("How many employees should I insert?\t"))
    for employ in range(0, ask10):
        auto_create_employees()
```

Το μένου της εφαρμογής θα έχει την εξής μορφή:

Welcome ζο [longoDB Test:

1| Create
2: Search
3: Edit
4: Delete
5: Auto*fill· db
ü: Exit

Εικόνα 6.1 κώδικας python για την χρήση Mongodb -options.

6.1.1. Συνδεσιμότητα

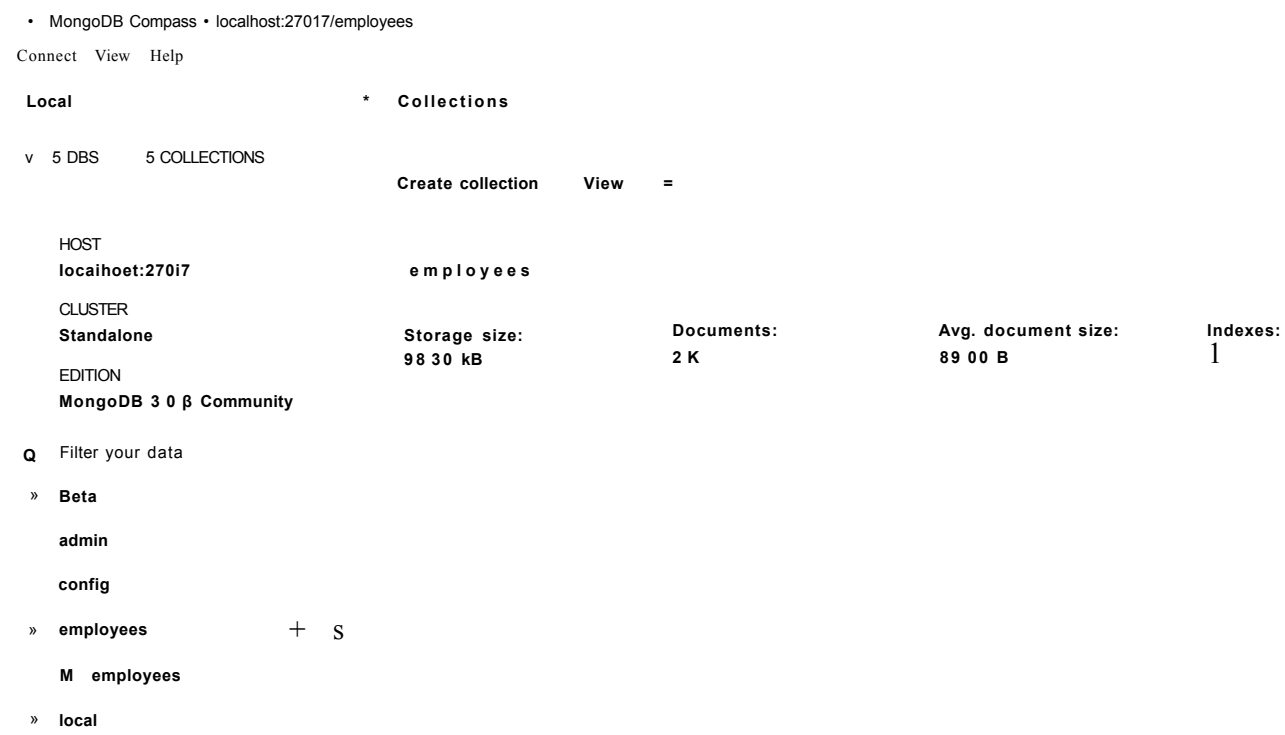
```
cluster = MongoClient("mongodb://localhost:27017/employees")

db = cluster["employees"]

collection = db["employees"]
```

Σε αυτό το σημείο ο κώδικας παίρνει την ip που βρίσκεται η βάση το όνομά της και το όνομα της συλλογής που θα εισάγει τα αρχεία.

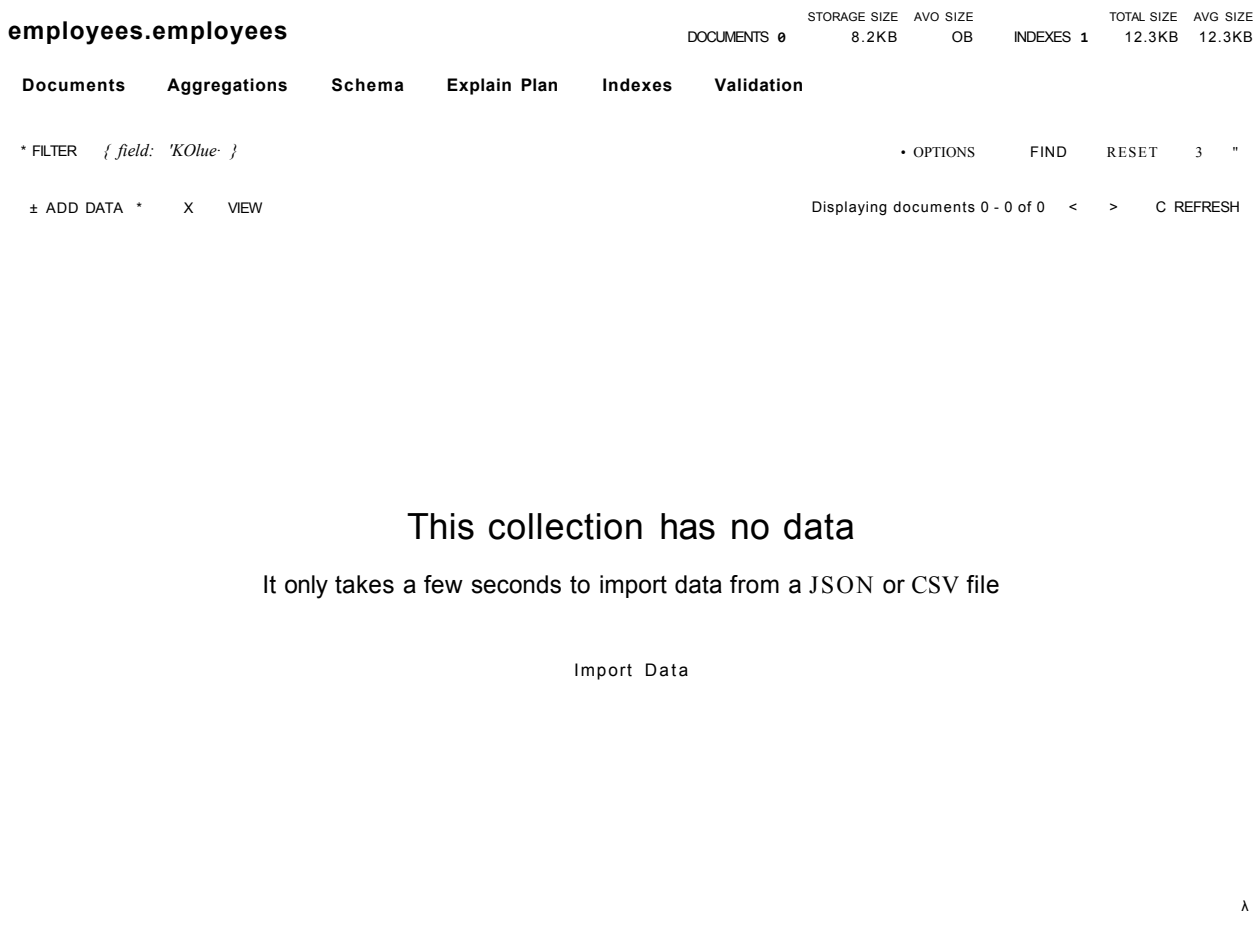
Όπως βλέπουμε και από το mongoDB Compass έχει συνδεθεί σε αυτά τα σημεία.



Εικόνα 6.2 κώδικας python για την χρήση Mongodb

6.1.2. Δημιουργία

Όπως παρατηρούμε, μέσω του εργαλείου Compass, η βάση είναι άδεια την στιγμή που ανοίγουμε και τρέχουμε το πρόγραμμα.



λ

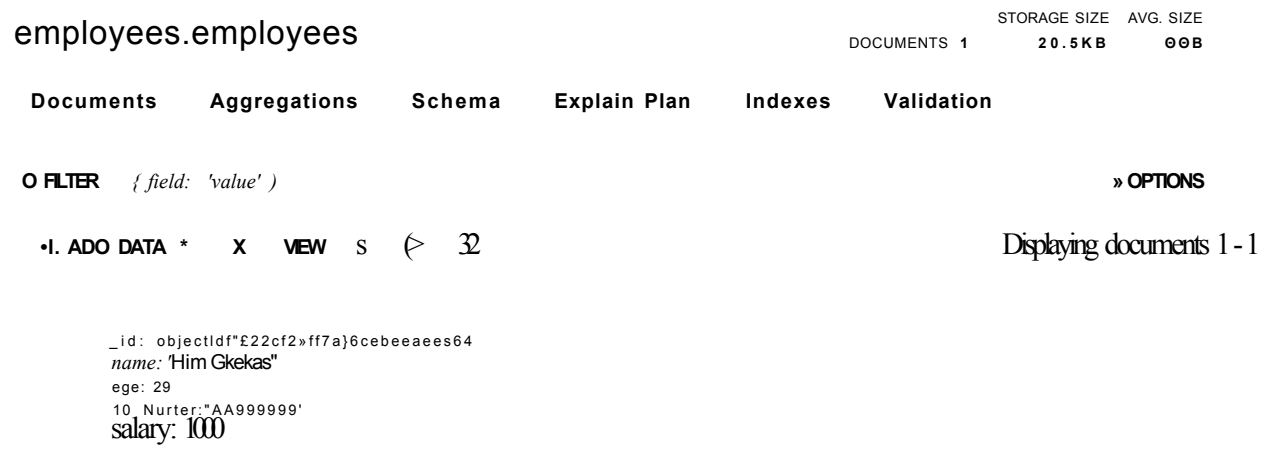
Εικόνα 6.3 κώδικας python για την χρήση mongodb -Create.

Διαλέγοντας την επιλογή Create το πρόγραμμα μας ρωτάει πόσα άτομα θα εισχωρήσουμε. Αφού επιλέξουμε εισάγουμε τα δεδομένα και όταν η διαδικασία ολοκληρωθεί μας ενημερώνει με μήνυμα οτι η εγγραφή πέτυχε.

```
Choose an option:
You want to create one or «ore element?
Insert a name:
Insert an age:
Insert ID Number:
Insert Salary:

Registration Success!
```

Εικόνα 6.4 κώδικας python για την χρήση mongodb -Create.



Εικόνα 6.5 κώδικας python για την χρήση mongodb -Create

Επιβεβαιώνουμε και από το Compass οτι η εγγραφή όντως πραγματοποιήθηκε. Ένα ενδιαφέρον και αξιοσημείωτο πράγμα που πρέπει να σημειωθεί είναι οτι η mongoDB έχει δώσει ένα _id το οποίο είναι μοναδικό μέσα στην βάση (δηλαδή δεν υπάρχει ούτε θα υπάρξει κανένα ίδιο μ αυτό) και θα είναι πολύ χρήσιμο για την χρήση των επόμενων επιλογών του προγράμματος.

Σε αυτό το σημείο της εργασίας θα αναφερθεί και η πέμπτη επιλογή του προγράμματος που είναι το auto-fill για να καλύψουμε όλες της επιλογές της δημιουργίας-Create. Το πρόγραμμα μας δίνει την επιλογή να γεμίσει αυτό την βάση με δικά του τυχαία αληθοφανή αρχεία.

Choose an option:
How many employees should I insert?

Εικόνα 6.6 κώδικας python για την χρήση mongodb -Create

Εικόνα 6.7 κώδικας python για την χρήση mongodb -Create.

Επιλέχθηκε να δημιουργηθούν 2000 αρχεία και όπως φαίνεται τώρα η βάση έχει 2001 αρχεία και το καθένα με δικό του διαφορετικό _id που είναι πολύ σημαντικό στην διαχείριση της βάσης όπως προαναφέραμε.

6.1.3. Αναζήτηση

Επόμενη επιλογή είναι η αναζήτηση των αρχείων μέσα στην βάση. Θα ερωτηθεί απο το πρόγραμμα με ποιο κριτήριο πρέπει να γίνει η αναζήτηση.

```
Choose an option:
Do you like to search base on name,age,10,income or the whole company'
Insert age:
{'.id·: Objectld(1622cf758f7a36ceb00a8e565'), 'name': 'Charles Tucker', 'age': 50, ΄I0 Number': 'VB6NGP', 'Salary': 1299}
{'_id': Objectldi'622cf758f7a36ceb88a8e589'), 'name': 'Jodi Lazewski', 'age': 50, 'ID Number': 'M9B8T0', 'Salary': 779}
{'.id': 0bjectld('622cf751f7a36ceb00a0e5ca·), 'name': 'Lori Flowers', 'age': 50, 'ID Number': '91PNZP', 'Salary': 1839}
{'_id': 0bjectld('622cf75lf7a36ceb08a0e5f5'), 'name': 'Mary Clark·, 'age': 50, ΄I0 Nuaber': 'ACqH7X; 'Salary·: 1626}
{'.id': Objectldi'622cf751f7a36ceb00a0e615'), 'name': 'Justin Zadow', 'age': 50, 'ID Number': 'A14N0U', 'Salary': 548}
{'_id': ObjectId('622cf751f7a36ceb00aOe62f'), 'name': 'Nina Pa»', 'age': 58, 'ID Number': 'UKFW»', 'Salary': 650}
{'.id': Objectld('622cf752f7a36cebO0aOe659[1]'), 'name': 'Yuk Cordova', 'age': 58, 'IDNunber': 'U301EU', 'Salary': 642}
{'.id': Objectld('622cf752f7a36ceb00aOe663[1]'), 'name': 'Kenneth Cushnan', 'age': 58, 'ID Nuaber': '590UWI', 'Salary': 1184}
{'_id': ObjectId('622cf752f7a36ceb00aOe609'), 'name': 'Eunice Guyton', 'age': 5Θ, 'ID Number': 'HL0NK2', 'Salary': 1799}
```

Εικόνα 6.8 κώδικας python για την χρήση mongodb - Search.

Εμφανίζονται όλοι οι εργαζόμενοι που έχουν την ηλικία που ζητήθηκε (στο παράδειγμα είναι 50). Η αναζήτηση μπορεί να γίνει με κάθε δυνατό κριτήριο όπου και μπορεί να ευρεθεί το _id του κάθε εργαζομένου ξεχωριστά.

6.1.4. Επεξεργασία

Επόμενη επιλογή είναι η επεξεργασία κάποιου αρχείου. Έχοντας αναζητήσει το αρχείο που πρέπει να επεξεργαστεί ανακτάται το _id από τον χρήστη και το εισάγει στην επιλογή 3 όταν ζητηθεί. Εδώ φαίνεται η χρησιμότητα της mongoDB που είχε εισάγει τα μοναδικά _id. Στην συνέχεια ρωτάτε από τον χρήστη ποιο στοιχείο θέλει να επεξεργαστεί και μετά να εισάγει το νέο στοιχείο.

```
•{,-jq,: OPiecfiqC.MSCtiBOtAeaecepoceosBPa.)' ,uaue,: .cpekfes incKSL,[1] ,ββε.: so' , id MnupeL; ,AB9Heb' ,2β/βι,λ,: jsw}

Choose an option:
Insert the object_id of the person you want to edit:
What you want to change'
To what?
Edit Done!
```

Choose an option:
Do you like to search base on name,age, ID,income or the whole company?
Insert a name:
{•.id': Objectldf022cf758f7a36ceb80a0e505'), 'name': 'Charles Tucker', 'age': '3D', 'ID Number': VB6NGP', 'Salary': 1299}

Εικόνα 6.9 κώδικας python για την χρήση mongodb - Edit.

Εικόνα 6.10 κώδικας python για την χρήση mongodb - Edit.

Ελέγχοντας και από το Compass η επεξεργασία του αρχείου είναι επιτυχής.

6.1.5. Διαγραφή

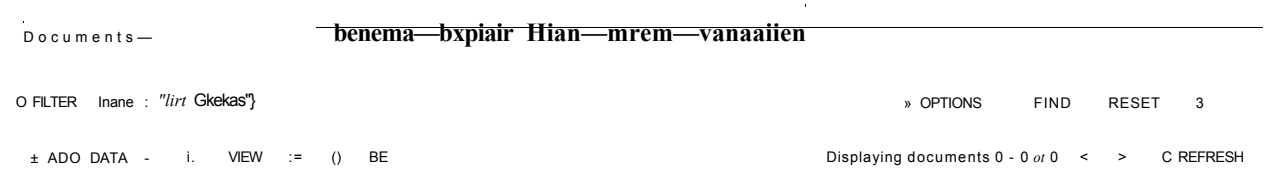
Επόμενη επιλογή στο πρόγραμμα είναι αυτή της διαγραφής. Το πρόγραμμα ζητάει από τον χρήστη το _id του αρχείου που θέλει να διαγράψει και το διαγράφει.

-('.id': OD}ecdfK022dB171f7a36ceb0Baeed55'), Tame": 'Jim GKekas', 'age[1]: 29, 'ID MmQ#: 'AA999999, 'Salary': 16M>

```
Choose an option:
insert trie id you want to ue ueieted:
PEleted^
Do you like to search base on name,age,ID,income or the whole company?
Insert a name:
1: Create
2: Search
3: Edit
4: Delete
5: Auto-fill db
6: Exit
Choose an option:
```

Εικόνα 6.11 κώδικας python για την χρήση mongodb - Delete.

Αφού διαγραφεί το αρχείο και ο χρήστης το αναζητήσει το πρόγραμμα δεν θα το βρει και θα τον οδηγήσει στο αρχικό μενού.

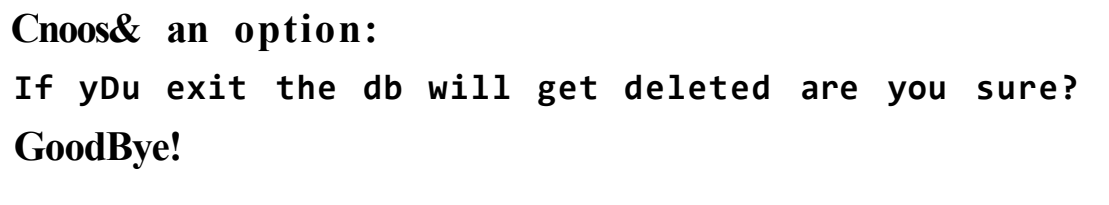


Εικόνα 6.12 κώδικας python για την χρήση mongodb - Delete.

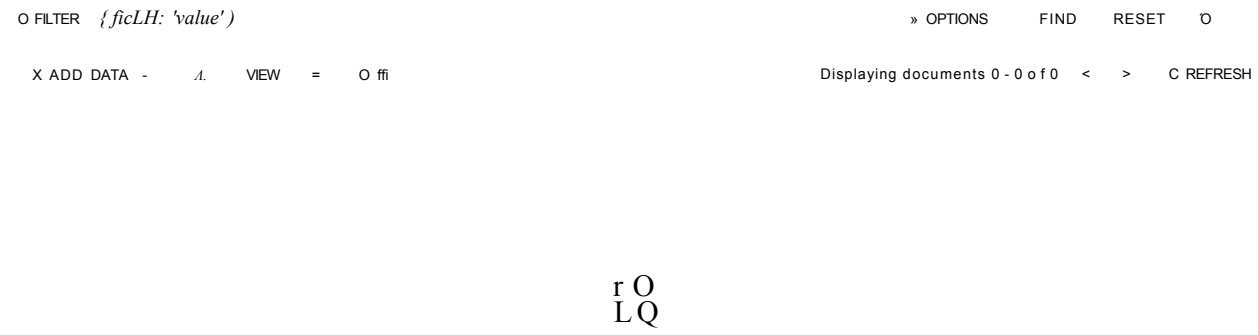
Γίνεται έλεγχος μέσο Compass και το αρχείο όντως έχει διαγραφεί.

6.1.6. Έξοδος

Τελευταία επιλογή στο πρόγραμμα είναι η έξοδος. Όταν ο χρήστης θελήσει να σταματήσει να χρησιμοποιεί την βάση επιλέγει αυτή την επιλογή και το πρόγραμμα τον προειδοποιεί ότι αν σταματήσει η βάση να αδειάσει, ειδάλλως αν πατήσεις κάτι άλλο θα τον γυρίσει στην επεξεργασία της βάσης.



Εικόνα 6.13 κώδικας python για την χρήση mongodb - Exit.



Εικόνα 6.14 κώδικας python για την χρήση mongodb - Exit.

Το πρόγραμμα τερματίστηκε και η βάση άδειασε.

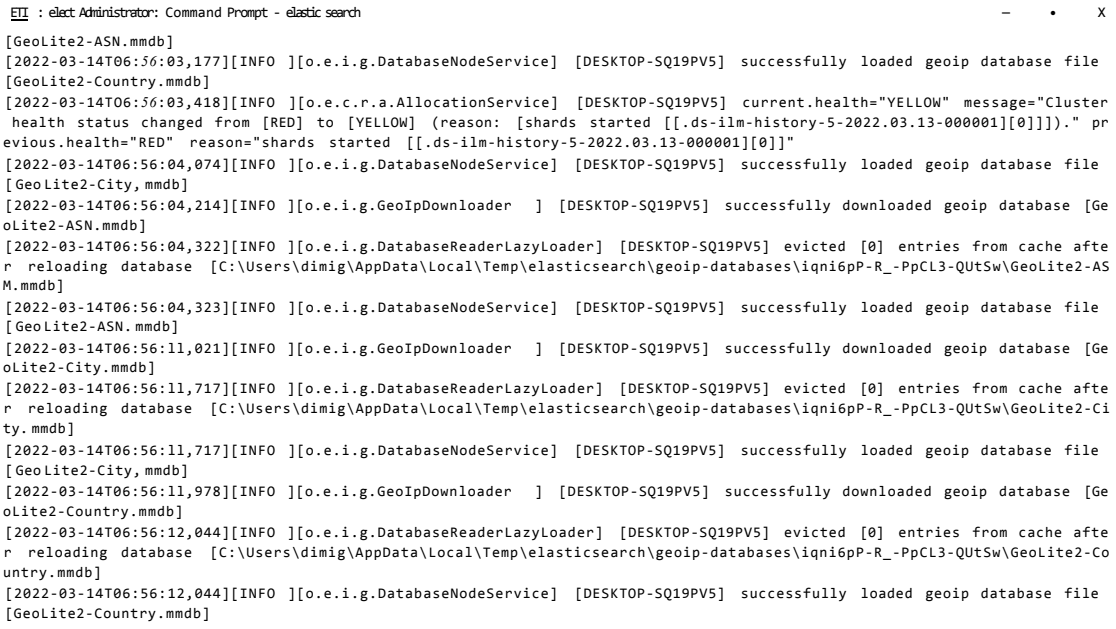
6.1.7. Διαπιστώσεις

Μερικές διαπιστώσεις αφότου δημιουργήθηκε το πρόγραμμα και έγινε επεξεργασία της βάσης. Η εγκατάσταση της mongoDB είναι αρκετά εύκολη σε σημείο που και ένα άτομο με μικρή εμπειρία πάνω στον τομέα με ένα απλό βίντεο καθοδήγησης ή την καθοδήγηση μέσω της σελίδας της monogoDB μπορεί να την εγκαταστήσει. Επίσης η βιβλιοθήκη για να μπορεί να λειτουργήση η python με τον mongoDB έχει πολλές και ποικίλες επιλογές εκτός τον βασικών CRUD που χρησιμοποιήσαμε στο πρόγραμμα. Η διαδικασία και το κομμάτι κώδικα που αφορά την συνδεσιμότητα ήταν πολύ απλό και εύκολο να βρεθεί. Η ταχύτητα ολόκληρης της διαδικασίας εγκατάστασης μέχρι και χρήσης της mongoDB είναι εξαιρετικά γρήγορη, χωρίς καθόλου χάσιμο χρόνου. Ο τρόπος που η βάση δέχεται και επεξεργάζεται τα αρχεία είναι πολύ γρήγορος εξίσου. Επιπρόσθετα η mongoDB παρέχει ένα πολύ ενδιαφέρον εργαλείο ,το mongoDB Compass που είναι πολύ χρήσιμο διότι επιτρέπει την επεξεργασία της βάσης και δίνει επιπλέον πληροφορίες όσο αφορά τα έγγραφα της βάσης άλλα και την ανάλυση τους, επιτρέπει επίσης την διαχείριση της βάσης μέσο την χρήση κώδικα.

6.2 Χρήση Elastic Search kai Lucene

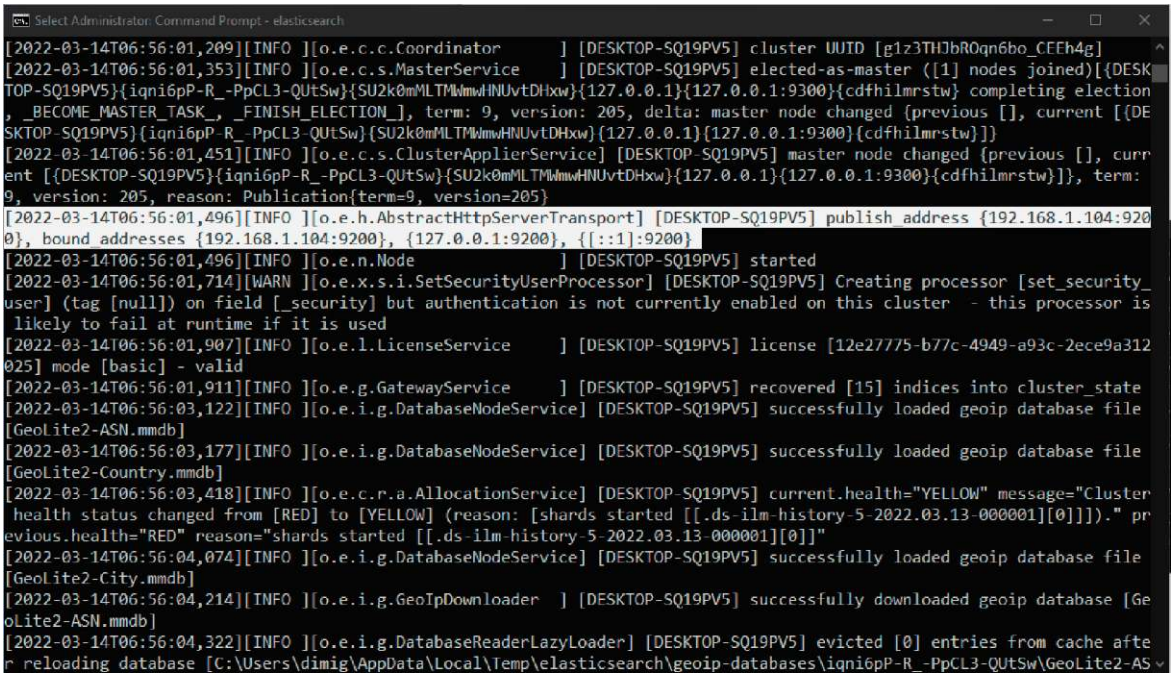
Για τη δοκιμή της Elastic Search και Lucene θα γίνει χρήση του kibana ,ενός εργαλείου της Elastic για την διαχείρηση δεδομένων.

Πρώτο βήμα είναι η εγκατάσταση και εκτέλεση της Elastic Search:



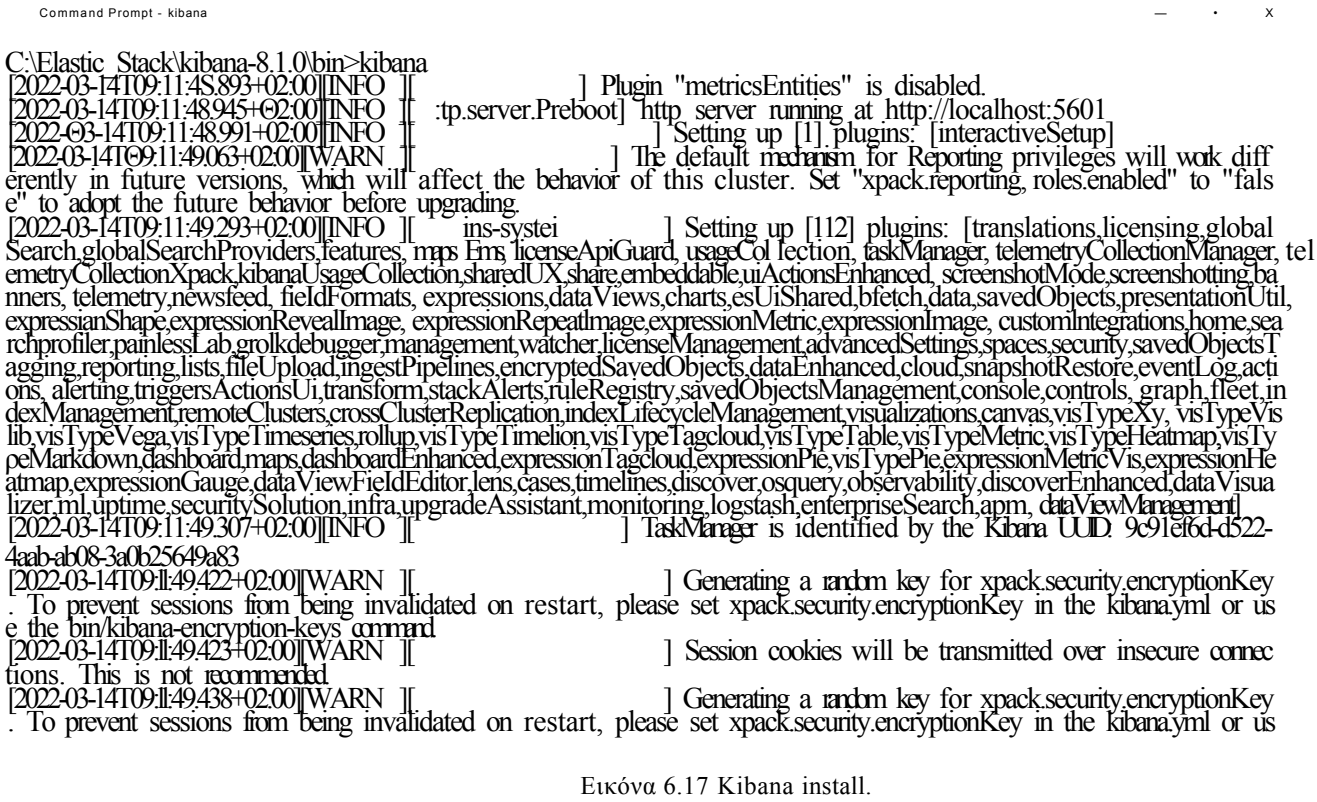
Εικόνα 6.15 Elastic Search install.

Όπως βλέπουμε ο sever της Elastic Search τρέχει σε localhost στο port 9200.



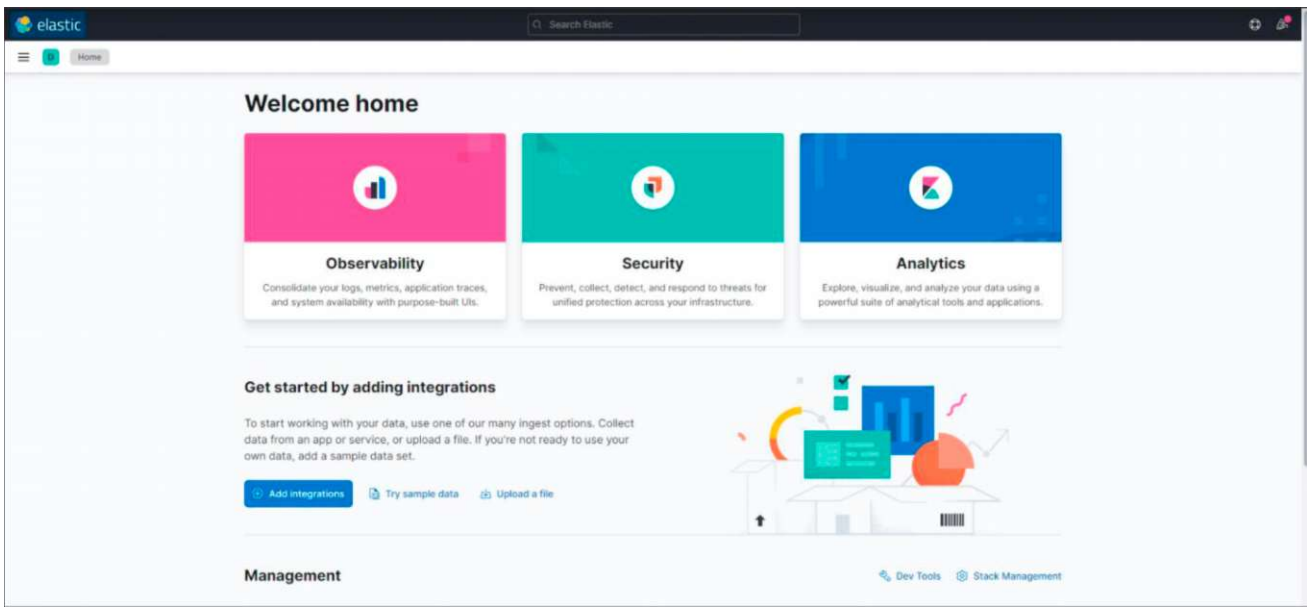
Εικόνα 6.16 Elastic Search install.

Εγκαθιστούμε και εκτελούμε το kibana το οποίο τρέχει στο http://localhost:5601



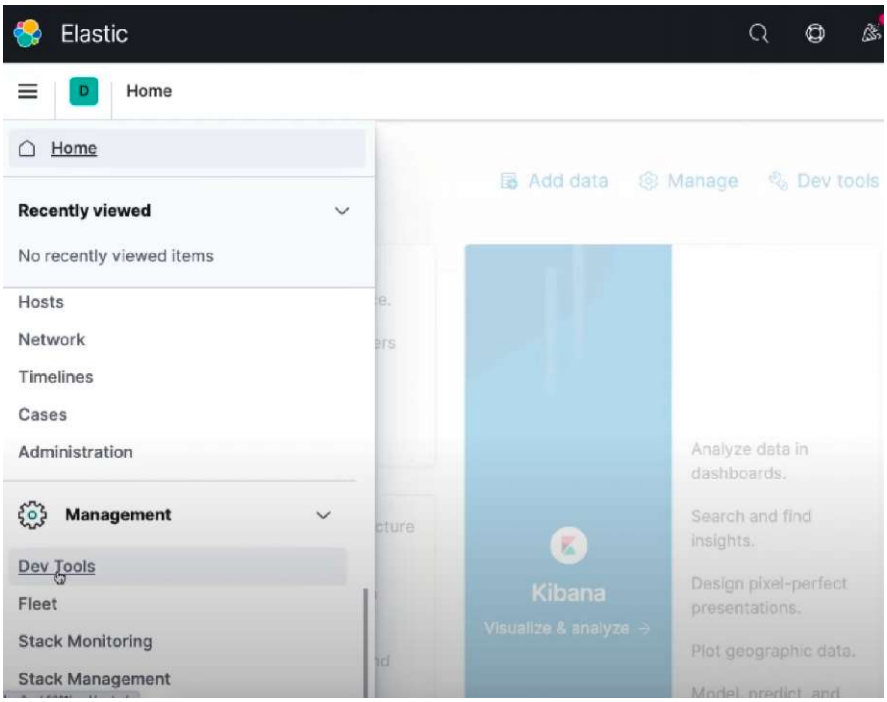
Εικόνα 6.17 Kibana install.

Ανοίγουμε ένα πρόγραμμα περιήγησης στο διαδίκτυο και χρησιμοποιούμε τον σύνδεσμο αυτό και με επιτυχία έχουμε εκτελέσει το kibana και μπορούμε να το χρησιμοποιήσουμε.



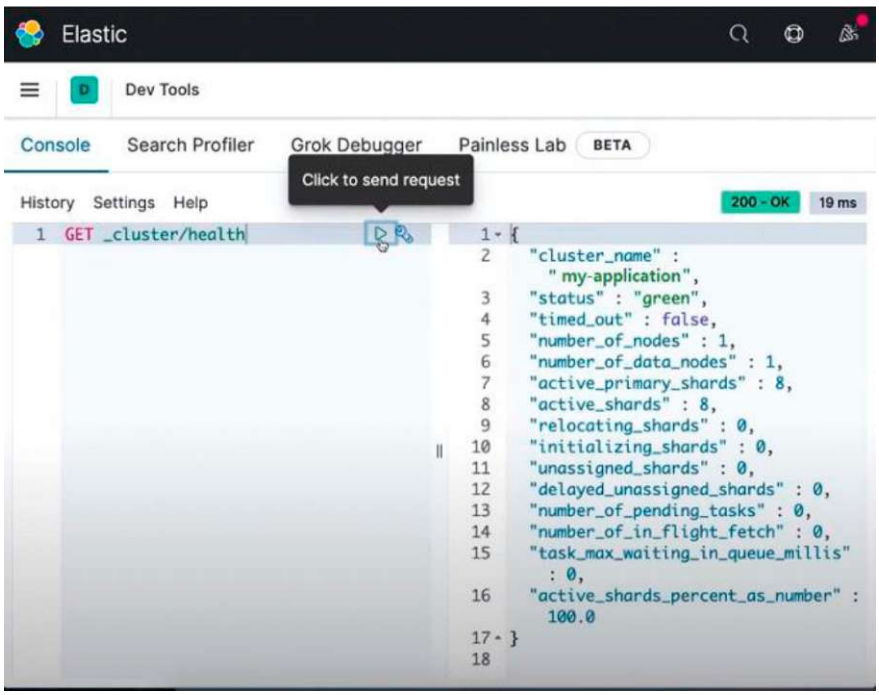
Εικόνα 6.18 kibana run.

Έπειτα ανοίγουμε το μενού και πηγαίνουμε στο dev tools όπου και θα τρέξουμε κάποιες εντολές για να δούμε την αλληλεπίδραση kibana με Elastic search δηλαδή πώς απαντάει το Elasticsearch στα request του kibana.



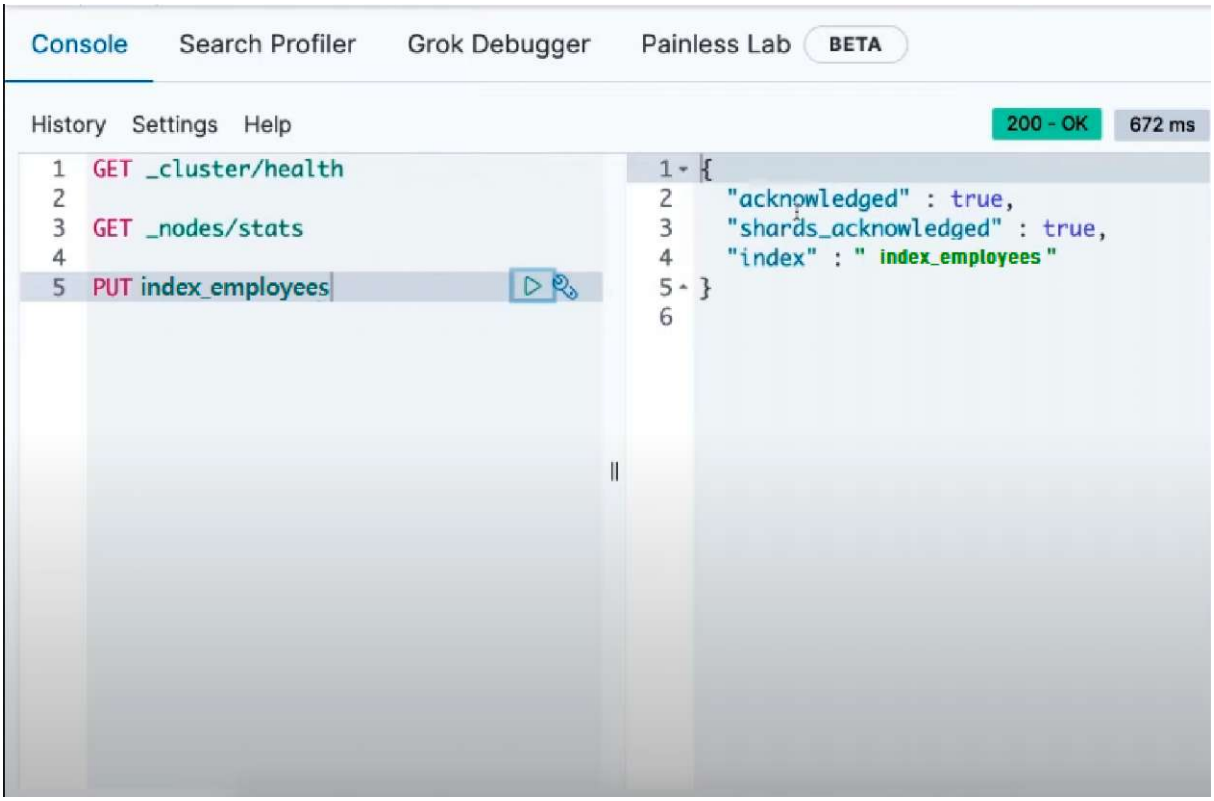
Εικόνα 6.19 Kibana run.

ϊεκινόντας με κάποιες δοκιμές όπως η εντολή GET _cluster/health για να πάρουμε πληροφορίες για την κατάσταση που βρίσκετε ο server .



Εικόνα 6.18 kibana run.

Επόμενο βήμα είναι να δοκιμάσουμε να δημιουργήσουμε ένα index με την εντολή PUT.



Εικόνα 6.21 Kibana run.

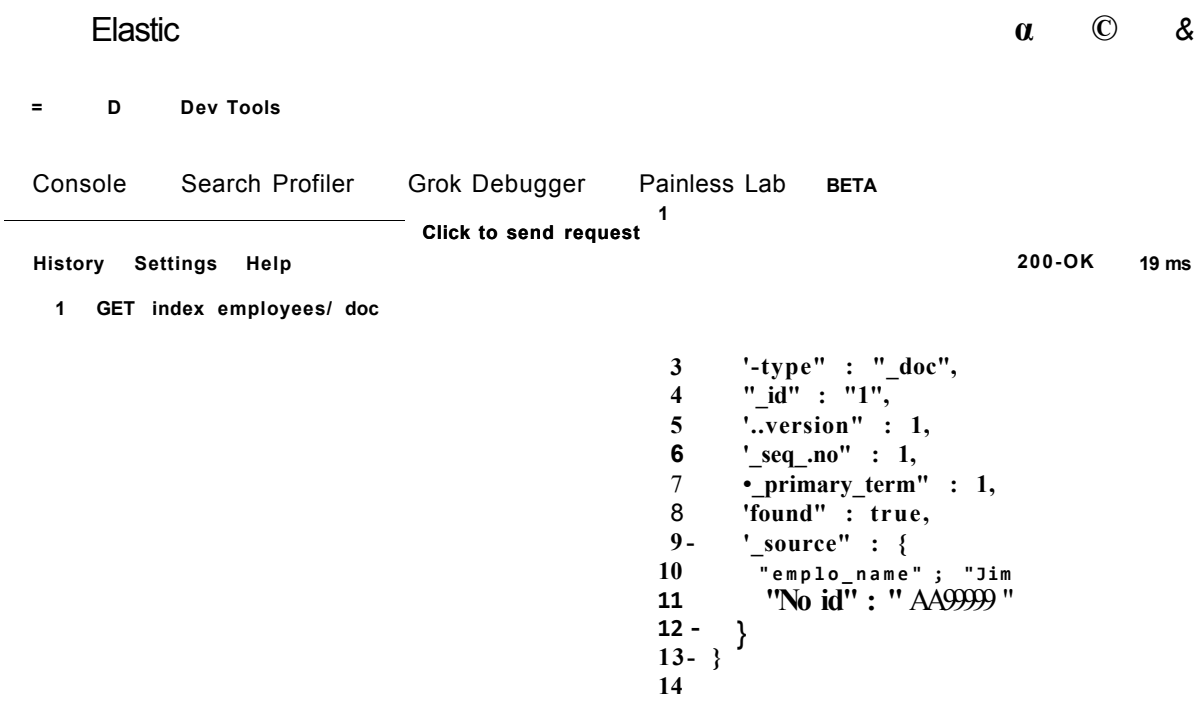
Βλέπουμε οτι το acknowledged είναι true οπότε το index έγινε επιτύχως.Στην συνέχεια θα είσαγουμε ένα αρχείο και η elastic search θα του δώσει ένα μονάδικό id ,θα χρησιμοποιήσουμε την εντολή POST.

```
1  GET .cluster/health
2
3  GET .nodes/stats
4
5  PUT  index employees
6
7  POST lndex_employees/_doc
8- {
9    "emplo_name": "Jim",
10   "No ID": "AA99999"
11- }
```

```
1- {
2    "_index"  :  "indexemployees ",
3    "_type" : "_doc" 1
4    "_id" : "KkjCcnYBTXx5H9HAmlAB
5    ".version" : 1,
6    "result" : "created",
7-   ".shords" : {
8      "total" : 2,
9      "successful" : 1.
10     "foiled" : 0
11-  },
12   "_seq_no" : 0,
13   "_primary_term"  : 1
14- }
15
```

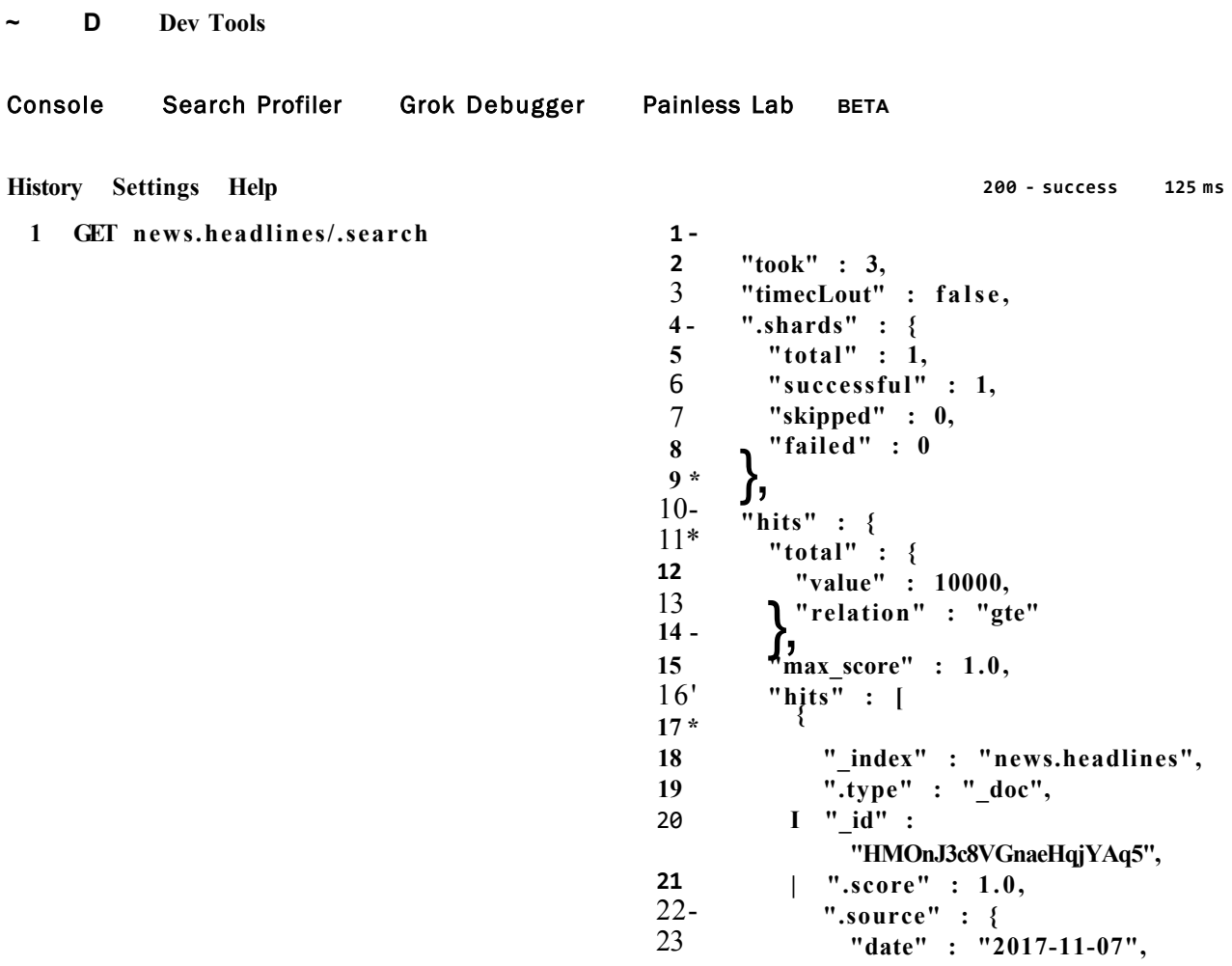
Εικόνα 6.18 kibana run.

Παρατειρήθηκε οτι δημιουργήθηκε ένα αρχείο με τα στοιχεία που είσαγαμε και ένα μοναδικό _id που έδωσε η elastic search.Επίσης μπορούμε να εκτελέσουμε την εντολή GET για να πάρουμε πληροφορίες για ένα αρχείο που έχουμε εισάγει.



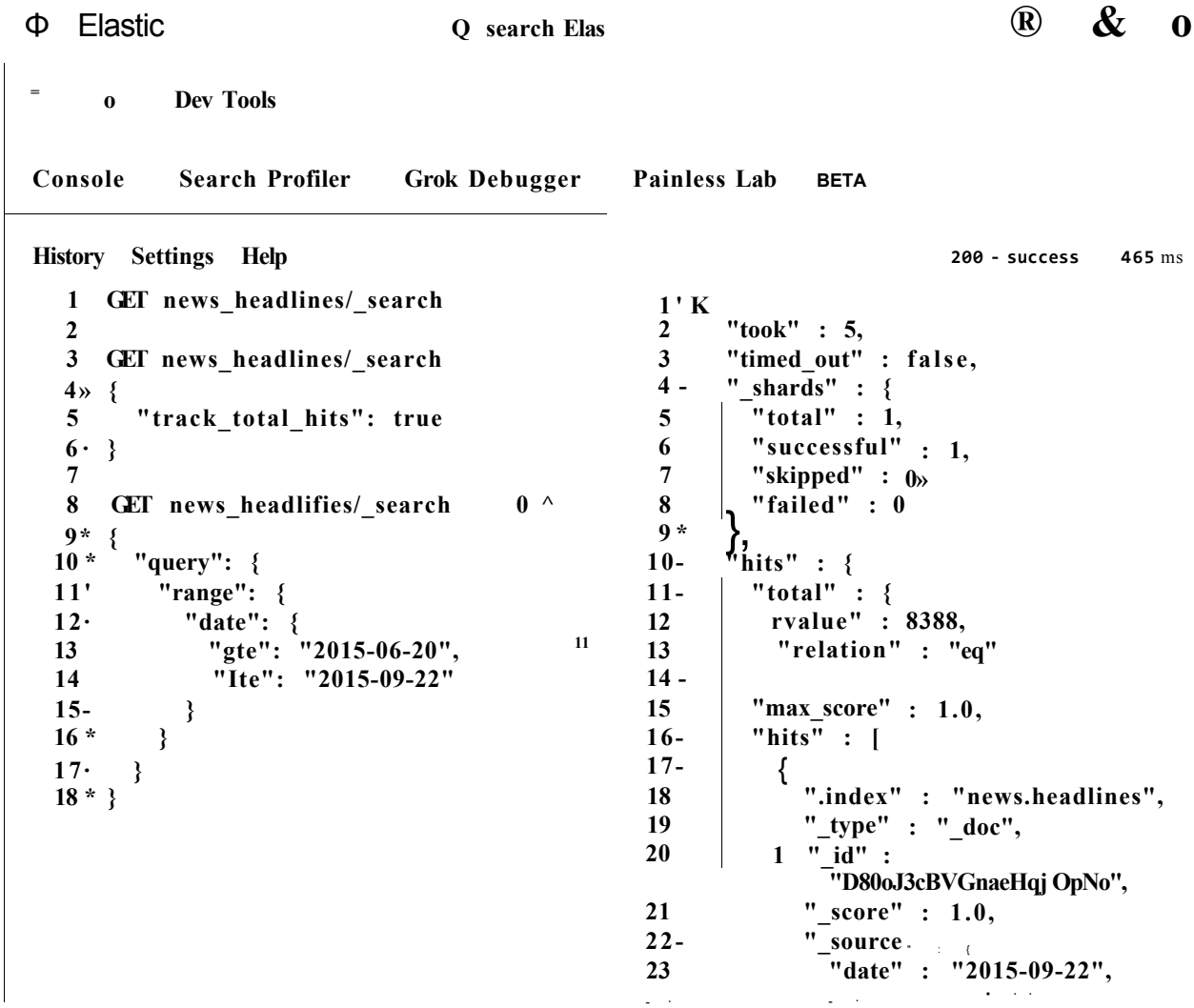
Εικόνα 6.23 kibana run.

Μπορούμε να κάνουμε Post για να κάνουμε edit κάποιο αρχείο και DELETE για να το σβήσουμε. Στη συνέχεια θα δούμε πιο προχωρημένη χρήση του elastic search προσθέτοντας δεδομένα στο kibana και αναζητόντας τα και μπορούμε να δούμε όλα τα περιεχόμενα που έχει κάθε αρχείο.



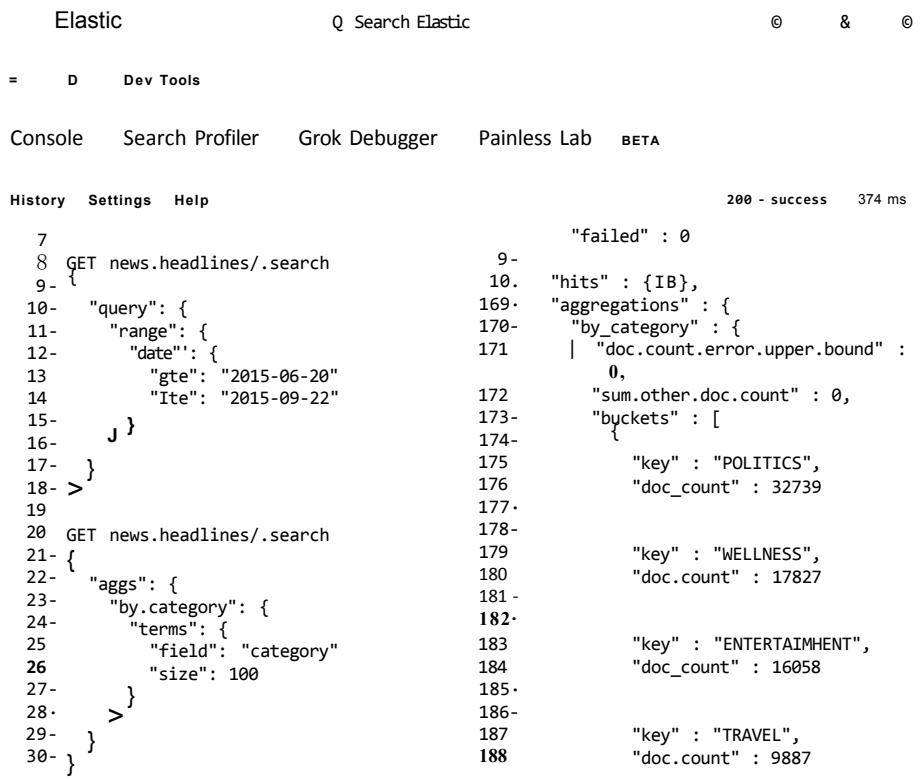
Εικόνα 6.24 kibana run.

Μπορούμε να κάνουμε αναζήτηση αρχείων με τα κριτήρια που εμείς θέλουμε μέσα σε ένα πολύ μεγάλο αριθμό δεδομένων.



Εικόνα 6.25 kibana run.

Οπώς βλέπουμε μας εμφανίζει τα αποτελέσματα από την αναζήτηση που κάναμε. Τέλος αλλο ένα παράδειγμα αναζήτησης , μπορούμε να αναζητήσουμε τις κατηγορίες στις οποίες είναι χωρισμένα τα δεδομένα. Καθώς και να αναζητήσουμε κατηγορίες οι οποίες πλήρουν κάποια κριτήρια.



Εικόνα 6.26 kibana run.

Εικόνα 6.18 kibana run.

6.2.1 Διαπιστώσεις

Τέλος θα αναφερθούμε σε κάποιες διαπιστώσεις για τη χρήση των elasticsearch και lucene. Η εγκατάσταση του elasticsearch και η ρύθμιση για την σύνδεση του με το kibana απετούσε ένα βαθμό εξοικείωσης με εντολές του τερματικού του ανάλογου λειτουργικού συστήματος που χρησιμοποιήσει ο κάθε χρήστης, καθώς και μεγάλη αναζήτηση για κάποιο οδηγό διότι το πρόγραμμα εμφάνιζε σφάλματα ανα περιόδους, αυτό μας οδηγεί στο συμπέρασμα οτι δεν είναι τόσο εύκολη η εγκατάστασή του. Όταν ο χρήστης φτάνει στο σημείο να χρησιμοποιήσει αύτη τη μηχανή αναζήτησης βλέπει οτι είναι πολύ γρήγορη παρόλο τον ογκό των δεδομένων που της έχουμε φορτώσει, σε αυτό ένα μεγάλο μερίδιο ευθύνης το πηγαίνει στη βιβλιοθήκη lucene λόγο του ότι η κατασκευή αυτής της μηχανής είναι βασισμένη απάνω της. Σε συνδυασμό και με άλλα εργαλεία της elastic (παράδηγμα το kibana που χρησιμοποιήσαμε αλλά και άλλα όπως το logstash που είναι ένα εξαίρετο API) μπορούμε να έχουμε μια πολύ γρήγορη και άνετη διαδικασία αναζήτησης με μια πληθώρα αποτελεσμάτων και πληροφοριών απάνω σε αυτά

ΚΕΦΑΛΑΙΟ 7

ΣΥΓΚΡΙΣΕΙΣ-ΣΥΜΠΕΡΑΣΜΑΤΑ

7.1 Βασικά χαρακτηριστικά, στόχοι, άδειες και δυνατά σημεία

Η MongoDB έχει αναπτυχθεί σε C++, ενώ οι Elasticsearch και Lucene σε JAVA. Η MongoDB είναι μια βάση δεδομένων NoSQL που αποθηκεύει τα δεδομένα μιας εφαρμογής σε έγγραφα. Μπορεί να αποτελέσει το βασικό σύστημα αποθήκευσης δεδομένων μια εφαρμογής. Η Elasticsearch, είναι μια μηχανή αναζήτησης που υποστηρίζει αναζήτηση και ανάκτηση εγγράφων που σχετίζονται με την προς αναζήτηση πληροφορία. Συχνά χρησιμοποιείται μαζί με SQL ή NoSQL βάσεις δεδομένων για να παρέχει αναζήτηση πλήρους κειμένου ή και επιπλέον ανάλυση των δεδομένων που αποθηκεύονται σε αυτές. Η Lucene είναι μια βιβλιοθήκη, η οποία παρέχει εργαλεία και δομικά χαρακτηριστικά για την σχεδίαση και υλοποίηση μηχανών αναζήτησης, ανεξαρτήτως της εφαρμογής και των εγγράφων. Δίνει την δυνατότητα σχεδίασης και υλοποίησης του πλήρους κύκλου ανάλυσης, ευρετηρίασης και αναζήτησης κειμένου.

Εκ πρώτης όψεως, τα τρία συστήματα φαντάζουν ξεχωριστά, παρόλα αυτά έχουν ομοιότητες, κοινές λειτουργίες και κοινούς στόχους σε ορισμένες περιπτώσεις. Η Elasticsearch είναι η πιο δημοφιλής μηχανή αναζήτησης, η MongoDB η πιο δημοφιλής βάση δεδομένων προσανατολισμένη σε έγγραφα, ενώ η Lucene η πιο δημοφιλής βιβλιοθήκη ανάπτυξης μηχανών αναζήτησης. Ένα κοινό χαρακτηριστικό και των τριών συστημάτων είναι ότι όλα τους είναι λογισμικά ανοιχτού κώδικα. Έχουν μεγάλες κοινότητες που τα υποστηρίζουν και τα αναπτύσσουν, διευρύνοντας τις δυνατότητές τους αλλά και διορθώνοντας ζητήματα που προκύπτουν. Η Elasticsearch και η Lucene υιοθετούν την άδεια Apache 2.0, ενώ η MongoDB την Server-Side Public License (SSPL) v1.0. Όλα τα συστήματα παρέχουν δωρεάν τον πυρήνα των λειτουργιών τους αλλά οι Elasticsearch και MongoDB παρέχουν και επί πληρωμή άδειες που προσφέρουν επιπλέον επιλογές ασφάλειας και παρακολούθησης των συστημάτων αποθήκευσης.

Και τα τρία συστήματα έχουν σχεδιαστεί και υλοποιηθεί για να λειτουργούν με έγγραφα, δηλαδή με NoSQL δομές αποθήκευσης. Η Lucene έχει σχεδιαστεί για να παρέχει δυνατότητες ανεξαρτήτως της δομής των εγγράφων, αλλά επίσης ακολουθεί την λογική αποθήκευσης προσανατολισμένης σε έγγραφα. Οι MongoDB και Elasticsearch έχουν σχεδιαστεί για να

αποθηκεύουν και να διαχειρίζονται έγγραφα, με ισχυρή υποστήριξη σε δεδομένα τύπου JSON. Επίσης και τα τρία συστήματα μπορούν να εγκατασταθούν και να λειτουργήσουν σε πληθώρα λειτουργικών συστημάτων, μεταξύ αυτών πολυάριθμες εκδόσεις Linux, Windows και MAC OS. Η ρύθμιση των βασικών παραμέτρων του συστήματος αποθήκευσης γίνεται σε πολλά ξεχωριστά αρχεία στην Elasticsearch και περιλαμβάνει πιο σύνθετες ρυθμίσεις από αυτές που απαιτεί η MongoDB, η οποία συνοψίζει όλες τις ρυθμίσεις σε ένα αρχείο.

Επιπλέον, εστιάζοντας στους στόχους και τον λόγο που αναπτύχθηκε κάθε σύστημα, δεν είναι δύσκολο να κατανοήσουμε και τον τομέα που αυτό είναι πιο αποδοτικό. Η Elasticsearch σχεδιάστηκε για να παρέχει αναζήτηση πλήρους κειμένου, ανάλυση αρχείων καταγραφής και εργαλεία μηχανικής εκμάθησης. Σε αυτούς τους τομείς και υπερέχει, αφού παρέχει εξαιρετική απόδοση αναζήτησης πλήρους κειμένου, αμέτρητες συναρτήσεις συγκέντρωσης, εργαλεία μηχανικής εκμάθησης και διασύνδεση με άλλα εργαλεία για τους σκοπούς αυτούς (όπως Kibana, Logstash κλπ). Η Lucene σχεδιάστηκε για να δώσει την δυνατότητα σε σχεδιαστές να αναλύσουν κείμενο, να δημιουργήσουν ευρετήρια και να υλοποιήσουν αναζήτηση σε έγγραφα ανεξαρτήτως εφαρμογής και δομής εγγράφων. Σε αυτό, λοιπόν και υπερτερεί, αφού μπορεί να αποδώσει σε κάθε εφαρμογή, παρέχει πολλές δυνατότητες εξατομίκευσης της διαδικασίας ανάλυσης και ευρετηρίασης και υποστηρίζει όλους του τύπους εγγράφων. Τέλος, η MongoDB δημιουργήθηκε ως μια βάση δεδομένων προσανατολισμένη σε έγγραφα ώστε να μπορεί να είναι η βασική μονάδα αποθήκευσης σε εφαρμογές τεραστίου όγκου δεδομένων. Εκεί και διαπρέπει, αφού παρέχει πολλά εργαλεία για την διαχείριση της κλιμάκωσης των δεδομένων και δίνει πλήρη έλεγχο των δεδομένων με δυνατότητες εγγραφής, ανάγνωσης, ανανέωσης και διαγραφής δεδομένων. [5] [7] [11] [13] [15]

7.2 Αποθήκευση, αναζήτηση και ευρετήρια

Η αρχιτεκτονική αποθήκευσης στην Lucene ακολουθεί μια πιο χρονοβόρα διαδικασία. Μετά από κάθε αποθήκευση τα δεδομένα αποθηκεύονται σε ένα νέο αντίγραφο και οι αλλαγές συγχωνεύονται στην κεντρική δομή αποθήκευσης ύστερε, με βάση τις ρυθμίσεις συγχώνευσης. Αυτό κάνει τις εγγραφές σε ιδιαίτερα αρχεία επιβαρυντικές για την απόδοση του συστήματος. Η Elasticsearch βελτιώνει το παραπάνω σε μεγάλο βαθμό, αφού συγκρατεί ένα αρχείο καταγραφής αλλαγών για τα ευρετήρια, κάνοντας τις συγκρίσεις και τις συγχωνεύσεις αρκετά πιο ελαφριές. Το αρχείο καταγραφής, επίσης, χρησιμεύει και σε περιπτώσεις ανάκτησης

δεδομένων από φθορά. Η MongoDB ακολουθεί ένα διαφορετικό μοντέλο. Χρησιμοποιεί έναν χάρτη μνήμης για να μεταφέρει τις δομές αποθήκευσης του σκληρού δίσκου στην μνήμη. Έτσι, κάθε αρχείο είναι μέρος μια λίστας, που οδηγεί από αρχείο σε αρχείο. Ακόμη, η MongoDB χρησιμοποιεί, επίσης, αρχεία καταγραφής δραστηριότητας, που είναι χρήσιμα σε περιπτώσεις σφαλμάτων και σε πιθανό ξαφνικό κλείσιμο του συστήματος, επιτρέποντας την ανάκτηση, ενώ μπορεί να τερματίζει αυτόματα σε περιπτώσεις εξάντλησης των υπολογιστικών πόρων. Η MongoDB για τους παραπάνω λόγους είναι ικανή να γράφει τεράστιες ποσότητες δεδομένων πιο αποδοτικά και να διαχειρίζεται μεγάλο φόρτο χωρίς να επιβαρύνει τον επεξεργαστή και τον σκληρό δίσκο.

Τα ευρετήρια στην MongoDB βασίζονται σε B Δέντρα και αφορούν συγκεκριμένα πεδία των εγγράφων. Επιπλέον πεδία μπορούν να ενταχθούν στα ευρετήρια, αλλά αυτό θα μειώσει την απόδοση των λειτουργιών εγγραφής. Τα ευρετήρια που υποστηρίζει η Elasticsearch βασίζονται σε αυτά που παρέχει και η Lucene. Οι Elasticsearch και Lucene υποστηρίζουν ανανέωση των ευρετηρίων σε πραγματικό χρόνο και διατήρηση ευρετηρίων για κάθε πεδίο των εγγράφων. Αυτό τους προσδίδει τεράστια δύναμη αναζήτησης, ανακτώντας τα κατάλληλα έγγραφα με εξαιρετική απόδοση.

Εδώ γίνεται ξεκάθαρος ο διαχωρισμός των στόχων και των δυνατών σημείων ανάμεσα σε Elasticsearch, Lucene και MongoDB. Οι δύο πρώτες στοχεύουν στην διατήρηση και συντήρηση ισχυρών ευρετηρίων για ταχύτερη αναζήτηση, κάνοντας συμβιβασμούς ως προς τις ταχύτητες εγγραφής. Η MongoDB στοχεύει στην αποδοτική ανάκτηση και εγγραφή δεδομένων σε μια ολοκληρωμένη βάση. Διατηρεί τα πιο συχνά χρησιμοποιούμενα πεδία σε ευρετήρια για να βελτιώσει την ταχύτητα ανάγνωσης στις πιο συχνές περιπτώσεις, αλλά μπορεί να γράφει εξίσου γρήγορα χάρη στα ελαφρά ευρετήριά της. Ο συμβιβασμός έρχεται σε περιπτώσεις ανάγνωσης πέραν του συνηθισμένου (εκτός ευρετηρίων) και σε περιπτώσεις που τα ευρετήρια απαιτούν επέκταση. Τέλος, η Elasticsearch δίνει την δυνατότητα ερωτημάτων με χρήση REST API και αναζήτησης πλήρους κειμένου, ενώ η MongoDB όχι. [3] [6] [15]

7.3 Κλιμάκωση, αντίγραφα και ευχρηστία

Όσον αφορά στην κλιμάκωση και στην διαχείριση ολοένα και αυξανόμενου όγκου δεδομένων και τα τρία συστήματα μεριμνούν και παρέχουν δυνατότητες προς αυτή την κατεύθυνση. Η Lucene δίνει την δυνατότητα διαχείρισης της κλιμάκωσης μέχρι ενός σημείου, αλλά δυστυχώς σε αυξημένες τάξεις μεγέθους πολλές εφαρμογές Lucene χρειάζονται προσαρμογές. Ως

κατανεμημένα συστήματα αποθήκευσης δεδομένων, Elasticsearch και MongoDB παρέχουν δυνατότητες για την κλιμάκωση του συστήματος αποθήκευσης και τον διαμοιρασμό του σε συστάδες υπολογιστών. Συγκεκριμένα, η Elasticsearch δίνει λύσεις στο πρόβλημα της Lucene χάρη στα θραύσματα που χρησιμοποιεί, διαμελίζοντας ευρετήρια και αναλαμβάνοντας την παραλληλοποίηση των ερωτημάτων για τον καλύτερο καταμερισμό του φόρτου εργασίας της αναζήτησης. Η MongoDB εκτός από την δυνατότητα διατήρησης συστάδας υπολογιστών, ως διακομιστή της βάσης, προσφέρει εξίσου οριζόντια κλιμάκωση, κατακερματίζοντας μια βάση σε επιμέρους μικρότερες βάσεις, παραλληλοποιώντας διαδικασίες. Οι MongoDB και Elasticsearch κάνουν καταπληκτική δουλειά σε επίπεδο κλιμάκωσης αναλαμβάνοντας τον κατακερματισμό και την σειριοποίηση όπου απαιτείται, απελευθερώνοντας τον χρήστη από αυτές τις αρμοδιότητες.

Η καταγραφή και η διαχείριση αντιγράφων ασφαλείας είναι κάτι στο οποίο υστερεί η Lucene. Δεν παρέχει ιδιαίτερα εργαλεία προς αυτή την κατεύθυνση και εστιάζει σε άλλες λειτουργίες. Η Elasticsearch υποστηρίζει τα αντίγραφα ασφαλείας με αύξοντα αριθμό, ώστε να είναι μοναδικά και να απορρίπτονται ευκολότερα. Παρέχει εξαιρετικά εργαλεία, ώστε με απλές εντολές, ο χρήστης μπορεί να καταγράψει στιγμιότυπα των δεδομένων, να τα απορρίψει, να τα ανακτήσει και να τα συνδέσει με τον αποθηκευτικό του χώρο στο νέφος. Η MongoDB υποστηρίζει τα αντίγραφα ασφαλείας μέσα από διάφορα επιμέρους εργαλεία. Υπάρχει η δυνατότητα εφαρμογής ερωτημάτων στα αντίγραφα αυτά και η δυνατότητα αντιγραφής ανά συλλογή, αλλά όχι η δυνατότητα για αντίγραφα ασφαλείας με αύξοντα αριθμό ή η δυνατότητα διασύνδεσης με αποθηκευτικό χώρο στο νέφος. Αυτές οι δυνατότητες πρέπει να υλοποιηθούν από τον χρήστη, αυξάνοντας την πολυπλοκότητα της σχεδίασης.

Μια υλοποίηση Lucene για μια μηχανή αναζήτησης προϋποθέτει μια ολοκληρωμένη σχεδίαση και ανάπτυξη του κύκλου ανάλυσης, ευρετηρίασης και αναζήτησης κειμένου. Δηλαδή, εκτός από τα κλασσικά βήματα σχεδίασης της δομής των εγγράφων και των συσχετίσεών τους, πρέπει γίνουν πολλές επιπλέον ενέργειες, ώστε να γίνει εφικτή η αναζήτηση. Μια μηχανή αναζήτησης σε Elasticsearch απαιτεί την παραμετροποίηση σε μεγάλο βαθμό για τις ανάγκες της βελτιστοποίησης της αναζήτησης, επιπλέον της δημιουργίας των ευρετηρίων και της σχεδίασης της δομής των εγγράφων. Αυτό καθιστά εξίσου πολύπλοκη την ανάπτυξη ενός τέτοιου συστήματος, χωρίς, όμως, να είναι αναγκαία η σχεδίαση όλων των βημάτων από το μηδέν. Από την άλλη, η MongoDB απαιτεί, σε γενικές γραμμές, μόνο την σχεδίαση των εγγράφων και ευρετηρίων. Αυτό την καθιστά το πιο εύχρηστο εργαλείο από τα παραπάνω. Επιπρόσθετα, Elasticsearch μπορεί να υποστηρίζει παραπάνω γλώσσες προγραμματισμού

(Java, Ruby, Javascript, GO, .NET, Python, PHP, Rust και Perl), ωστόσο και η MongoDB πέραν από την επίσημη υποστήριξη που παρέχει (σε C, C++, Scala και Swift) λειτουργεί και με τους ανοιχτού κώδικα διακομιστές πολλών γλωσσών προγραμματισμού που έχει αναπτύξει η κοινότητά της. [3] [5] [6] [11] [15]

7.4 Προβλήματα

Ένα από τα βασικά ζητήματα που προέκυψαν κατά την έρευνα γύρω από τις MongoDB, Elasticsearch και Lucene είναι η περιορισμένη βιβλιογραφία. Παρότι τα τρία αυτά συστήματα υποστηρίζονται από καλώς ορισμένα έγγραφα και υπάρχει πληθώρα οδηγών στο διαδίκτυο για την χρήση τους, δεν υπάρχουν πολλές πειραματικές ακαδημαϊκές έρευνες που να τα αναλύουν. Ιδιαίτερα, δεν υπάρχουν έρευνες που να τα συγκρίνουν μεταξύ τους και που να βλέπουν και τα τρία συστήματα από την ίδια σκοπιά. Από την μια, τα τρία συστήματα αποθήκευσης και αναζήτησης χρησιμοποιούνται ευρέως στην βιομηχανία και έχουν σχεδιαστεί για να την εξυπηρετούν και όχι για ερευνητικούς και εκπαιδευτικούς σκοπούς. Από την άλλη, τα συστήματα αυτά είναι μεν εδραιωμένα, αλλά δεν παύουν να είναι αρκετά νεότερα από άλλα είδη βάσεων δεδομένων. Αυτό οδηγεί τους ερευνητές σε ακαδημαϊκό επίπεδο να στρέφουν τις έρευνές τους σε άλλα συστήματα. Παρόλα αυτά, για τις ανάγκες της εργασίας, ήταν τελικά αρκετές κάποιες επιμέρους έρευνες για κάθε σύστημα, όπως και η επίσημη τεκμηρίωση των MongoDB, Elasticsearch και Lucene από τους δημιουργούς τους.

Μια ακόμα δυσκολία που παρουσιάστηκε στην συγκριτική μελέτη των τριών αυτών συστημάτων, ήταν οι διαφορές που αυτά εμφανίζουν μεταξύ τους. Η MongoDB είναι μια βάση δεδομένων με βάση τα έγγραφα. Η Lucene είναι μια βιβλιοθήκη που επιτρέπει την ανάπτυξη μηχανών αναζήτησης. Η Elasticsearch είναι μια μηχανή αναζήτησης βασισμένη στην Lucene. Η σύγκριση των τριών απαιτεί την εύρεση κοινών σημείων μεταξύ τους και την επιλογή των κατάλληλων συνιστωσών της σύγκρισης. Ουσιαστικά, η σύγκριση των τριών ανόμοιων συστημάτων έγινε σε επιμέρους τμήματα τα οποία και παρουσιάζουν κοινές λειτουργίες.

Ένας ακόμη προβληματισμός, που αντιμετωπίστηκε κατά την διάρκεια της εκπόνησης της παρούσας εργασίας ήταν τα έντονα τεχνικά χαρακτηριστικά που περιλαμβάνει η εργασία αυτή. Η συγκριτική μελέτη των τριών συστημάτων απαιτεί και την ανάλυση των δυνατοτήτων αυτών συνοπτικά. Τα όρια, ωστόσο στις λεπτομέρειες στις οποίες έπρεπε να εισχωρήσει η εργασία είναι δυσδιάκριτα. Σκοπός της

εργασίας είναι να δώσει την γενική εικόνα του κάθε συστήματος, ώστε να είναι σαφή τα συμπεράσματα, οι συγκρίσεις και ο τρόπος χρήσης αυτών σε υψηλό επίπεδο. Από την άλλη, όμως, κάποιες λεπτομέρειες είναι απαραίτητες ώστε να γίνουν κατανοητά τα παραπάνω. Το βάθος στο οποίο έπρεπε να εισχωρήσει η εργασία στα τεχνικά χαρακτηριστικά των MongoDB, Elasticsearch και Lucene ήταν μια πρόκληση ώστε αφενός να είναι σαφής και πλήρης η εργασία αλλά αφετέρου να μην χαθεί και το κεντρικό θέμα της καταλήγοντας σε έναν ακόμη οδηγό χρήσης των συστημάτων.

7.5 Γενικά Συμπεράσματα

Τα τρία συστήματα που μελετήθηκαν και ιδιαίτερα οι MongoDB και Elasticsearch είναι συστήματα αποθήκευσης και ανάκτησης εγγράφων που έχουν σχεδιαστεί και υλοποιηθεί για να δώσουν λύσεις σε βιομηχανικό επίπεδο. Παρέχουν διευκολύνσεις στους σχεδιαστές, βελτιστοποιούν διαδικασίες και υποστηρίζουν συμβατότητα με πληθώρα εφαρμογών. Είναι ξεκάθαρο πως έχουν σχεδιαστεί και υλοποιηθεί για να μπορούν να υποστηρίξουν εφαρμογές κάθε είδους, να λειτουργούν σταθερά σε αυξανόμενου μεγέθους δεδομένα και να παρέχουν διασύνδεση σε εφαρμογές του διαδικτύου. Επιτρέπουν την σύνδεση με τις πιο συχνά χρησιμοποιούμενες γλώσσες προγραμματισμού, χρησιμοποιώντας βέλτιστες τακτικές για την αποθήκευση και ανάκτηση της πληροφορίας. Ακόμη, έχουν σχεδιαστεί και υλοποιηθεί κατά την διάρκεια της εποχής των διαδικτυακών εφαρμογών, λαμβάνοντας υπ' όψιν τις τεχνικές και τα εργαλεία που χρησιμοποιούνται στις μέρες μας αλλά και τις ανάγκες και τις τάσεις της εποχής μας.

Τα τρία συστήματα, με έμφαση στις MongoDB και Elasticsearch έχουν σχεδιαστεί και υλοποιηθεί για να είναι αποδοτικά σε τεράστιου όγκου δεδομένα. Είναι διανεμημένα συστήματα αποθήκευσης, χρησιμοποιούν ευρετήρια ταχεία αναζήτησης και παρέχουν οριζόντια κλιμάκωση σε μεγάλο επίπεδο. Από τις πιο σημαντικές εφαρμογές των δύο αυτών συστημάτων είναι η υποστήριξη εφαρμογών Big Data καθώς η απόδοσή τους σε μεγάλους όγκους δεδομένων είναι κορυφαία.

Ένα ακόμη σημαντικό συμπέρασμα είναι ότι MongoDB και Elasticsearch μπορούν να συνυπάρξουν σε εφαρμογές, ώστε να αλληλοσυμπληρώνονται για να καλυφθούν καλύτερα οι ανάγκες κάποιας

εφαρμογής. Από την μία, η Elasticsearch θα μπορούσε να αναλάβει τον ρόλο της μηχανής αναζήτησης με πλήρη ευρετηρίαση, παρέχοντας αποδοτική αναζήτηση στην εφαρμογή. Από την άλλη, η MongoDB θα μπορούσε να αναλάβει την κεντρική βάση δεδομένων της εφαρμογής, ώστε να διεκπεραιώνει τις βασικές ανάγκες εγγραφής ταχύτερα. Για την ακρίβεια, το παραπάνω μοντέλο χρησιμοποιείται ευρέως στην βιομηχανία λογισμικού, στις μέρες μας, σε μεγάλες εφαρμογές και παρουσιάζει πολύ μεγάλο ενδιαφέρον.

Εν κατακλείδι, ένα από τα πιο σημαντικά συμπεράσματα που προκύπτουν είναι πως τα τρία αυτά συστήματα έχουν διαφορετικούς στόχους και για αυτό είναι κορυφαία σε διαφορετικούς τομείς, προσφέρουν συμπληρωματικές υπηρεσίες και κανένα δεν μπορεί να αντικαταστήσει τα υπόλοιπα. Η επιλογή του κατάλληλου συστήματος, εξαρτάται από την εφαρμογή. Η Lucene είναι μια κορυφαία βιβλιοθήκη ανάπτυξης μηχανών αναζήτησης, αφού δίνει τεράστια ελευθερία σχεδίασης, πολλά εργαλεία ανάλυσης και ευρετηρίασης και υποστηρίζει κάθε εφαρμογή και κάθε είδος εγγράφων. Δεν παρέχει, ωστόσο, έτοιμες λύσεις σε πολλά ζητήματα, όπως η κλιμάκωση και απαιτεί πολύπλοκη σχεδίαση και υλοποίηση πριν την χρήση. Η Elasticsearch είναι κορυφαία μηχανή αναζήτησης ή αναλυτής αρχείων καταγραφής για εύρεση ανωμαλιών. Παρέχει ανάλυση πλήρους κειμένου σε εξαιρετική απόδοση, μεγάλο αριθμό μοντέλων μηχανικής εκμάθησης, προσφέρει κλιμάκωση και ότι χρειάζεται μια εφαρμογή για αναζήτηση σε έγγραφα ανεξαρτήτως μεγέθους της βάσης. Παρόλα αυτά είναι επιρρεπείς σε αστοχίες εγγραφής και αποδίδει ελάχιστα σε τέτοιου είδους λειτουργίες. Τέλος, η MongoDB είναι κορυφαία βάση δεδομένων προσανατολισμένη σε έγγραφα καθώς αποδίδει εξαιρετικά τόσο σε λειτουργίες ανάγνωσης όσο και σε εγγραφής, υποστηρίζει πολυεπίπεδη κλιμάκωση και είναι αξιόπιστη. Όμως, δεν παρέχει αναζήτηση πλήρους κειμένου, ενώ χάνει την απόδοσή της σε εφαρμογές αναζήτησης που απαιτούν ευρετήρια σε κάθε πεδίο.

7.6 Μελλοντικές Επεκτάσεις

Η παρούσα εργασία αφορά μια συγκριτική μελέτη μεταξύ MongoDB, Elasticsearch και Lucene και εστιάζει σε θεωρητικό υπόβαθρο. Θα είχε ιδιαίτερο ενδιαφέρον μια επέκταση της παρούσας εργασίας σε πιο πειραματικό επίπεδο. Μια καλή τέτοια επέκταση θα ήταν η ανάπτυξη μιας εφαρμογής που απαιτεί ένα σύστημα αποθήκευσης δεδομένων. Θα μπορούσαν να χρησιμοποιηθούν διαδοχικά οι MongoDB, Elasticsearch και Lucene για την υλοποίηση της εφαρμογής, ώστε να καταγραφούν σε πρακτικό επίπεδο κάποιες μετρικές για την αξιολόγηση των δυνατοτήτων των τριών συστημάτων. Για την καλύτερη και πιο πλήρη αξιολόγηση των συστημάτων θα ήταν χρήσιμη η ανάπτυξη περισσότερων της μιας

εφαρμογής, ώστε να αξιολογηθούν τα συστήματα τόσο σαν βάσεις δεδομένων, όσο και σαν μηχανές αναζήτησης σε διάφορες συνθήκες.

Στην παρούσα μελέτη τα προς ανάλυση συστήματα αποτελούν κατανεμημένα συστήματα διαχείρισης αποθήκευσης βασισμένα στα έγγραφα. Υπήρξε ένας συσχετισμός με τις παραδοσιακές σχεσιακές βάσεις δεδομένων, αλλά αυτός στόχευε περισσότερο στην κατανόηση των NoSQL συστημάτων αποθήκευσης παρά στην ουσιαστική σύγκριση των δυο τεχνολογιών. Θα είχε, λοιπόν, μεγάλο ενδιαφέρον, μια επιπλέον συγκριτική μελέτη που να εστιάζει στις δύο αυτές τεχνολογίες και που να αποσαφηνίζει τις δυνατότητες και τις αδυναμίες κάθε μιας, αλλά και τους τρόπους που αυτές μπορούν αποδώσουν σε συνεργασία. Μια πρακτική εφαρμογή με τρεις υλοποιήσεις μιας παρόμοιας βάσης σε σχεσιακή, σε βασισμένη σε έγγραφα και σε συνδυασμό των δύο θα ήταν επίσης ιδιαίτερα ενδιαφέρουσα προοπτική στην μελέτη.

Πολλές εφαρμογές χρησιμοποιούν ταυτόχρονα MongoDB και Elasticsearch για τις ανάγκες τους. Η μελέτη αυτής της διασύνδεσης θα είχε τεράστιο ενδιαφέρον, ως διεύρυνση της παρούσας εργασίας. Ο τρόπος αναζήτησης, εγγραφής και διαχείρισης δεδομένων σε μια τέτοια εφαρμογή, όπως και οι τεχνικές για την αποδοτικότερη διασύνδεση MongoDB και Elasticsearch θα μπορούσαν να αναλυθούν ευκολότερα, ύστερα από την παρούσα συγκριτική μελέτη των δυνατοτήτων τους. Η μελέτη της διασύνδεσης αυτής θα είχε πολλά πράγματα να αναλύσει όπως το συνολικό κόστος συντήρησης, τα πλεονεκτήματα, τα μειονεκτήματα και τις αρμοδιότητες κάθε συστήματος. Τέλος, μια μικρή εφαρμογή, σαν παράδειγμα της διασύνδεσης θα μπορούσε να συμπληρώσει ιδανικά την επιπρόσθετη αυτή μελέτη.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Χρίστος Μπανάτσας, (2017), "Εφαρμογή Android για εύρεση και βαθμολογία παιδότοπων", Αλεξάνδρειο Τεχνολογικό Εκπαιδευτικό Ίδρυμα Θεσσαλονίκης, Τμήμα Μηχανικών Πληροφορικής.

[2] Παναγιώτης Κρουσταλιός, (2015), "Τεχνικές Βελτιστοποίησης της Αποθήκευσης και Αναζήτησης Big Data χρησιμοποιώντας την Βάση Δεδομένων MongoDB", Εθνικό Μετσόβιο Πολυτεχνείο, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Τομέας επικοινωνιών, ηλεκτρονικής και συστημάτων πληροφορικής.

[3] Kyle Banker, Peter Bakkum, Shaun Verch, Douglaw Garrett, Tim Hawkins, (2016), "MongoDB in Action", Second Edition, Manning Publications Co., Shelter Island, New York, USA, ISBN 9781617291609.

[4] Κωνσταντίνος Κοκολόγος, (2015), "Επεξεργασία Χωρικών Επερωτήσεων στην MongoDB", Πανεπιστήμιο Πειραιά, Τμήμα Ψηφιακών Συστημάτων, Ψηφιακά Συστήματα και Υπηρεσίες, Μεγάλα Δεδομένα και Αναλυτική.

[5] MongoDB, (2021), "The MongoDB Documentation", MongoDB, · https://docs.mongodb.com.

[7] Clinton Gormley, Zachary Tong (2015), "Elasticsearch: The Definitive Guide", First Edition, O'Reilly Media, Inc., 1005 Gravenstein, Highway North, Sebastopol, USA, ISBN 9781449358549.

[8] Yuvraj Gupta, (2015), "Kibana Essentials", First Edition, Packt Publishing Ltd., 35 Livery Street, Birmingham, UK, ISBN 9781784394936.

[9] Σταύρος Δημάρχου, (2014), " Σχεδιασμός και ανάπτυξη ενός κατανεμημένου συστήματος για τη συλλογή, ανάλυση και αποθήκευση των δεδομένων χρήσης που παράγονται από τη λειτουργία ενός υπολογιστικού νέφους", Εθνικό Μετσόβιο Πολυτεχνείο, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Τομέας τεχνολογίας πληροφορικής και υπολογιστών.

[10] Αστέριος Μπαμπάκης, (2019), "Ανάλυση ποιότητας λογισμικού και εφαρμογή στο Elasticsearch", Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, Σχολή Θετικών Επιστημών, Τμήμα Πληροφορικής.

[11] Δημήτριος Π. Σπυρόπουλος, (2017), "Εξετάζοντας δημόσια δεδομένα με τη βοήθεια του Elastic Search", Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών, Σχολή Θετικών Επιστημών, Τμήμα Πληροφορικής και Τηλεπικοινωνιών.

[12] Elasticsearch, (2021), "Elasticsearch Guide", Elasticsearch, https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html.

[13] Erik Hatcher, Otis Gospodnetic, Michael McCandless, (2010), "Lucene in Action", Second Edition, Manning Publications, Shelter Island, New York, USA, ISBN 9781933988177

[14] Andrzej Bialecki, Robert Muir, Grant Ingersoll, (2012), "Apache Lucene 4", Proceedings of the SIGIR 2012 Workshop on Open Source Information Retrieval, Portland, Oregon, USA

[15] Mamatha Balipa, Balasubramani R, (2015), "Search Engine using Apache Lucene", International Journal of Computer Applications (0975 - 8887), Volume 127 - No.9

[16] Edwood Ng, Vineeth Mohan, (2015), "Lucene 4 Cookbook", First Edition, Packt Publishing Ltd., 35 Livery Street, Birmingham, UK, ISBN 9781782162285

[17] Ramakrishnan, Raghu, Gehrke, Johannes, (2012), "Συστήματα Διαχείρισης Βάσεων Δεδομένων", 3η έκδοση, Εκδόσεις Τζιόλα., Φιλίππου 91, Θεσσαλονίκη, Ελλάδα, ISBN 9789604184118

[18] Jing Han, Haihong E, Guan Le, Jian Du, (2011). "Survey on NoSQL database", 2011 6th International Conf

erence on Pervasive Computing and Applications, pp. 363-366, doi: 10.1109/ICPCA.2011.6106531, IEEE