Αποθηκευτική Διαχείριση Δεδομένων Χρονοσειρών για Διαδραστική Ανάκτηση και Απεικόνιση

Η Μεταπτυχιακή Διπλωματική Εργασία

υποβάλλεται στην ορισθείσα
από τη Συνέλευση
του Τμήματος Μηχανικών Η/Υ και Πληροφορικής
Εξεταστική Επιτροπή

από τον

Γιάννη Μπάκο

ως μέρος των υποχρεώσεων για την απόκτηση του

ΔΙΠΛΩΜΑΤΟΣ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΗ ΠΛΗΡΟΦΟΡΙΚΗ
ΜΕ ΕΙΔΙΚΕΥΣΗ
ΣΤΑ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

Πανεπιστήμιο Ιωαννίνων
Πολυτεχνική Σχολή
Ιωάννινα 2021

Εξεταστική επιτροπή:

- ***Αναστασιάδης Στέργιος,*** Αναπλ. Καθηγητής Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων (Επιβλέπων)
- ***Παληός Λεωνίδας,*** Καθηγητής Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων
- ***Τσαπάρας Παναγιώτης,*** Αναπλ. Καθηγητής Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων

Αφιερωση

Αφιερώνω την παρούσα εργασία σε όλους τους πυροσβέστες της χώρας οι οποίοι μάχονται με όλες τους τις δυνάμεις σε όλα τα μέτωπα και ήταν η αφορμή να εκπονηθεί η παρούσα εργασία...

Ευχαριστιες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Στέργιο Αναστασιάδη που με τις συμβουλές και τις προτροπές έγινε εφικτό να εκπονηθεί η παρούσα εργασία.

Επίσης, σημαντικές ευχαριστίες θα ήθελα να εκφράσω στην ομάδα Υπολογιστικών Συστημάτων του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων που όλο το διάστημα της φοίτησης μου στο τμήμα ήταν οι άνθρωποι που συμβουλεύτηκα σε οποιοδήποτε θέμα αντιμετώπισα. Πιο συγκεκριμένα, θα ήθελα να ευχαριστήσω τον κ. Καππέ Γιώργο, τον κ. Δημουλή Ευαγγελό, τον κ. Κατσουλιέρη Αργύρη, τον κ. Φάνη Μπαλάσκα, τον κ. Μπακοχρήστο Παναγιώτη.

Τέλος, θα ήθελα να ευχαριστήσω όλους τους καθηγητές τους τμήματος μας που κάνουν τα πάντα για να αξιοποιηθούν όσο το δυνατό καλύτερα όλοι οι διαθέσιμοι πόροι -υλικοί και πνευματικοί- σε αυτή την δύσκολη εποχή.

ΠΕΡΙΕΧΟΜΕΝΑ

Καταλογος Σχημάτων iii

Περίληψη vi

Extended Abstract viii

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή 9

1.1 Στόχοι 9

1.2 Δομή της Εργασίας 10

ΚΕΦΑΛΑΙΟ 2 Υπόβαθρο 11

2.1 Περιγραφή του προβλήματος και των δεδομένων 11

2.2 Περιγραφή της τελικής εφαρμογής....13

2.3 Διαχείριση χρονοσειρών 15

2.4 Διαχείριση γεωγραφικών δεδομένων....17

2.4.1 Ευρετηριοποίηση....18

2.5 Συστήματα Βάσεων Δεδομένων κατάλληλων για χρονοσειρές και γεωχωρικά δεδομένα 19

2.5.1 Το σύστημα διαχείρισης δεδομένων TimescaleDB 20

2.5.2 Το σύστημα διαχείρισης δεδομένων MongoDB....20

ΚΕΦΑΛΑΙΟ 3 Σχεδίαση 22

3.1 Οφέλη του συστήματος της MongoDB 22

3.2 Σχεδίαση του αποθηκευτικού χώρου για το μετεωρολογικό μοντέλο 25

3.2.1 Απλή αποθήκευση μετεωρολογικών δεδομένων....28

3.2.2 Προηγμένη αποθήκευση μετεωρολογικών δεδομένων....31

3.2.3 Σχεδίαση ευρετηρίων για την προηγμένη αποθήκευση μετεωρολογικών δεδομένων 34

3.2.4 Αποθήκευση δεδομένων με βάση το επίπεδο ανάλυσης.....35
3.2.5 Σύνοψη για την αποθήκευση των δεδομένων με τις δύο εκδοχές 36
3.3 Σχεδίαση του αποθηκευτικού χώρου για τα δεδομένα από αισθητήρες καιρικών συνθηκών 37
3.3.1 Αποθήκευση δεδομένων της πρόσφατης πληροφορίας 38
3.3.2 Αποθήκευση δεδομένων ιστορικού χαρακτήρα 39
3.3.3 Ευρετηριοποίηση των συλλογών αποθήκευσης δεδομένων από αισθητήρες.....40
3.3.4 Αντίγραφα ασφαλείας των δεδομένων από τους αισθητήρες των μετεωρολογικών σταθμών.....41
ΚΕΦΑΛΑΙΟ 4 Υλοποίηση **42**
4.1 Υλοποίηση αποθήκευσης δεδομένων για την πρόγνωση.....43
4.2 Υλοποίηση τελικής εφαρμογής για την πρόγνωση 47
4.3 Υλοποίηση αποθήκευσης δεδομένων για τους μετεωρολογικούς σταθμούς 52
4.3.1 Αποθήκευση πρόσφατων δεδομένων από μετρήσεις 54
4.3.2 Αποθήκευση ιστορικών δεδομένων μετεωρολογικών σταθμών.....55
4.4 Υλοποίηση τελικής εφαρμογής για τους μετεωρολογικούς σταθμούς.....57
ΚΕΦΑΛΑΙΟ 5 Πειραματική Αξιολόγηση **61**
5.1 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή της πρόγνωσης ...62
5.2 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή των μετεωρολογικών σταθμών..... 69
ΚΕΦΑΛΑΙΟ 6 Επίλογος **72**
Βιβλιογραφία **74**

Κατάλογος Σχημάτων

Σχήμα 2.1: Στιγμιότυπο από την εκτέλεση της διαδικτυακής εφαρμογής windy.com. 13
Σχήμα 2.2: Μέθοδος ευρετηριοποίησης με βάση δέντρα περιοχών. 18
Σχήμα 3.1: Δομή των δεδομένων στην MongoDB. 24
Σχήμα 3.2: Επικοινωνίας μοντέλου πρόγνωσης και συστήματος αποθήκευσης. 25
Σχήμα 3.3: Δομή αρχείου NetCDF. 26
Σχήμα 3.4: Σχήμα επικοινωνίας δεδομένων από μετεωρολογικούς σταθμούς και συστήματος αποθήκευσης. 27
Σχήμα 3.5: Δομή απλής αποθήκευσης μετεωρολογικών δεδομένων 29
Σχήμα 3.6: Διαδικασία επιπλέον επεξεργασίας στην συλλογή 31
Σχήμα 3.7: Δομή προηγμένης αποθήκευσης μετεωρολογικών δεδομένων 33
Σχήμα 3.8: Δομή αποθήκευσης δεδομένων αισθητήρων. 39
Σχήμα 4.1: Κύρια διάκριση βάσεων δεδομένων. 43
Σχήμα 4.2: Εσωτερική σχεδίαση της βάσης δεδομένων για την περιοχή της Ηπείρου 43
Σχήμα 4.3: Εσωτερική σχεδίαση της συλλογής θερμοκρασίας για την περιοχή της Ηπείρου 44
Σχήμα 4.4: Εσωτερική σχεδίαση της συλλογής για την περιοχή της Ηπείρου 45
Σχήμα 4.5: Ευρετήρια για την συλλογή των φίλτρων 46
Σχήμα 4.6: Στιγμιότυπο από την εφαρμογή απεικόνισης της πρόγνωσης 47
Σχήμα 4.7: Υλοποίηση δημιουργίας απεικονίσεων δεδομένων 48
Σχήμα 4.8: Εμφάνιση πληροφοριών με ένα κλικ 49
Σχήμα 4.9: Κώδικας ερωτήματος για τα φίλτρα 50

Σχήμα 4.10: Δεδομένα επιστροφής από κλήση API 51
Σχήμα 4.11: Απεικόνιση χειροκίνητων φίλτρων φίλτρων κινδύνου πυρκαγιάς 52
Σχήμα 4.12: Ερωτήματα για φιλτράρισμα με βάση τα δοθέντα κριτήρια 53
Σχήμα 4.13: Διαχωρισμός αποθήκευσης συλλογών δεδομένων μετεωρολογικών σταθμών 54
Σχήμα 4.14: Εσωτερική σχεδίαση της συλλογής αποθήκης όλων των δεδομένων τρίτων σταθμών 55
Σχήμα 4.15: Σχεδίαση της συλλογής αποθήκευσης των δεδομένων μίας παραμέτρου 56
Σχήμα 4.16: Σχεδίαση της συλλογής αποθήκευσης δεδομένων μίας παραμέτρου (δεύτερη εκδοχή) 57
Σχήμα 4.17: Εφαρμογή απεικόνισης μετεωρολογικών σταθμών 58
Σχήμα 4.18: Εκτέλεση ερωτήματος για πρόσφατα δεδομένα 58
Σχήμα 4.19: Αποθήκευση πρόσφατων πληροφοριών σταθμού 59
Σχήμα 4.20: Κλήση μεθόδου για ιστορικά δεδομένα σταθμών 60
Σχήμα 5.1: Μετρήσεις πειραμάτων στην εφαρμογή ιστορικού πρόγνωσης πάνω σε δύο συλλογές με ευρετήριο και χωρίς αντίστοιχα..... 62
Σχήμα 5.2: Μετρήσεις πειραμάτων στην εφαρμογή ιστορικού πρόγνωσης πάνω σε δύο συλλογές διαφορετικών μεγεθών..... 63
Σχήμα 5.3: Ερώτημα για την ανάκτηση των πρόσφατων πληροφοριών σημείου...64
Σχήμα 5.4: Ανάκτηση πρόσφατων δεδομένων σημείου από την εφαρμογή ή απευθείας από την βάση δεδομένων..... 65
Σχήμα 5.5: Ανάκτηση πρόσφατων δεδομένων σε αποσυνδεδεμένη σχεδίαση αποθήκευσης 66
Σχήμα 5.6: Χρόνος εκτέλεσης για την ανάκτηση των δυναμικών δεδομένων με βάση την ακτίνα επιλογής 67
Σχήμα 5.7: Πλήθος σημείων που ανακτώνται από την χρήση των δυναμικών δεδομένων με βάση την ακτίνα επιλογής 67
Σχήμα 5.8: Μετρήσεις πειραμάτων στην εφαρμογή μετεωρολογικών σταθμών με διαφορετικές μηχανές αποθήκευσης 69

Σχήμα 5.9: Χρόνος εκτέλεσης ερωτημάτων πρόσφατων δεδομένων με χρήση των διαφορετικών εκδοχών 70
Σχήμα 5.10: Χρόνος εκτέλεσης ερωτημάτων πρόσφατων δεδομένων με χρήση των διαφορετικών εκδοχών 71

Περιληψη

Γιάννης Μπάκος, Δ.Μ.Σ. στη Πληροφορική Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική Σχολή, Πανεπιστήμιο Ιωαννίνων, Αύγουστος 2021
Αποθηκευτική Διαχείριση Δεδομένων Χρονοσειρών για Διαδραστική Ανάκτηση και Απεικόνιση
Επιβλέπων: Στέργιος Αναστασιάδης Αναπληρωτής Καθηγητής

Η τεράστια εξάπλωση των μεγάλων δεδομένων τα τελευταία χρόνια οδηγεί στην αξιοποίησή τους από όλο και περισσότερες εφαρμογές. Οι εφαρμογές μεγάλων δεδομένων επιτρέπουν εξαγωγή χρήσιμης πληροφορίας από την ανάλυσή τους, αλλά οδηγούν και σε μια σειρά από προκλήσεις, όπως είναι η αποθήκευση και η ανάκτησή τους. Στην παρούσα μεταπτυχιακή εργασία εξετάζουμε το πρόβλημα της αποθήκευσης δεδομένων χρονοσειρών γεωγραφικής φύσης σε σύγχρονα συστήματα διαχείρισης δεδομένων με στόχο την υποστήριξη της ανάκτησης από εφαρμογές άμεσης απόκρισης.

Ειδικότερα εξετάζουμε γεωγραφικά δεδομένα χρονοσειρών που αφορούν τις μετεωρολογικές προγνώσεις και μετρήσεις μετεωρολογικών σταθμών. Η σχεδίαση της αποθήκευσης των δεδομένων αυτών βασίζεται στην βασική ιδέα της καλύτερης εξυπηρέτησης της λειτουργίας της κάθε εφαρμογής που τα χρησιμοποιεί. Σχεδιάζουμε και υλοποιούμε διαφορετικές δομές αποθήκευσης για λειτουργίες που απαιτούν γρήγορη απόκριση από πρόσφατα δεδομένα και διαφορετικές για την ανάκτηση ιστορικών δεδομένων μεγάλου όγκου. Στις εφαρμογές ιστού που χρησιμοποιούν τις συγκεκριμένες δομές ο χρόνος απόκρισης κυμαίνεται από λιγότερο του ενός δευτερολέπτου για τα πρόσφατα δεδομένα μέχρι αρκετά δευτερό-

λεπτα για ιστορικά δεδομένα. Αυτοί οι χρόνοι απόκρισης επιτυγχάνουν τους στόχους που θέσαμε για τις αντίστοιχες διαδραστικές εφαρμογές που περιγράφουμε.

Extended Abstract

Giannis Bakos, M.Sc. in Data and Computer Systems Engineering, Department of Computer Science and Engineering, School of Engineering, University of Ioannina, Greece, August 2021

Time-series Data Storage Management for Interactive Retrieval and Display

Advisor: Stergios Anastasiadis Associate Professor

The huge spread of big data in recent years leads to their utilization by more and more applications. Big data applications allow useful information to be extracted from their analysis, but they also lead to a number of challenges, such as their storage and retrieval. In this master's thesis we examine the problem of storing time series data of a geographical nature in modern data management systems in order to support recovery from instant response applications.

In particular, we examine geographical data of time series that concern the meteorological forecasts and measurements of meteorological stations. The design of the storage of this data is based on the basic idea of the best service of the operation of each application that uses them. We design and implement different storage structures for functions that require rapid response from recent data and different ones for retrieving large volumes of historical data.

In web applications that use these structures, the response time ranges from less than one second for recent data to several seconds for historical data. These response times achieve the goals we set for the respective interactive applications we describe.

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Στόχοι

1.2 Δομή της Εργασίας

1.1 Στόχοι

Στην παρούσα διπλωματική εργασία ο κύριος στόχος είναι η δημιουργία μιας προηγμένης διαδικτυακής εφαρμογής η οποία διαχειρίζεται γεωγραφικά δεδομένα χρονοσειρών. Σημαντικό κριτήριο επιτυχίας της εφαρμογής αυτής αποτελεί η ταχύτητα απόκρισης και η ελαχιστοποίηση της καθυστέρησης. Για να επιτευχθεί αυτός ο απώτερος στόχος, πρέπει να επιλυθεί ένα πλήθος από υποπροβλήματα όπως: διαχείριση χρονοσειρών, διαχείριση γεωγραφικών δεδομένων, διαχείριση πολλαπλών πηγών δεδομένων καθώς και προβλήματα απεικόνισης πάνω σε διαδικτυακή εφαρμογή.

Ένα σημαντικό πρόβλημα που απαιτεί επίλυση είναι διαχείριση των συγκεκριμένων δεδομένων σε μεγάλη κλίμακα γεγονός που χαρακτηρίζει όλη την σχεδίαση και την υλοποίηση της εφαρμογής.
Συνοψίζοντας, οι στόχοι διαμορφώνονται ως εξής:

- Υλοποίηση προηγμένης διαδικτυακής εφαρμογής

- Ελαχιστοποίηση καθυστέρησης και άμεση απόκριση
- Διαχείριση χρονοσειρών και γεωγραφικών δεδομένων
- Διαχείριση μεγάλου όγκου δεδομένων

1.2 Δομή της Εργασίας

Η εργασία περιέχει 6 κεφάλαια.

Στο πρώτο κεφάλαιο της Εισαγωγής παρουσιάζουμε τους στόχους της εργασίας αυτής καθώς και τα υποπροβλήματα που προβάλλονται.

Στο δεύτερο κεφάλαιο του Υποβάθρου δίνουμε με λεπτομέρειες το πρόβλημα που έχουμε να αντιμετωπίσουμε ως προς την πολυπλοκότητα των δεδομένων και της τελικής εφαρμογής.

Στο τρίτο κεφάλαιο της Σχεδίασης παρουσιάζουμε την σχεδίαση που προτείνεται για κάθε ένα από τα επιμέρους προβλήματα που έχουν τονιστεί στο Υπόβαθρο.

Στο τέταρτο κεφάλαιο της Υλοποίησης παρουσιάζουμε την υλοποίηση της αποθηκευτικής διαχείριση καθώς και την αλληλεπίδραση μεταξύ αποθηκευτικής μηχανής και της τελικής εφαρμογής.

Στο πέμπτο κεφάλαιο των Πειραμάτων παρουσιάζουμε μερικά χρήσιμα αποτελέσματα με βάση μετρήσεις από τις εφαρμογές ιστού με τις οποίες έχουμε πρόσβαση στην εφαρμογή μας.

Στο έκτο κεφάλαιο του Επιλόγου παραθέτουμε κάποιες ιδέες σχετικά με τους στόχους, την υλοποίηση και την επίτευξη των στόχων.

ΚΕΦΑΛΑΙΟ 2

Υποβαθρο

2.1 Περιγραφή του προβλήματος και των δεδομένων

2.2 Περιγραφή της τελικής εφαρμογής

2.3 Διαχείριση χρονοσειρών

2.4 Διαχείριση γεωγραφικών δεδομένων

2.5 Συστήματα Βάσεων Δεδομένων κατάλληλων για χρονοσειρές και γεωχωρικά δεδομένα

Στο παρόν κεφάλαιο θα δώσουμε περισσότερες λεπτομέρειες σχετικά με το πρόβλημα που έχουμε να αντιμετωπίσουμε ώστε να φέρουμε εις πέρας το ευρύτερο πρόβλημα που είναι η ταχύτητα απόκρισης μιας προηγμένης διαδικτυακής εφαρμογής.

Επίσης, θα παρουσιάσουμε κάποιες ιδέες που κυριαρχούν όσον αφορά την διαχείριση δεδομένων χρονοσειρών και γεωγραφικών δεδομένων. Σημαντική έννοια στην διαχείριση των δεδομένων αυτών είναι η ευρετηριοποίηση την οποία θα δούμε παρακάτω στο πλαίσιο του κεφαλαίου αυτού.
Επιπλέον, θα παρουσιάσουμε την μηχανή την οποία θα χρησιμοποιήσουμε για την αποθήκευση, που κατά την γνώμη μας με βάση τις ιδέες που κυριαρχούν σε αυτό τον τομέα είναι αυτή που ταιριάζει πιο πολύ.

2.1 Περιγραφή του προβλήματος και των δεδομένων

Σε αυτή την υποενότητα παρουσιάζεται η μορφή των δεδομένων που έχουμε να αποθηκεύσουμε. Για να γίνει πιο κατανοητή η περιγραφή της μορφής

των δεδομένων, το παράδειγμα των δεδομένων που χρησιμοποιείται είναι μετεωρολογικά δεδομένα που προκύπτουν από μοντέλο μετεωρολογικής πρόγνωσης καθώς και επίσης δεδομένα από μετεωρολογικούς σταθμούς οι οποίοι είναι εγκατεστημένοι στην περιοχή της Ηπείρου και της Δυτικής Μακεδονίας.

Δεδομένα από μοντέλο WRF (Weather Research and Forecasting). Το μοντέλο WRF αποτελεί ένα αριθμητικό μοντέλο πρόγνωσης του καιρού σχεδιασμένο τόσο για την χρήση στην έρευνα όσο και στην υπηρεσιακή χρήση από εφαρμογές πρόγνωσης καιρού [13]. Το μοντέλο παρέχει μια σειρά από δεκάδες αποτελέσματα πάνω σε μετρικές που έχουν να κάνουν με την πρόγνωση του καιρού όπως θερμοκρασία, ταχύτητα και κατεύθυνση ανέμου, υετό , πίεση κ.α.

Αρχικά, θα πρέπει να αναλύσουμε τα στοιχεία κλειδιά του μετεωρολογικού μοντέλου τα οποία επηρεάζουν όλη την σχεδίαση και την υλοποίηση. Κάθε σημείο γεωγραφικού ενδιαφέροντος συγκαταλέγεται σε ένα ευρύτερο πλέγμα, το οποίο καλείται πεδίο (domain) μετεωρολογικής πρόγνωσης π.χ. πεδίο της Ελλάδας, πεδίο της Ηπείρου. Κάθε σημείο αποτελείται από τις συντεταγμένες του και τα διανύσματα, τα οποία περιέχουν τις τιμές για κάθε μετεωρολογική παράμετρο.

Δεδομένα από μετεωρολογικούς σταθμούς. Σημαντική πρόκληση για την συγκεκριμένη εργασία αποτέλεσε και η διαχείριση δεδομένων από μετεωρολογικούς σταθμούς της περιοχής. Πριν προχωρήσουμε όμως στην περιγραφή της σχεδίασης για την αποθήκευση τους μείζονα ρόλο παίζει η ανάλυση της δομής τους.

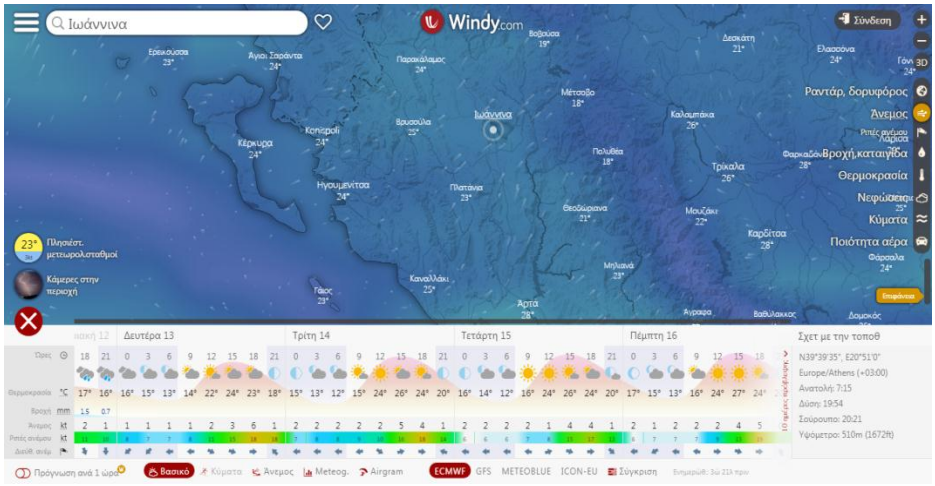
Η ιδέα είναι η εξής: σε πολλά σημεία στην περιοχή της Ηπείρου υπάρχουν γύρω στους 30 μετεωρολογικούς σταθμούς δημόσια προσιτούς όσον αφορά τα δεδομένα. Αξίζει να αναφερθεί ότι τα πραγματικά δεδομένα από τους μετεωρολογικούς σταθμούς αποτελούν πάντα σημαντική πληροφορία για μία περιοχή. Σαν απλός χρήστης κάποιος μπορεί να έχει μία εικόνα για τις καιρικές συνθήκες που επικρατούν σε μια περιοχή αλλά σε μετεωρολογικά μοντέλα μπορούν να χρησιμοποιηθούν ως ανατροφοδότηση ή αρχικοποίηση του μοντέλου.

2.2 Περιγραφή της τελικής εφαρμογής

Στο σημείο αυτό περιγράφουμε την δομή της διαδικτυακής εφαρμογής που σχεδιάζουμε η οποία θα καθορίσει την σχεδίαση της αποθηκευτικής δομής. Πολλές διαδικτυακές εφαρμογές για την πρόγνωση του καιρού έχουν αναπτυχθεί τα τελευταία χρόνια.

2.2.1 Διαδικτυακές εφαρμογές με χρονοσειρές και γεωχωρικά δεδομένα

Σε αυτή την υποενότητα θα περιγράψουμε τις ιδιότητες μερικών από των πιο δημοφιλών εφαρμογών που υπάρχουν στο διαδίκτυο και αποτελούν πηγή πληροφοριών για εκατομμύρια χρήστες. Τέτοιες εφαρμογές είναι windy[8], windfinder[12] κ.α.



Σχήμα 2.1: Στιγμιότυπο από την εκτέλεση της διαδικτυακής εφαρμογής windy.com

Στο παραπάνω σχήμα παρατηρούμε την διαδικτυακή εφαρμογή windy.com [8], όπως φαίνεται από περιηγητή διαδικτύου. Χαρακτηριστικό αυτής της εφαρμογής και γενικά όλων των παρόμοιων εφαρμογών είναι ότι αποτελούν εφαρμογές ιστού και είναι άμεσα προσιτές από εκατομμύρια χρήστες ανά τον κόσμο. Τελευταία βλέπουμε ότι ακόμα και στα ειδησεογραφικά δελτία, στο τμήμα εκείνο που αναφέρονται στην πρόγνωση του καιρού πλέον χρησιμοποιούν τέτοιου είδους εφαρμογές.

Ποια είναι όμως η διαφορά αυτών σε σχέση με προηγούμενους τρόπους ενημέρωσης πάνω στις καιρικές συνθήκες; Η διαφορά έγκειται στο ότι οι εφαρμογές αυτές αποτελούν δυναμικά και διαδραστικά συστήματα ενώ στην άλλη περίπτωση τα δεδομένα είναι στατικά και ο χρήστης δεν έχει καμία απολύτως διάδραση με τις εφαρμογές αυτές. Ο λόγος που δεν είναι διαδραστικές οι προηγούμενες εφαρμογές είναι η ταχύτητα διάδρασης και άμεσης απόκρισης. Αν δηλαδή, για μια λειτουργία που απαιτεί ένας χρήστης η εφαρμογή αποκρίνεται σε χρόνο μεγαλύτερο από αυτόν που είναι ανεκτός (π.χ. 100ms) τότε η εμπειρία χρήσης γίνεται πολύ αργή και κακή και ο χρήστης δεν μπορεί πλέον να χρησιμοποιεί μία τέτοια εφαρμογή. Υπάρχουν διάφοροι λόγοι που μία εφαρμογή δεν μπορεί να χρησιμοποιηθεί λόγω της αργής απόκρισης. Μπορεί να είναι τόσο η τεχνολογία που χρησιμοποιείται αλλά το σημαντικότερο που αντλείται μέσα από αυτή την εργασία είναι η δομή της αποθηκευτικής βάσης δεδομένων.

Επιπλέον, ένα κύριο χαρακτηριστικό των εφαρμογών αυτών αποτελεί το γεγονός ότι χρησιμοποιούν βιβλιοθήκης τρίτων ανοιχτού κώδικα και δεν απαιτείται η συνδρομή σε κάποια υπηρεσία. Η κύρια βιβλιοθήκη που χρησιμοποιείται είναι η leaflet [11]. Η leaflet είναι μία βιβλιοθήκη ανοιχτού κώδικα σε JavaScript για διαδικτυακές εφαρμογές με διαδραστικούς χάρτες. Έχει μια μεγάλη ποικιλία χαρακτηριστικών, ιδιοτήτων και επιμέρους λειτουργιών την οποία την κάνουν πολύ σημαντική και κάθε προγραμματιστής που σχεδιάζει μια τέτοια εφαρμογή πρέπει να γνωρίζει. Σημαντική δύναμη για την χρήση της βιβλιοθήκης αυτής αποτελεί η ύπαρξη πολλών πρόσθετων δυνατοτήτων τα λεγόμενα plugins. Ενδεικτικά αναφέρουμε τα εξής: λειτουργία προσαρμοσμένης προβολής εικόνων πάνω σε χάρτη, λειτουργία μεγεθύνσεων με διάφορα πλήκτρα και πολλά άλλα.

Οπότε, συνοψίζοντας θα τονίζαμε ότι οι εφαρμογές έχουν τα εξής σημαντικά χαρακτηριστικά:

1. Γρήγορη απόκριση στην επιλογή του χρήστη
2. Ολοκληρωμένη εικόνα για τις συνθήκες που επικρατούν σε ένα σημείο
3. Καλαίσθητη απεικόνιση όλων των πληροφοριών
4. Χρήση βιβλιοθηκών ανοιχτού κώδικα.

2.3 Διαχείριση χρονοσειρών

Η ανάλυση των μεγάλων δεδομένων χρησιμοποιείται σήμερα με σκοπό να αποφέρει μία τεράστια γνώση σε μία πληθώρα από προβλήματα μεταξύ άλλων: έρευνα καρκίνου, διαχείριση κέντρου δεδομένων (datacenter). Οι διάφορες εφαρμογές για τις οποίες η ανάλυση των μεγάλων δεδομένων κρίνεται αναγκαία δεν υιοθετούν κάποιο γενικό μοντέλο ικανό να λύσει όλα τα προβλήματα αλλά κάθε πεδίο στο οποίο χρησιμοποιούνται τα μεγάλα δεδομένα απαιτεί να αναπτυχθούν διαφορετικοί τύποι αποθήκευσης και ανάλυσης των ερωτημάτων. Γενικά, όσον αφορά τους τύπους των μεγάλων δεδομένων υπάρχει ένα τεράστιο εύρος έρευνας και εφαρμογής τους. Ωστόσο, σε ένα πεδίο το οποίο θα μας απασχολήσει σε αυτή την εργασία δεν έχει την ανάλογη ερευνητική προσοχή και αυτό είναι: οι αριθμητικές χρονοσειρές [4].

Σχετικά με τα μεγάλα δεδομένα έχει γίνει περισσότερη εργασία πάνω στην ανάλυση τους από συστήματα επεξεργασίας ροής και εξόρυξης δεδομένων και όχι τόσο σχετικά με την αποθήκευση και διενέργεια ερωτημάτων πάνω στα αποθηκευμένα δεδομένα. Για παράδειγμα σύγχρονα συστήματα σχεσιακών βάσεων δεδομένων δεν απεικονίζουν εύκολα την χρήση των μοντέλων των μεγάλων δεδομένων. Για αυτό όπως αναφέρεται και σε συναφή εργασία αντί των βάσεων δεδομένων χρησιμοποιούνται τεχνικές κατά περίπτωσης (ad-hoc) με την χρήση αρχείων τα οποία διαβάζονται με MATLAB, R κ.α. αναδεικνύοντας προβλήματα επεκτασιμότητας και διαχειρισιμότητας. Καταλαβαίνει κανείς ότι το να γίνεται αποθήκευση δεδομένων σε αρχεία δεν είναι κάτι εύκολο, ούτε μπορεί να χρησιμοποιείται εφόσον το μοντέλο των δεδομένων έχει αλλάξει.

Για να αντιμετωπιστεί το ζήτημα της αποθήκευσης και της υποστήριξης συγκεκριμένων λειτουργιών πάνω σε χρονοσειρές θα πρέπει να αντιμετωπιστούν οι ανάγκες που προκύπτουν. Αρχικά, θα πρέπει να αποφασίσουμε αν θα ακολουθήσουμε ένα είδος αποθήκευσης σχεσιακών βάσεων δεδομένων ή μη σχεσιακών βάσεων δεδομένων. Οι σχεσιακές βάσεις δεδομένων συγκεντρώνουν ένα πλήθος από λειτουργίες και εγγυήσεις οι οποίες είναι μη αναγκαίες για αριθμητικές χρονοσειρές.

1. **Υποστήριξη για μη αριθμητικά ερωτήματα**. Εντολές-τελεστές που είναι αρκετά ακριβοί ως προς την υπολογιστική απόδοση των σχεσιακών ερωτημάτων όπως η εντολή LIKE και JOIN δεν είναι εφαρμόσιμες σε δεδομένα με χρονοσειρές. Συνήθως αυτές οι εντολές έχουν εφαρμογές σχετικές με ερωτήματα πάνω σε αλφαριθμητικές τιμές.
2. **Υποστήριξη για αυθαίρετες τροποποιήσεις σε υπάρχοντα δεδομένα.** Στις περισσότερες εφαρμογές τα δεδομένα μπορεί να έχουν πολλαπλές εκδόσεις και να απαιτείται ή να εφαρμόζεται τροποποίηση των δεδομένων για μία τιμή κλειδί σε αυτά τα δεδομένα. Αντίθετα, σε εφαρμογές με χρονοσειρές δεν απαιτείται τόσο έντονη χρήση των τροποποιήσεων πάνω σε τιμές χρονοσειρών γιατί τα δεδομένα είτε λαμβάνονται από αισθητήρες που οι τιμές είναι οι δεδομένες για την δεδομένη χρονική στιγμή και δεν αλλάζει κάτι μελλοντικά για παρελθοντικές τιμές.
3. **Υποστήριξη διαδικασίας διαγραφής τυχαίων τιμών χρονοσειρών.** Συνήθως οι τιμές χρονοσειρών διαγράφονται σε ένα εύρος μεγαλύτερο της μίας εγγραφής δηλαδή γίνεται διαγραφή σε ομάδες από εγγραφές συγκεκριμένου χρονολογικού εύρους.

Στην παραπάνω καταγραφή έγιναν σαφείς οι λειτουργίες των σχεσιακών βάσεων δεδομένων οι οποίες δεν είναι αναγκαίες για δεδομένα χρονοσειρών. Παρακάτω, θα δούμε στα συστήματα εξόρυξης δεδομένων ποιες λειτουργίες χρησιμοποιούνται συχνότερα σε τέτοιου είδους δεδομένα.

1. **Υποστήριξη λειτουργιών ερωτημάτων διαστήματος**. Κεντρική αρχή σε όλες σχεδόν τις μεθόδους εξόρυξης χρονοσειρών αποτελεί η χρησιμοποίηση μεθόδων και ερωτημάτων εύρους μεταξύ δύο σημείων – χρονοσειρών. Σαν αντιστοιχία στην εντολή LIKE της SQL μπορεί να χαρακτηριστεί η διαδικασία αναζήτησης ενός διαστήματος χρονοσειρών μέσα σε ένα άλλο μεγαλύτερο.
2. **Υποστήριξη προ-επεξεργασίας**. Μία μέθοδος που χρησιμοποιείται ευρέως πάνω σε οπτικοποίηση κυρίως τέτοιων δεδομένων είναι η προεπεξεργασία σε ένα σταθερό ρυθμό είτε αυτό αναφέρεται στο πεδίο της μίας ημέρας, είτε της μίας εβδομάδας είτε οποιοδήποτε σταθερό μέγεθος.

3. **Συμπίεση**. Σημαντική ιδιότητα που χρησιμοποιείται σε δεδομένα χρονοσειρών είναι η συμπίεση τους χωρίς να χάνεται οποιαδήποτε πληροφορία.

Σύμφωνα με τις απαιτήσεις, τις ανάγκες και τις ιδιότητες των χρονοσειρών ως προς την αποθήκευση και την ανάκτηση τους όπως προαναφέρθηκαν παραπάνω το μοντέλο τον μη-σχεσιακών βάσεων δεδομένων φαίνεται να ταιριάζει καλύτερα. Αποθηκεύοντας ταξινομημένες χρονοσειρές σε ξεχωριστές νοητές περιοχές-τμήματα (π.χ. ημέρα), αυτού του είδους οι βάσεις δεδομένων μπορούν να αποδώσουν πιο αποτελεσματικά σε ερωτήματα εύρους και συμπίεση δεδομένων από τα παραδοσιακά συστήματα σχεσιακών βάσεων δεδομένων. Οπότε, καταλήγουμε να θέσουμε ως υποψήφιο σύστημα για αποθήκευση χρονοσειρών τα συστήματα μη-σχεσιακών βάσεων δεδομένων.

2.4 Διαχείριση γεωγραφικών δεδομένων

Σε αυτή την υποενότητα θα εξετάσουμε το δεύτερο σκέλος του γενικότερου προβλήματος που έχουμε να αντιμετωπίσουμε και αυτό είναι η διαχείριση των γεωγραφικών δεδομένων για μια συγκεκριμένη περιοχή [5].

Το ζήτημα της διαχείρισης των δεδομένων γεωγραφικών σημείων έχει αναπτυχθεί ιδιαίτερα τα τελευταία χρόνια με την ανάπτυξη των τεχνολογιών διαδικτύου των πραγμάτων (Internet-Of-Things). Αισθητήρες, ρολόγια, κινητά τηλέφωνα και άλλες συσκευές έχουν δημιουργήσει ένα τέτοιο οικοσύστημα τα οποίο έχει σαν χαρακτηριστικό ότι τα παραγόμενα δεδομένα είναι γεωχωρικά από την φύση τους. Εφαρμογές οι οποίες χρησιμοποιούν τέτοιου είδους μεγάλα δεδομένα πρέπει να είναι ικανές να ανταπεξέρχονται σε ευρετηριοποίηση και εκτέλεση ερωτημάτων με αποδοτικότητα.

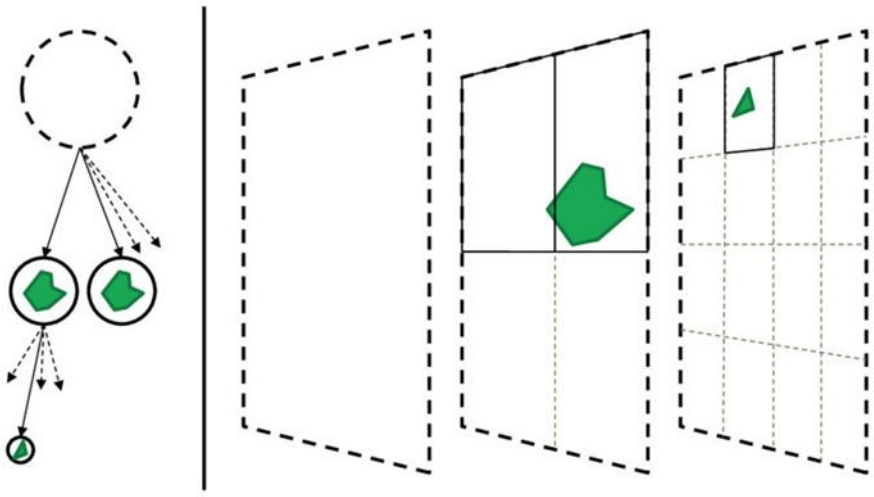
Στην δική μας εργασία δεν μας ενδιαφέρει τόσο η διαχείριση δεδομένων από συσκευές που είναι συνδεδεμένες σε κάποιο Internet-Of-Things αλλά παρόλα αυτά μας απασχολεί να αναλύσουμε τις τεχνικές που υιοθετούνται από τέτοια αρκετά κρίσιμα συστήματα γιατί αν μπορέσουμε να υιοθετήσουμε παρόμοιες τεχνικές μπορεί το σύστημα μας να γίνει εύκολα κλιμακώσιμο.

Παρόλο που τα σύγχρονα συστήματα αποθήκευσης διαθέτουν υποστήριξη για λειτουργίες για ισότητες και ερωτήματα εύρους πάνω σε δεδομένα, η υποστήριξη

για λειτουργίες πάνω σε γεωχωρικά δεδομένα όπως εκτέλεση γεωχωρικών ερωτημάτων πάνω σε πολύγωνα με τομές και ενώσεις είναι αρκετά περίπλοκο ζήτημα και απαιτεί πιο προχωρημένες τεχνικές. Επιπλέον χαρακτηριστικό σε σχέση με τα γεωχωρικά δεδομένα είναι ο ρυθμός παραγωγής δεδομένων από τις διάφορες πηγές. Ας σκεφτούμε ότι σε ένα μεγάλο δίκτυο υπηρεσίας καιρού μπορούν να παράγονται εκατομμύρια δεδομένα σε κάθε σάρωση οπότε ο ρυθμός παραγωγής-ταχύτητα των συσκευών αποτελεί ένα ζήτημα το οποίο αξίζει να αναλυθεί.

Επίσης, ένα άλλο χαρακτηριστικό που πρέπει να λαμβάνεται υπόψη σε τέτοια δεδομένα είναι η αναγκαιότητα να παρέχουμε σε μια εφαρμογή όσο πιο νέα δεδομένα από τους αισθητήρες ούτως ώστε το σύστημα να έχει τα καλύτερα δυνατά αποτελέσματα. Για παράδειγμα, συστήματα που έχουν να κάνουν με αποκατάσταση καταστροφών μπορεί να έχουν κάποια μειονεκτήματα αν τα δεδομένα δεν είναι ανανεωμένα.

Επομένως, τα τρία αυτά χαρακτηριστικά: γεωχωρικά δεδομένα, πλήθος – ταχύτητα-συχνότητα δεδομένων και κρισιμότητα στην παροχή των πιο «φρέσκων» δεδομένων συντελούν να έχουμε πολύ αυστηρές προϋποθέσεις και απαιτήσεις στην σχεδίαση ενός τέτοιου συστήματος.



Σχήμα 2.2: Μέθοδος ευρετηριοποίησης με βάση δέντρα περιοχών

2.4.1 Ευρετηριοποίηση

Ως λύση που προτείνεται σε σύγχρονες εργασίες στον τομέα των γεωχωρικών δεδομένων για αποθήκευση και άμεση ανάκτηση είναι η υλοποίηση ευρετη-

ρίων τα οποία να απαντούν στις προκλήσεις που αναφέρονται παραπάνω. Ο ρόλος των γεωχωρικών ευρετηρίων είναι να οργανώσουν δύο διαστάσεων ή και πιο πολλών διαστάσεων αντικείμενα με στόχο την άμεση εξυπηρέτηση των ερωτημάτων. Ο τρόπος για να επιτευχθεί αυτό γενικά είναι να ορίζονται ομάδες αντικειμένων οι οποίες ομαδοποιούν με κάποιο τρόπο τα δεδομένα.

Στο Σχήμα 2.2 παρατηρούμε τον τρόπο με τον οποίο το σύστημα SIFT υλοποιεί την ευρετηριοποίηση των γεωχωρικών δεδομένων. Στην αριστερή πλευρά παρατηρούμε το δέντρο το οποίο δημιουργείται κατά την διάρκεια της εισαγωγής των δεδομένων στο ευρετήριο. Στην δεξιά πλευρά παρατηρούμε τα δομικά στοιχεία στα οποία μοντελοποιείται το δέντρο της αριστερής πλευράς. Ο κόμβος ρίζα είναι το πεδίο όλης της περιοχής ανάλυσης. Το δεύτερο επίπεδο από κόμβους είναι οι υποπεριοχές στις οποίες χωρίζεται ο κόμβος ρίζα. Έτσι σημεία τα οποία είναι διάσπαρτα μπορούν να θεωρηθούν ως υποομάδα μιας γεωγραφικής υποπεριοχή.

Για αποτελεσματικότερη αναζήτηση γειτονικών σημείων ένα γεωχωρικό ευρετήριο οργανώνει αυτές τις υποομάδες με τέτοιο τρόπο ούτως ώστε σημεία που είναι κοντά στο πεδίο μεταξύ τους είναι και γειτονικά στο ευρετήριο. Το πιο δημοφιλές ευρετήριο είναι τα δέντρα (Trees) τα οποία ιεραρχικά ομαδοποιούν τα σημεία σε περιοχές.

2.5 Συστήματα Βάσεων Δεδομένων κατάλληλων για χρονοσειρές και γεωχωρικά δεδομένα

Σε αυτή την υποενότητα θα δώσουμε κάποιο σημαντικό υπόβαθρο σχετικά με διαθέσιμα συστήματα βάσεων δεδομένων τα οποία θα μπορούσαν να φανούν σαν διαθέσιμες επιλογές σχετικά με την αποθήκευση των δεδομένων που έχουμε να αποθηκεύσουμε.

Στις Ενότητα 2.3 και 2.4 παρουσιάσαμε μερικές ιδιότητες που αποτελούν βάση για την επιλογή του κατάλληλου συστήματος αποθήκευσης. Παρακάτω θα δούμε μερικές από τις διαθέσιμες επιλογές που εκπληρώνουν τις ανάγκες που προκύπτουν από τα χρονοχωρικά δεδομένα.

2.5.1 Το σύστημα διαχείρισης δεδομένων TimescaleDB

Μια διαθέσιμη επιλογή για την αποθήκευση χρονοσειρών κυρίως αποτελεί το σύστημα TimescaleDB [14]. Η TimescaleDB είναι ένα σύστημα βάσεων δεδομένων ανοιχτού κώδικα το οποίο επιτρέπει να γίνει η SQL κλιμακώσιμη για χρονοσειρές. Έχει αναπτυχθεί από την PostgresSQL και αποτελεί προσθήκη της. Παρέχει την πολύ σημαντική δυνατότητα της αυτόματης κατάτμησης στον χρόνο.

Η TimescaleDB επεκτείνει την PostgresSQL για χρονοσειρές παρόλα αυτά διατηρεί την κύρια διεπαφή της PostgresSQL. Πιο συγκεκριμένα, η TimescaleDB με πρώτη ματιά δεν είναι τίποτα άλλο από μια συλλογή από συνηθισμένους πίνακες που βλέπουμε συχνά στις βάσεις δεδομένων, αλλά στην πράξη αυτοί οι πίνακες είναι από μόνοι τους μια συλλογή από επιμέρους πίνακες που συνθέτουν τα δεδομένα. Με άλλα λόγια κάθε διάσταση αποτελεί έναν υποπίνακα, οι οποίοι περιλαμβάνονται σε αυτό που καλείται υπερπίνακας (hypertable).

Αυτό που φαίνεται στον τελικό χρήστη της διεπαφής του συστήματος αυτού είναι ο υπερπίνακας στον οποίο διενεργούνται όποιες λειτουργίες απαιτεί ο χρήστης όπως: εισαγωγή των δεδομένων, διαγραφή των δεδομένων, υποβολή ερωτήματος.

2.5.2 Το σύστημα διαχείρισης δεδομένων MongoDB

Μια άλλη εναλλακτική επιλογή για την αποθήκευση χρονοχωρικών δεδομένων αποτελεί το σύστημα βάσεων δεδομένων της MongoDB το οποίο κατηγοριοποιείται ως σύστημα NoSQL βασισμένο στην ιδέα αποθήκευσης κλειδιού-τιμής. Το σύστημα MongoDB είναι μια βάση δεδομένων εγγράφων που βασίζεται σε μια αρχιτεκτονική κλιμάκωσης που έχει γίνει δημοφιλής στους προγραμματιστές όλων των ειδών που δημιουργούν κλιμακούμενες εφαρμογές χρησιμοποιώντας ευέλικτες μεθοδολογίες [1].

Η αρχιτεκτονική βασισμένη σε εγγραφές (documents) αποτελεί μια εξαιρετικά ευέλικτη τεχνική η οποία επιτρέπει να έχουμε όποια μορφή δεδομένων αποθηκευμένα με τρόπο που μπορεί να μην είναι ολοκληρωμένα μεταξύ τους τα έγγραφα. Υπάρχει μια αντιστοίχιση μεταξύ του συστήματος αυτού και των σχεσια-

κών συστημάτων βάσεων δεδομένων. Τα πεδία των εγγραφών της MongoDB είναι κατ'αντιστοιχία οι στήλες των πινάκων σε σχεσιακές βάσεις δεδομένων.

Σημαντικό πλεονέκτημα του συστήματος αυτού είναι η ύπαρξη ειδικού ευρετηρίου για γεωχωρικά δεδομένα, παρόμοιο με αυτό που περιγράφεται σε σύγχρονες εργασίες και τονίζουν την ιδέα της κατάτμησης των σημείων σε υποομάδες ανά περιοχή.

Η MongoDB είναι μια έγγραφο-κεντρική βάση δεδομένων το οποίο σημαίνει ότι τα δεδομένα αποθηκεύονται σε μορφή JSON.

ΚΕΦΑΛΑΙΟ 3

Σχεδιαση

3.1 Οφέλη του συστήματος της MongoDB

3.2 Σχεδίαση του αποθηκευτικού χώρου για το μετεωρολογικό μοντέλο

3.3 Σχεδίαση του αποθηκευτικού χώρου για τα δεδομένα από μετεωρολογικούς αισθητήρες

Σε αυτό το κεφάλαιο ενότητα θα περιγράψουμε την επιλογή του συστήματος βάσεων δεδομένων που επιλέγουμε, την σχεδίαση του κάθε υποσυστήματος ξεχωριστά και την ενοποίηση του για την τελική εφαρμογή.

3.1 Οφέλη του συστήματος της MongoDB

Η υποενότητα αυτή περιλαμβάνει την καταγραφή εκείνων των ιδιοτήτων που κατέστησαν το σύστημα της MongoDB ως το καταλληλότερο για μια διαδικτυακή εφαρμογή με χρονοχωρικά δεδομένα. Μελετώντας τα οφέλη της MongoDB σε ένα υψηλό επίπεδο μπορούμε να πούμε ότι την καθιστούν μια πολύ πιθανή επιλογή. Ωστόσο, η πειραματική αξιολόγηση που έγινε συγκριτικά με τα άλλα συστήματα που προτείνοντα, έδειξαν ότι είναι η καλύτερη επιλογή από άποψη απόδοσης. Τα αποτελέσματα των πειραμάτων τα βλέπουμε στην ενότητα 5.3.

Η MongoDB μας προσφέρει ένα πλήθος από ιδιότητες οι οποίες την καθιστούν επίσης κατάλληλη για χρήση σε συστήματα που αλληλεπιδρούν με διαδικτυακές εφαρμογές. Πιο συγκεκριμένα, οι ιδιότητες της για χρονοσειρές συμπυκνώνονται στα παρακάτω:

- Υποστήριξη ερωτημάτων εύρους

- Δυνατότητα προ-επεξεργασίας
- Υποστήριξη συμπίεση χωρίς απώλειες στην απόδοση

Και όσον αφορά τα γεωχωρικά δεδομένα επιλύει τις ιδιότητες που τέθηκαν:

- Συχνότητα των πηγών και των τροποποιήσεων των δεδομένων
- Άμεση απόκριση σε «φρέσκα δεδομένα»

Επίσης, στην MongoDB τα δεδομένα αποθηκεύονται σε μορφή JSON. Το γεγονός αυτό αποτελεί από μόνο σημαντικό πλεονέκτημα διότι οι περισσότερες διαδικτυακές εφαρμογές (με API) απαιτούν τα δεδομένα να επιστρέφονται σε αυτή τη μορφή. Αυτός ο τρόπος πιστεύουμε είναι ο πιο φυσιολογικός να αποθηκευτούν τα δεδομένα από το να αποθηκευτούν σε παραδοσιακά συστήματα βάσεων δεδομένων που στηρίζονται στο πρότυπο γραμμή/στήλη. Τα παραδοσιακά συστήματα σε κάθε προσθήκη ενός πεδίου απαιτούν μια διαδικασία από τον σχεδιαστή της βάσης δεδομένων να προσθέσει και ένα πεδίο στον εκάστοτε πίνακα. Αντίθετα, στα μη-σχεσιακά συστήματα η παραπάνω ανάγκη δεν υπάρχει οπότε ο σχεδιαστής προσθέτει το κάθε πεδίο όποτε η πηγή το παρέχει στο σύστημα.

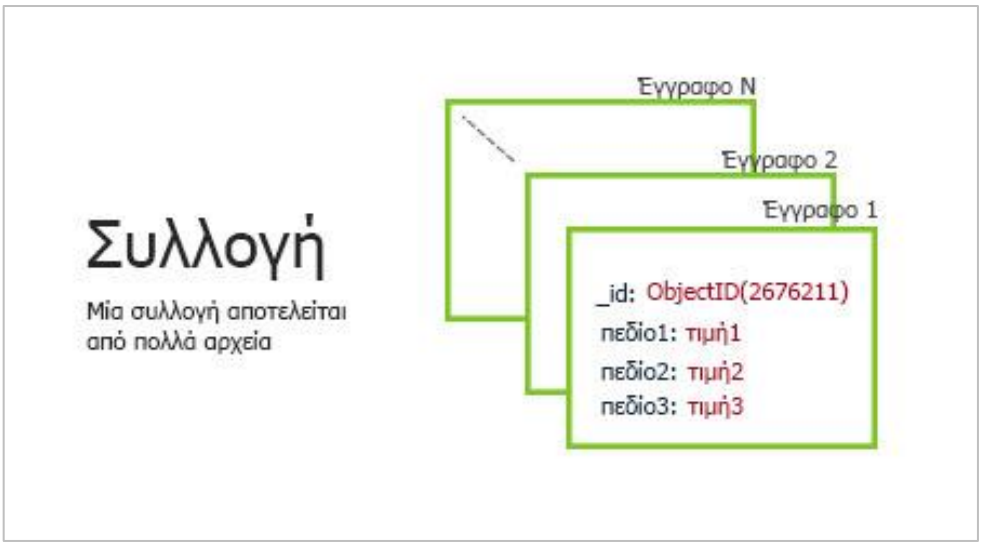
Για παράδειγμα σε ένα σύστημα MySQL ενδέχεται να έχουμε ένα πίνακα με A1, A2, A3, A4… πεδία και όλες οι εγγραφές να έχουν δεδομένα σε αυτά πεδία του πίνακα. Αντίθετα, στο NoSQL σχήμα ενδέχεται σε κάποιες εγγραφές να έχουμε τιμές στα πεδία A1, A2, A3, A4, B1, B6 τα οποία δεν ορίζονται εξ' αρχής στο σχήμα.

Παρακάτω παραθέτουμε μερικές από τις ιδιότητες της MongoDB οι οποίες μας παρέχουν ένα πού δυνατό εργαλείο για την επίλυση των ζητημάτων που έχουμε να αντιμετωπίσουμε.

- Έγγραφο-κεντρική (document-oriented): διαχείριση των δεδομένων σε ελεύθερη και ευέλικτη μορφή
- Ερωτήματα για συγκεκριμένο σκοπό(ad-hoc queries): η MongoDB επιτρέπει αναζήτηση με βάση κάποιο πεδίο, ερωτήματα εύρους και κανονικές εκφράσεις και να επιστραφούν συγκεκριμένα πεδία ως αποτέλεσμα.
- Ευρετηριοποίηση (Indexing): επιτρέπει την ευρετηριοποίηση σε οποιοδήποτε πεδίο του σχήματος

- Αντίγραφα (Replication): η MongoDB έχει υψηλή διαθεσιμότητα των δεδομένων μιας και κρατάει σύνολα από αντίγραφα. Κάθε αντίγραφο έχει τον ρόλο.

Μπορεί κάποιο αντίγραφο να θεωρηθεί το κύριο και το οποίο να είναι στον κεντρικό εξυπηρετητή και να αλληλεπιδράει με το σύστημα πελάτη. Τα δευτερεύοντα αντίγραφα διατηρούν ένα αντίγραφο του κύριου μέρους. Με αυτό τον τρόπο αν αποτύχει το κεντρικό τότε τα δευτερεύοντα έχουν ένα διαθέσιμο αντίγραφο, εξασφαλίζοντας υψηλή διαθεσιμότητα στην εφαρμογή-πελάτη.



Σχήμα 3.1: Δομή των δεδομένων στην MongoDB

Στο Σχήμα 3.1 παρουσιάζουμε την δομή που έχει μια συλλογή στην MongoDB με μεγαλύτερη λεπτομέρεια. Μια συλλογή αποτελείται από έγγραφα (documents). Κάθε εγγραφή με τη σειρά της αποτελείται από τα πεδία και τις τιμές των πεδίων αυτών. Το σημαντικό στα γεωχωρικά δεδομένα είναι ότι μπορούμε να ορίσουμε πάνω από την συλλογή ένα ευρετήριο το οποίο ονομάζεται 2dsphere στη MongoDB. Το ευρετήριο αυτό υποστηρίζει ερωτήματα τα οποία υπολογίζουν γεωμετρίες σε μορφή παρόμοιες με τη γη (σφαίρα). Επιτρέπει ερωτήματα όπως: συμπερίληψης (ένα σημείο αν περιλαμβάνεται σε μια περιοχή), τομής και γειτονικότητας. Τα ερωτήματα είναι τα πιο κρίσιμα στην σχεδίαση και την υλοποίηση της αποθηκευτικής δυνατότητας και της τελικής εφαρμογής.

Με βάση τις παραπάνω ιδιότητες και την πειραματική αξιολόγηση τελικά κρίθηκε ότι το σύστημα αυτό θα αποτελέσει την βάση πάνω στην οποία θα οικοδομηθεί όλη η σχεδίαση και υλοποίηση.

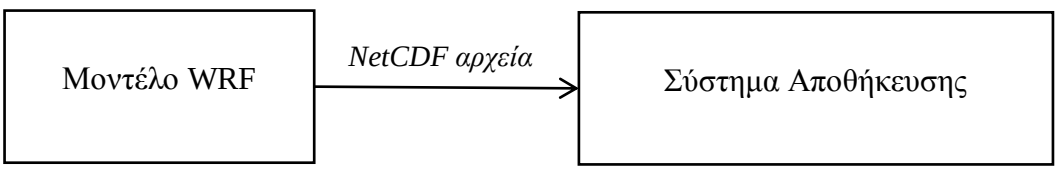
3.2 Σχεδίαση του αποθηκευτικού χώρου για το μετεωρολογικό μοντέλο

Στην υποενότητα αυτή θα περιγράψουμε με περισσότερες λεπτομέρειες τα διάφορα χαρακτηριστικά του μετεωρολογικού μοντέλου WRF, την δομή των δεδομένων και τα κύρια στοιχεία.

Το μοντέλο WRF αποτελεί ένα αριθμητικό μοντέλο πρόγνωσης του καιρού σχεδιασμένο τόσο για την χρήση στην έρευνα όσο και στην υπηρεσιακή χρήση από εφαρμογές πρόγνωσης καιρού. Το μοντέλο παρέχει μια σειρά από δεκάδες αποτελέσματα πάνω σε μετρικές που έχουν να κάνουν με την πρόγνωση του καιρού όπως θερμοκρασία, ταχύτητα και κατεύθυνση ανέμου, υετό, πίεση κ.α.

Τα δεδομένα της μετεωρολογικής πρόγνωσης παρέχονται στο Εργαστήριο Υπολογιστικών Συστημάτων από το Εργαστήριο Μετεωρολογίας του τμήματος Φυσικής του Πανεπιστημίου Ιωαννίνων. Μέσω μίας κοινής υπολογιστικής υποδομής στο Εργαστήριο Μετεωρολογίας του τμήμα Φυσικής και κάποια αντίγραφα από τα αρχικά δεδομένα στεγάζονται σε υποδομές του Εργαστηρίου Συστημάτων Υπολογιστών – μας επιτρέπει την επεξεργασία των δεδομένων αυτών με σκοπό την καλύτερη διαχείριση τους ώστε να επιτευχθούν όλοι οι στόχοι που τέθηκαν.

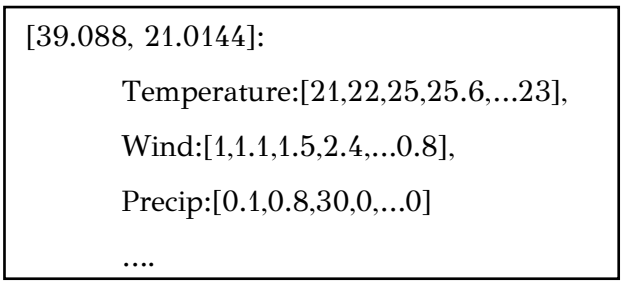
Αφού ολοκληρωθεί το στάδιο της επεξεργασίας των δεδομένων μέσω του μοντέλου WRF και εξαχθούν όλα τα δεδομένα της πρόγνωσης, τότε μία ρουτίνα αναλαμβάνει να εισάγει όλη την πληροφορία στο επιμέρους σύστημα αποθήκευσης.



Σχήμα 3.2: Επικοινωνίας μοντέλου πρόγνωσης και συστήματος αποθήκευσης

Τα δεδομένα που εξάγει το μετεωρολογικό μοντέλο αποθηκεύονται σε προσωρινά αρχεία τύπου NetCDF (Network Common Data Form) [15]. Τα αρχεία αυτά με τη σειρά τους επιβάλλονται σε επεξεργασία και εισάγονται εν τέλει στις τελικές δομές. Το NetCDF είναι ένα σύνολο βιβλιοθηκών λογισμικού και μορφών δεδομένων αυτο-περιγραφόμενων, ανεξάρτητων από μηχανήματα που υποστηρίζουν τη δημιουργία, πρόσβαση και κοινή χρήση επιστημονικών δεδομένων προσανατολισμένων στον πίνακα. Το αρχείο NetCDF είναι μια μορφή αρχείου για την αποθήκευση πολυδιάστατων επιστημονικών δεδομένων (μεταβλητές) όπως η θερμοκρασία, η υγρασία, η πίεση, η ταχύτητα του ανέμου και η κατεύθυνση.

Επομένως, για κάθε πεδίο ενδιαφέροντος π.χ. Ελλάδα, το μοντέλο WRF εξάγει ένα αρχείο το οποίο έχει την εξής δομή: Για κάθε σημείο που ανήκει στο πεδίο ενδιαφέροντος υπάρχει μία εγγραφή για το σημείο αυτό. Η κάθε εγγραφή έχει την εξής δομή: για κάθε μετεωρολογική παράμετρο υφίστανται πολλαπλά διανύσματα. Το κάθε διάνυσμα περιλαμβάνει την αριθμητική τιμή για κάθε χρονική στιγμή για την οποία εξάγουμε αποτελέσματα. Για παράδειγμα, έστω το σημείο ενδιαφέροντος είναι το [39.088, 21.0144].



Σχήμα 3.3: Δομή αρχείου NetCDF

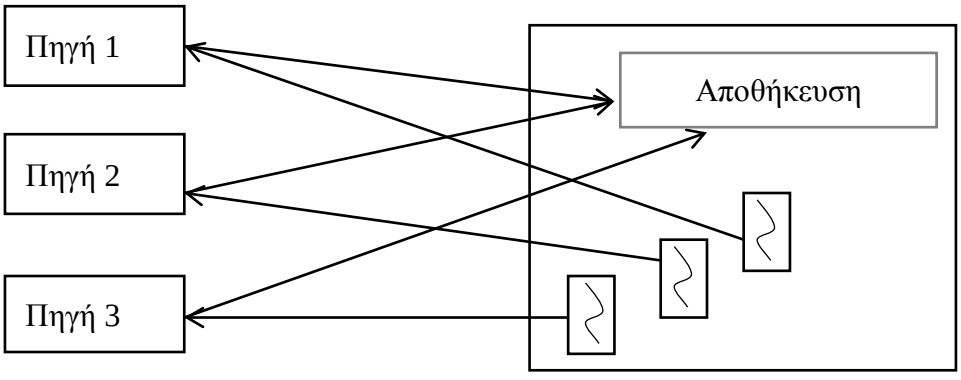
Στο Σχήμα 3.3 παρατηρούμε το αρχείο πηγή για το σύστημα μας. Στο συγκεκριμένο μοντέλο που χρησιμοποιούμε στην εργασία, οι παράμετροι έχουν χρονική απόσταση 2 ωρών με αρχική χρονική στιγμή τις 02:00 UTC. Στο σχήμα 3.3 παρατηρούμε ότι έχουμε 21 βαθμούς στις 02:00 UTC, 22 στις 04:00UTC και 25 στις 06:00 UTC.

Κάθε διάνυσμα μιας μετεωρολογική παράμετρος έχει 36 τιμές από το οποίο συμπεραίνουμε ότι αυτό το μοντέλο μας εξάγει μια πρόγνωση για τις επόμενες 3 ημέρες.

Το βασικό στοιχείο στα δεδομένα αυτά αποτελεί η πυκνότητα του πλέγματος και το πλήθος των γεωγραφικών σημείων. Πιο συγκεκριμένα, αν για παράδειγμα μελετάμε το πεδίο της Ελλάδας τότε το πλέγμα που εμφανίζεται είναι της τάξης των 35000 σημείων ενώ αντίστοιχα για το πεδίο της Ηπείρου και της Δυτικής Μακεδονίας 21000 σημεία. Επίσης, η πυκνότητα των σημείων στην Ελληνική Επικράτεια είναι στα 6x6 χιλιόμετρα ενώ αντίστοιχα για την Ήπειρο είναι 2χλμ x 2χλμ. Αυτό πρακτικά σημαίνει ότι το ένα σημείο ενδιαφέροντος από το άλλο έχει απόσταση 6 και 2 χιλιόμετρα αντίστοιχα.

Επομένως, το μοντέλο πρόγνωσης του καιρού WRF παράγει τρισδιάστατα δεδομένα: το γεωγραφικό μήκος, το γεωγραφικό πλάτος και τον χρόνο.

Δεδομένα από μετεωρολογικούς σταθμούς. Σημαντική πρόκληση για την συγκεκριμένη εργασία αποτέλεσε και η διαχείριση δεδομένων από μετεωρολογικούς σταθμούς της περιοχής. Πριν προχωρήσουμε όμως στην περιγραφή της σχεδίασης για την αποθήκευση τους μείζονα ρόλο παίζει η ανάλυση της δομής τους. Η ιδέα είναι η εξής: σε πολλά σημεία στην περιοχή της Ηπείρου υπάρχουν γύρω στους 30 μετεωρολογικούς σταθμούς δημόσια προσιτούς όσον αφορά τα δεδομένα.



Σχήμα 3.4: Σχήμα επικοινωνίας δεδομένων από μετεωρολογικούς σταθμούς και συστήματος αποθήκευσης

Στο Σχήμα 3.4 παρουσιάζουμε σε υψηλό επίπεδο την διαδικασία εξαγωγής των δεδομένων. Να τονιστεί ότι ειδικές ρουτίνες αναλαμβάνουν να ανακτήσουν από διαθέσιμες πηγές τα απαιτούμενα δεδομένα. Οι διαθέσιμες πηγές είναι 3 τύπων όπως και οι ρουτίνες που τα εξάγουν:

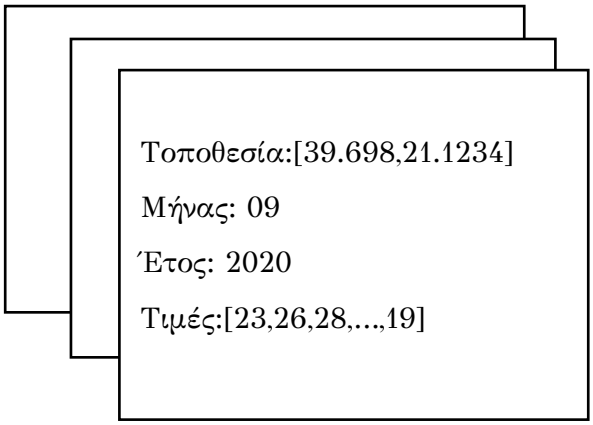
- Τύπος Α: Δεδομένα από αισθητήρες προσβάσιμους εσωτερικά από την ομάδα Μετεωρολογίας
- Τύπος Β: Δεδομένα από αισθητήρες προσβάσιμους μέσω βιβλιοθήκης κλήσεων API
- Τύπος Γ: Δεδομένα από αισθητήρες δημόσιας πρόσβασης

Η δομή των δεδομένων των σταθμών διαφέρει ως προς την δομή των δεδομένων της πρόγνωσης. Πιο συγκεκριμένα, κάθε σταθμός για μία χρονική στιγμή προβάλει στο σύστημα μας τις τρέχουσες καιρικές συνθήκες. Η προβολή της πληροφορίας αυτής γίνεται με κάποιας μορφής ανάκτησης από δημόσια προσιτούς ιστοτόπους. Επομένως, για κάθε 30 λεπτά ανακτούμε από τους σταθμούς αυτούς μία τιμή για κάθε μετεωρολογική παράμετρο που αντλούν οι αισθητήρες (θερμοκρασία, άνεμος, βροχόπτωση, πίεση κ.ο.κ.). Η διαφορά στα συγκεκριμένα δεδομένα με της πρόγνωσης έγκειται στο ότι οι μετεωρολογικοί σταθμοί σαν σημεία στον χώρο είναι πολύ λιγότερα από το πλήθος των σημείων του πλέγματος αλλά η πυκνότητα των δεδομένων σε μία ημέρα είναι 4-πλάσια σε σχέση με το μετεωρολογικό μοντέλο.

Επομένως, καταλαβαίνουμε ότι το πλήθος των δεδομένων από τα δύο παραπάνω μοντέλα φτάνει σε μεγάλο όγκο και η διαχείριση και των δύο αυτών μορφών δεδομένων αποτελεί σημαντική πρόκληση για την εργασία αυτή.

3.2.1 Απλή αποθήκευση μετεωρολογικών δεδομένων

Πρώτη ιδέα για την αποθήκευση των δεδομένων από τους μετεωρολογικούς σταθμούς είναι να αποθηκεύσουμε κάθε μετεωρολογική παράμετρο σε ξεχωριστές συλλογές-πίνακες στην βάση δεδομένων. Για παράδειγμα, θα έχουμε μία συλλογή για την θερμοκρασία για τον άνεμο κ.ο.κ. Σε κάθε εκτέλεση του μετεωρολογικού μοντέλου οι νέες τιμές που θα προέκυπταν θα τοποθετούνταν στον κάθε πίνακα ως συνέχεια. Μια εγγραφή στον πίνακα αυτό έχει την δομή του σχήματος 3.5:



Σχήμα 3.5: Δομή απλής αποθήκευσης μετεωρολογικών δεδομένων

Ο λόγος που χρησιμοποιούμε τιμές για έναν μήνα ολόκληρο είναι για να δώσουμε μια μορφή ομαδοποίησης στα δεδομένα. Θα μπορούσαν τα δεδομένα να ήταν αποθηκευμένα ανά ημέρα αλλά αυτή η προσέγγιση θα ήταν πολύ απλοϊκή και δεν θα βοηθούσε στο να έχουμε πολλά δεδομένα εύκολα ανακτήσιμα με ένα απλό ερώτημα στην βάση.

Αυτή η πρώτη εκδοχή έχει κάποια πλεονεκτήματα:

- Εύκολη και γρήγορη εισαγωγή στην βάση δεδομένων μετά από την εκτέλεση του μετεωρολογικού μοντέλου
- Ομαδοποίηση των δεδομένων με βάση τον μήνα

Ας σκεφτούμε όμως το εξής σενάριο: κάποιος χρήστης της εφαρμογής απαιτεί να λάβει πληροφορίες για ένα γεωγραφικό σημείο για μία συγκεκριμένη μελλοντική στιγμή. Για παράδειγμα, κατά την ημερομηνία 29 Ιουλίου 2021 κάποιος χρήστης επιλέγει να ερευνήσει τις καιρικές συνθήκες για τις 30 Ιουλίου 2021 και ώρα 12 το μεσημέρι στο γεωγραφικό σημείο [39.1222, 21.4555].

Για να γίνει αυτό εφικτό πρέπει να εκτελεστούν πολλαπλά ερωτήματα στην βάση για κάθε μετεωρολογική παράμετρο που εμφανίζουμε στην εφαρμογή. Θα μπορούσε κάποιος να πει ότι εφόσον κάποιος ενδιαφέρεται για την θερμοκρασία θα εκτελέσουμε απλά ένα ερώτημα στον πίνακα με την θερμοκρασία. Με αυτό τον τρόπο ναι μεν κάνουμε εύχρηστη την λειτουργία αλλά χάνουμε μεγάλη πληροφορία για το σημείο αυτό αφού ταυτόχρονα με την θερμοκρασία μας ενδιαφέ-

ρει συνήθως να δούμε και άλλες παραμέτρους για να λάβουμε επαρκή πληροφορία για το σημείο αυτό.

Επομένως, θα χρειαστεί να κάνουμε διαδοχικά ερωτήματα σε κάθε πίνακα στην βάση δεδομένων. Εκτός αυτού, η ομαδοποίηση των δεδομένων που κάνουμε ανά μήνα ενός έτους βοηθάει μόνο όταν χρειαζόμαστε ιστορικά στοιχεία για ένα σημείο. Όταν απαιτεί ο χρήστης μόνο μία συγκεκριμένη χρονική στιγμή τότε τα πράγματα γίνονται λίγο περίπλοκα με αυτή την προσέγγιση. Ας σκεφτούμε το εξής παράδειγμα για να δούμε πόσο δύσκολη μπορεί να γίνει η ανάκτηση μιας τέτοιας τιμής.

Για διευκόλυνση της παρουσίασης απλοποιούμε λίγο το πρόβλημα και έστω ότι απαιτούμε μόνο την θερμοκρασία για ένα σημείο και για μια δεδομένη χρονική στιγμή. Επομένως, θα εκτελέσουμε μόνο ένα ερώτημα στον πίνακα Θερμοκρασία. επίσης, έστω ότι αναζητούμε την χρονική στιγμή 15 Ιουνίου του 2021 και ώρα 12 το μεσημέρι. Μόλις κάνουμε το ερώτημα με βάση τον μήνα, το έτος και το σημείο τότε θα μας επιστραφεί άμεση η εγγραφή στην βάση που περιλαμβάνει όλη αυτή την πληροφορία. Μέσα στα πεδία που επιστρέφονται είναι και το πεδίο με τις τιμές για το συγκεκριμένο μήνα και έτος.

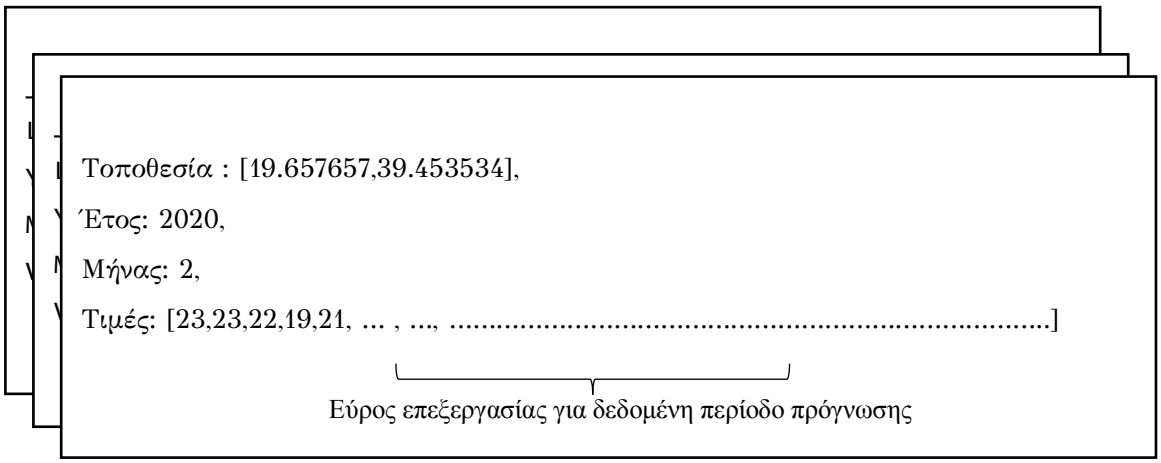
Το πεδίο με τις τιμές είναι ένας πίνακας. Αφού στην αρχή της υποενότητας αναφέραμε ότι σε κάθε εισαγωγή του μετεωρολογικού μοντέλου τοποθετούμε στο τέλος κάθε πρόγνωση αυτό σημαίνει ότι τοποθετούμε 36 τιμές κάθε φορά για κάθε μέρα που εκτελείται το μετεωρολογικό μοντέλο. Επομένως το μέγεθος του πίνακα είναι όσο οι μέρες του μήνα επί 36 τιμές. Για την συνεχή αποθήκευση των δεδομένων στην αρχή οι πίνακες γεμίζουν με τιμές πολύ διαφορετικές από τις φυσιολογικές και μετά γίνονται τροποποίηση από τα πραγματικά δεδομένα. Για τον λόγο αυτό κάθε πίνακας γεμίζει αρχικά με την παραδοχή ότι ο μήνας έχει 31 ημέρες. Οπότε, το πλήθος των τιμών στον πίνακα αυτόν είναι: 31X36(1116).

Πίσω στο παράδειγμα μας τώρα και με βάση αυτά που αναφέραμε, για να βρούμε την τιμή που αντιστοιχεί στην 15η μέρα του μήνα και στις 12 το μεσημέρι πρέπει να κάνουμε κάποιον αριθμητικό υπολογισμό. Αρχικά, θα πρέπει να βρούμε την θέση της τιμής αυτής στον πίνακα. Κάθε 36 τιμές έχουμε μία μέρα οπότε η πρόγνωση για την 15η ημέρα ξεκινάει από το στοιχείο 14*36 + 1 του πίνακα. Οι πρώτες 504 τιμές αντιστοιχούν στις πρώτες 14 ημέρες. Οπότε, στο συγκεκριμένο

παράδειγμα η 505 τιμή είναι η πρώτη τιμή της 15ης ημέρας. Η τιμή που αντιστοιχεί στις 12 το μεσημέρι είναι η 6η τιμή της πρόγνωσης από τις συνολικά 36 που υπάρχουν για όλο το τριήμερο της πρόγνωσης. Τελικά, η τιμή που ψάχνουμε βρίσκεται στην 510 θέση του πίνακα.

Στην παραπάνω παράγραφο είδαμε ένα πολύ απλό παράδειγμα για το πώς θα δούλευε η ανάκτηση της θερμοκρασίας από μία τέτοια σχεδίαση του αποθηκευτικού χώρου. Προκύπτει ότι εκτός από την καθυστέρηση που έχει η εκτέλεση του ερωτήματος υπάρχει και μια προγραμματιστική επιβάρυνση η οποία θα μπορούσε να μην υφίσταται καν. Η επιπλέον επεξεργασία φαίνεται στο σχήμα 3.6.

Συνοψίζοντας, θα λέγαμε ότι η προσέγγιση αυτή αποτελεί μια καλή σχεδίαση για το ιστορικό πάνω σε δεδομένα από πρόγνωση καιρού μιας και έχει μικρή επιβάρυνση για την εισαγωγή των δεδομένων ωστόσο από την άλλη για λειτουργίες άμεσης απόκρισης δεν αποτελεί μια προτεινόμενη σχεδιαστική λύση.



Σχήμα 3.6: Διαδικασία επιπλέον επεξεργασίας στην συλλογή

3.2.2 Προηγμένη αποθήκευση μετεωρολογικών δεδομένων

Σε αυτή την υποενότητα θα αναλύσουμε την προηγμένη αποθήκευση μετεωρολογικών δεδομένων για την αποθήκευση των δεδομένων που προκύπτουν από το μετεωρολογικό μοντέλο.

Κατά την πρώτη εκδοχή όπως θα δούμε και στα αποτελέσματα η καθυστέρηση είναι σημαντική και η χρήση της εφαρμογής γίνεται δύσκολη. Οπότε με απλά λόγια η προσέγγιση αυτή αποσύρεται μιας και δεν θα χρησιμοποιηθεί καθόλου για τις λειτουργίες της άμεσης απόκρισης. Η προσέγγιση αυτή θα χρησιμοποιηθεί μόνο για την εξαγωγή ιστορικών στοιχείων πάνω στις καιρικές συνθήκες.

Η δεύτερη προσέγγιση θα την χαρακτηρίζαμε ως λειτουργιο-κεντρική. Ο λόγος που την χαρακτηρίζουμε έτσι είναι διότι εξαρτάται κατά πολύ από το πώς θέλουμε να παρουσιάζονται οι λειτουργίες. Εκτός από λειτουργιο-κεντρική θα μπορούσε να χαρακτηριστεί ως σημείο-κεντρική. Ο λόγος που την χαρακτηρίζουμε έτσι είναι διότι επιλέγουμε να έχουμε όλη την πληροφορία για κάθε σημείο ομαδοποιημένη σε μία εγγραφή.

Στην πρώτη προσέγγιση διακρίνουμε πολλές συλλογές δεδομένων για όλες τις μετρικές που έχουμε και μέσα σε αυτές τοποθετούμε τα νέα δεδομένα. Αντίθετα, στην δεύτερη προσέγγιση αποσυνδέουμε τα πρόσφατα δεδομένα από όλα τα δεδομένα που υπάρχουν. Δηλαδή κάθε φορά που εκτελείται το μοντέλο, αντί να τοποθετούμε τις τιμές σε κάθε συλλογή, επινοούμε μία νέα συλλογή στην οποία αποθηκεύονται μόνο τα δεδομένα της πρόσφατης πρόγνωσης. Αναφέραμε ότι η εισαγωγή γίνεται πρώτα στις συλλογές του ιστορικού. Προκύπτει λοιπόν μια επιπλέον επιβάρυνση όσον αφορά την δημιουργία της νέας συλλογής από το μετεωρολογικό μοντέλο. Αυτό σημαίνει ότι κατά την εισαγωγή θα πρέπει να προσπελάσουμε για κάθε σημείο κάθε συλλογή και να τα ομαδοποιήσουμε. Η προσέγγιση αυτή απαιτεί μόνο αυτή την επιβάρυνση την οποία ο χρήστης δεν θα την καταλάβει ποτέ.

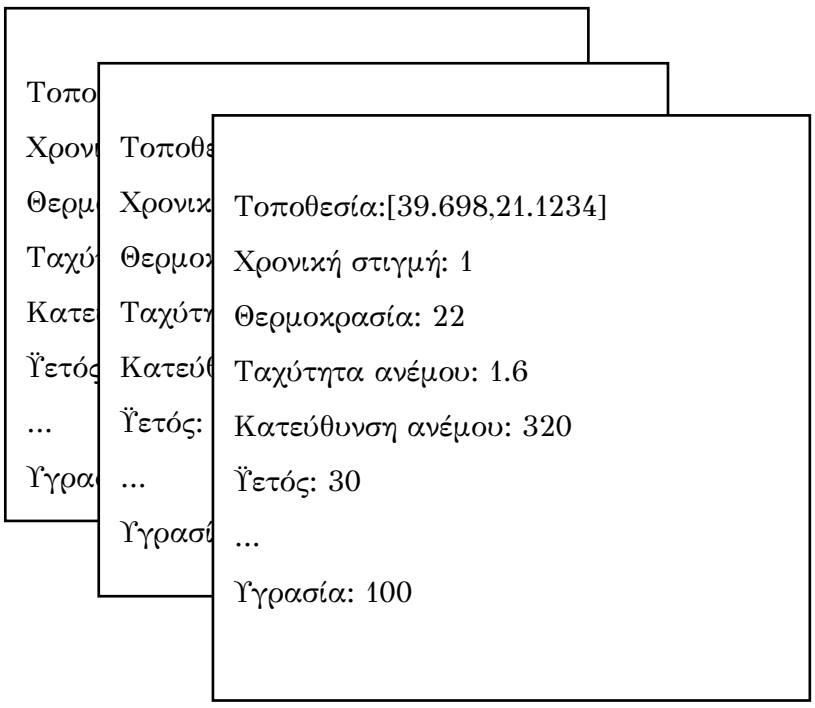
Πιο συγκεκριμένα, κάθε φορά που εκτελείται το μετεωρολογικό μοντέλο τα δεδομένα αποθηκεύονται αρχικά σε κάθε συλλογή σχετική με την μετεωρολογική παράμετρο που μας ενδιαφέρει. Για παράδειγμα, εισάγουμε τα δεδομένα για την θερμοκρασία στην συλλογή για την θερμοκρασία, τα αντίστοιχα δεδομένα για τον άνεμο στην αντίστοιχη συλλογή κ.ο.κ. Στην συνέχεια, μία άλλη διεργασία αναλαμβάνει να δημιουργήσει την συλλογή με τις πρόσφατες συνθήκες από το μετεωρολογικό μοντέλο.

Η διαδικασία της δημιουργίας και ολοκλήρωσης της δεύτερης συλλογής την οποία θα χρησιμοποιούμε πιο συχνά όπως είπαμε και πριν είναι χρονοβόρα και ο λόγος είναι ο εξής: εφόσον τα δεδομένα συνολικά βρίσκονται σε συλλογές για κάθε μετεωρολογική παράμετρο ανά σημείο, για να δημιουργήσουμε μία πιο δομημένη συλλογή πρέπει κάθε φορά να διατρέχουμε μία-μία τις συλλογές και να εξάγουμε την πληροφορία που μας ενδιαφέρει. Και ο αλγόριθμος είναι ο παρακάτω:

- Για κάθε σημείο και χρονική στιγμή της πρόγνωσης
- Διατρέχουμε κάθε συλλογή από τις μετρικές
- Προσθέτουμε την εγγραφή για το σημείο και την χρονική στιγμή στην νέα συλλογή

Για να καταλάβουμε καλύτερα την εισαγωγή των δεδομένων θα πρέπει να δούμε την δομή των εγγραφών της νέας συλλογής. Η συλλογή μας αποτελείται από πολλές εγγραφές. Κάθε εγγραφή αντιστοιχεί σε κάθε σημείο μια δεδομένη χρονική στιγμή. Επομένως, το πλήθος των εγγραφών είναι το γινόμενο του πλήθους των σημείων επί το πλήθος των χρονικών στιγμών στο διάστημα της μετεωρολογικής πρόγνωσης. Δηλαδή αν έχουμε περίπου 21000 σημεία για το πεδίο της Ηπείρου (32000 ολόκληρη Ελλάδα) και 36 χρονικές στιγμές (2ωρα για τις επόμενες 3 ημέρες) τότε μας προκύπτει ένα αρκετά μεγάλο πλήθος εγγραφών περίπου 700.000 (1.000.000) εγγραφές για ολόκληρη την Ελλάδα).

Πιο συγκεκριμένα κάθε εγγραφή αποτελείται από τα παρακάτω:



Σχήμα 3.7: Δομή προηγμένης αποθήκευσης μετεωρολογικών δεδομένων

Για παράδειγμα στην βάση μας υπάρχει η ακόλουθη εγγραφή:

- 38.56, 21.89 (γεωγραφικό πλάτος και μήκος)
- 2 (χρονική στιγμή που αντιστοιχεί στο 2ο δίωρο της πρώτης ημέρας , δηλαδή στις 6 το πρωί)
- 17 (θερμοκρασία σε βαθμούς Κελσίου)
- 4 (ταχύτητα ανέμου σε μέτρα ανά δευτερόλεπτο)
- 230 (διεύθυνση ανέμου σε μοίρες)
- 1100 (πίεση σε hPa)
- 4 (υετός σε mm ανά δίωρο)
- 80 (υγρασία σε %)
- κ.ο.κ.

Τα δύο πιο κρίσιμα πεδία στις παραπάνω δομές αποτελούν η τοποθεσία και η χρονική στιγμή. Και αυτό διότι υπάρχουν μια σειρά από ενδιαφέροντα ερωτήματα και αιτήσεις οι οποίες βασίζονται σε τοπικά χαρακτηριστικά και χρονολογικά χαρακτηριστικά είτε και συνδυασμός και των δύο.

Για παράδειγμα, αναζητούμε τα σημεία εκείνα που μια δεδομένη χρονική στιγμή έχουν θερμοκρασία μεγαλύτερη από μία τιμή που επιλέγει ο χρήστης, τις χρονικές στιγμές για τις οποίες ένα σημείο έχει μεγαλύτερη θερμοκρασία από μία τιμή και ταχύτητα ανέμου μεγαλύτερη από μία τιμή και πολλά άλλα ερωτήματα που έχουν σαν κύρια κριτήρια αναζήτησης την θέση και την χρονική στιγμή. Γι' αυτό και τα συστήματα αυτά ονομάζονται χωροχρονικά, με την έννοια ότι ασχολούνται με θέση και χρόνο.

3.2.3 Σχεδίαση ευρετηρίων για την προηγμένη αποθήκευση μετεωρολογικών δεδομένων

Στο σημείο αυτό, θα πρέπει να αναλύσουμε σε βάθος τον κύριο μηχανισμό με τον οποίο η σχεδίαση και η υλοποίηση τέτοιων ερωτημάτων γίνεται εφικτή. Αυτόν τον μηχανισμό μας τον παρέχουν παραδοσιακά όλα τα συστήματα βάσεων δεδομένων είτε είναι σχεσιακά είτε όχι. Αυτός ο μηχανισμός δεν είναι άλλος από την ευρετηριοποίηση. Στην συγκεκριμένη σχεδίαση έχουμε επιλέξει το σύστημα βάσης δεδομένων της MongoDB. Ένας από τους πολλούς λόγους που χρησιμοποιήθηκε

αυτό το σύστημα είναι η διαχείριση που παρέχει για ευρετήρια γεωγραφικών σημείων και χρονοσειρών.

Στην MongoDB χρησιμοποιείται το ευρετήριο 2dsphere, αρκεί να έχουμε ορίσει το σημείο ως point. Ένα ευρετήριο 2dsphere υποστηρίζει ερωτήματα που υπολογίζουν γεωμετρίες σε μια σφαίρα σαν τη γη. Το ευρετήριο 2dsphere υποστηρίζει όλα τα γεωχωρικά ερωτήματα MongoDB: ερωτήματα για συμπερίληψη, διασταύρωση και εγγύτητα. Το ευρετήριο 2dsphere υποστηρίζει δεδομένα που αποθηκεύονται ως αντικείμενα GeoJSON και ζεύγη συντεταγμένων. Με ένα τέτοιο εργαλείο μπορούμε εύκολα να ανακτούμε πληροφορία για κοντινότερα σημείο ως προς αυτό που έχουμε ως παράμετρο, κοντινά με βάση κάποια ακτίνα, αποστάσεις μεταξύ σημείων και ούτω καθεξής. Αυτές οι λειτουργίες χωρίς ευρετήριο και χωρίς συγκεκριμένο 2dsphere ευρετήριο θα απαιτούσαν να διατρέχουν όλα τα σημεία και να κάνουμε πράξεις αφού τα διατρέξουμε το οποίο προφανώς και είναι ασύμφορο και μη αποδεκτό για μια τέτοια εφαρμογή. Επομένως, έχοντας το ευρετήριο για τα σημεία καταφέρνουμε να αποθηκεύουμε με βάση την τοπικότητα των σημείων. Κοντινά σημεία στον χάρτη είναι και κοντινά σημεία στο ευρετήριο αυτό.

Παράλληλα με την δημιουργία του ευρετηρίου για την γεωγραφική θέση, δημιουργούμε και ένα ευρετήριο για την χρονική στιγμή. Οπότε εξασφαλίζουμε και την τοπικότητα μεταξύ των χρονικών στιγμών. Κοντινές χρονικές στιγμές αποθηκεύονται κοντά στο ευρετήριο. Το ευρετήριο αυτό μας παρέχει μία δυνατότητα την οποία χρησιμοποιούμε πάρα πολύ στην τελική εφαρμογή. Με ένα κλικ ανακτούμε την πληροφορία ενός σημείου για όλη την πρόγνωση. Για να γίνει αυτό εφικτό πρέπει για το σημείο που έγινε κλικ άμεσα να πάρουμε όλες τις εγγραφές για το σημείο αυτό για όλες τις χρονικές στιγμές. Με την χρήση ενός τέτοιου ευρετηρίου καθίσταται εφικτή μία τέτοια λειτουργία.

Συνοψίζοντας σχετικά με τα ευρετήρια τονίζουμε ότι χωρίς την χρήση αυτών των μηχανισμών δεν θα μπορούσε να ήταν εφικτές πολλές λειτουργίες που έχουμε αναπτύξει στην τελική εφαρμογή.

3.2.4 Αποθήκευση δεδομένων με βάση το επίπεδο ανάλυσης

Γενικά, στις εφαρμογές διαχείρισης και οπτικοποίησης δεδομένων πάνω σε χάρτη είναι εμφανές ότι τα διάφορα στοιχεία που εμφανίζονται εξαρτώνται από

το επίπεδο ανάλυσης. Δηλαδή, ανάλογα με το επίπεδο ζουμ που έχει επιλέξει ο χρήστης να εμφανίζονται τα διάφορα δεδομένα πάνω στον χάρτη. Αυτό γίνεται φανερό και εύκολα αντιληπτό όταν χρησιμοποιούμε τους ίδιους τους χάρτες. Όταν το επίπεδο ζουμ είναι αρκετά μικρό τότε βλέπουμε σχεδόν ηπείρους και ωκεανούς οπότε η πληροφορία δεν χρειάζεται να απεικονίζεται ολόκληρη παρά μόνο μία υποομάδα αυτών των δεδομένων χωρίς να χάνεται η ουσία. Οπότε απεικονίζεται κάθε φορά ένας αριθμός από εικόνες που βασίζεται στο επίπεδο ανάλυσης. Στην περίπτωση μας αυτό μπορεί να γίνει κατά την οπτικοποίηση των δεδομένων είτε κατά την περίπτωση που θέλουμε να απεικονίσουμε το επίπεδο του ανέμου με διάφορα βελάκια κατεύθυνσης. Εφαρμόζουμε λοιπόν την ίδια τεχνική που εφαρμόζουν γενικά οι χάρτες στο επίπεδο του ανέμου. Για να απεικονίσει κάποιος πάνω σε ένα χάρτη διαδραστικά όλη την πληροφορία της μετεωρολογική παράμετρος του ανέμου θα οδηγήσει σε καθυστέρηση της εφαρμογής που προσπαθεί να απεικονίσει την κατεύθυνση αυτή. Οπότε, για αυτό τον λόγο υιοθετούμε την τεχνική που προείπαμε.

Για να γίνει αυτό εφικτό θα πρέπει να ομαδοποιηθούν τα σημεία σε κατηγορίες ανάλογα με το ζουμ επιλογής. Οπότε φτιάχνουμε μία συλλογή για κάθε επίπεδο ανάλυσης για τον άνεμο και κάθε συλλογή περιέχει μία υποομάδα από τα δεδομένα μας. Συνεπώς αντί να έχουμε όλα τα δεδομένα σε κάθε συλλογή αποφασίζουμε με βάση τις μετρήσεις να έχουμε γύρω μέγιστο αριθμό 3000 σημεία σε κάθε συλλογή.

3.2.5 Σύνοψη για την αποθήκευση των δεδομένων με τις δύο εκδοχές

Στο σημείο αυτό συνοψίζουμε την ενότητα για την αποθήκευση των δεδομένων τονίζοντας για μία τελευταία φορά τα υπέρ και τα κατά των δύο εκδοχών που αναλύσαμε στο παραπάνω κεφάλαιο.

Στην πρώτη εκδοχή αναλύσαμε μία προσέγγιση αποθήκευσης των δεδομένων σε συλλογές που περιείχαν συναφή δεδομένα ως προς κάθε μετεωρολογική παράμετρο που θέλουμε να αποθηκεύσουμε. Τονίσαμε ότι η προσέγγιση αυτή έχει το πλεονέκτημα ότι η αποθήκευση των νέων δεδομένων γίνεται γρήγορα με αποτέλεσμα τα νέα δεδομένα από την στιγμή που θα εξαχθούν από το μοντέλο θα

γίνουν άμεσα διαθέσιμα. Επίσης, τονίσαμε ότι για ερωτήματα που έχουν να κάνουν με απεικόνιση των δεδομένων αυτών πάνω σε χάρτη χωρίς ορατή καθυστέρηση η χρήση αυτής της σχεδίασης γίνεται ανέφικτη. Αυτή η προσέγγιση θα υιοθετηθεί μόνο για αποθήκευση των δεδομένων για εργασίες που δεν μας απασχολεί πόσο χρόνο θα πάρουν για να εκτελεστούν.

Στην δεύτερη εκδοχή προτείναμε έναν τρόπο αποθήκευσης των δεδομένων με γνώμονα τις λειτουργίες που θέλουμε να πετύχουμε στην εφαρμογή διεπαφής χρήστη. Με βάση λοιπόν την αρχή αυτή προτείνουμε να αποσυνδέσουμε τα δεδομένα της τελευταίας πρόγνωσης από τα υπόλοιπα δεδομένα με σκοπό να επιτύχουμε αμεσότερη προσπέλαση των δεδομένων αυτών και εν τέλει μια πολύ καλή εμπειρία χρήστη. Έτσι, σχεδιάσαμε μία δομή, όπως την χαρακτηρίσαμε σημειοκεντρική, η οποία με κέντρο κάθε σημείο και κάθε χρονική στιγμή στο εύρος της μετεωρολογικής πρόγνωσης αποθηκεύει όλες τις μετρικές σαν χαρακτηριστικά των 2 κύριων μεταβλητών, της θέσης και της χρονικής στιγμής.

Επομένως, οι δύο εκδοχές θα χρησιμοποιηθούν και εφόσον έχουμε κάνει την παραδοχή ότι κύριο μέλημα μας είναι η εμπειρία χρήστη αρχικά δεν μας απασχολεί να έχουμε κάποια πληροφορία ως διπλότυπη.

3.3 Σχεδίαση του αποθηκευτικού χώρου για τα δεδομένα από αισθητήρες καιρικών συνθηκών

Σε αυτή την ενότητα θα δώσουμε περισσότερες λεπτομέρειες σχετικά με τα δεδομένα που προκύπτουν από τους αισθητήρες μετεωρολογικών δεδομένων. Όπως προείπαμε τα δεδομένα αυτά έχουν μία ιδιότητα που τα κάνει δύσκολα στην διαχείριση και αυτή είναι η διαφορετικότητα των πηγών από όπου αντλούνται τα δεδομένα αυτά. Υπάρχουν δεδομένα που αντλούνται από αισθητήρες που είναι ήδη εγκατεστημένοι και ανήκουν στο τμήμα Φυσικών του Πανεπιστημίου Ιωαννίνων, αισθητήρες που ανήκουν στην Αποκεντρωμένη Διοίκηση Ηπείρου και Δυτικής Μακεδονίας όπως και επίσης αισθητήρες που βρίσκονται ελεύθεροι στο διαδίκτυο. Η δυσκολία που επιφέρει αυτή η διαφορετικότητα αντισταθμίζεται με την πολυμορφία και την ποικιλία των δεδομένων. Δηλαδή με λίγα λόγια ναι μεν είναι δύσκολο να γίνει η συλλογή από πολλαπλές πηγές αλλά είναι αρκετά ενδι-

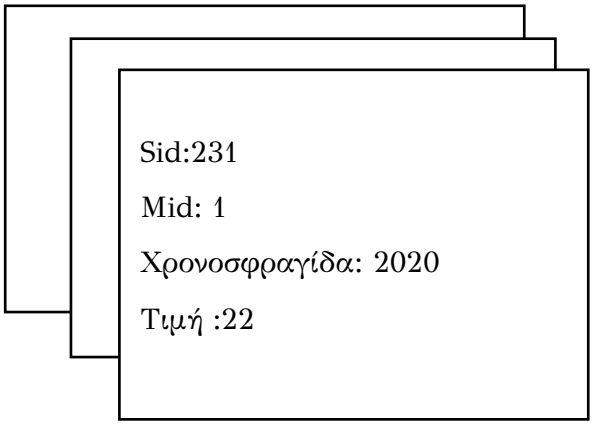
αφέρον ότι για την περιοχή που μας απασχόλησε κατά την διάρκεια της εργασίας η κάλυψη φτάνει σε ένα αρκετά ικανοποιητικό βαθμό. Η πυκνότητα των σταθμών αυτών μας επιτρέπει καλύτερη ανάλυση πάνω στην περιοχή που μας ενδιαφέρει για οτιδήποτε μελετάμε από μετεωρολογικής φύσεως όπως: επικινδυνότητα πυρκαγιάς, μελέτες σχετικά με αγροκαλλιέργειες, πάνω στην κτηνοτροφία κ.α. Όπως είπαμε και πριν τα δεδομένα προέρχονται από τρεις διαφορετικές πηγές. Παρόλα αυτά θα πρέπει να αποθηκευτούν με ένα ομοιογενές σχήμα στις συλλογές της βάσης δεδομένων.

Πριν εξηγήσουμε τις αποφάσεις που πήραμε για την σχεδίαση της βάσης δεδομένων για τα δεδομένα αυτά θα πρέπει να απαντήσουμε στις εξής ερωτήσεις: τι θέλουμε τα δεδομένα αυτά; για ιστορικό έλεγχο μιας περιοχής, για ανάλυση των συνθηκών που επικρατούν κατά την δεδομένη χρονική στιγμή ή και τα δύο;

Η δυνατότητα να αποθηκεύει κάποιος τέτοιου είδους δεδομένα για μία περιοχή αποτελεί σημαντικό εργαλείο οποιουδήποτε θέλει να χρησιμοποιήσει τέτοια δεδομένα. Ένας μετεωρολόγος μπορεί να τα χρησιμοποιήσει σαν ανατροφοδότηση σε κάποιο μοντέλο πρόγνωσης καιρού, δηλαδή να βλέπει τις τιμές που εξάγει το μοντέλο να τις συγκρίνει με αυτές που επικρατούν αυτή τη στιγμή μέσω των σταθμών και να κάνει τυχόν διορθώσεις στο μοντέλο. Ένας πυροσβέστης έχει την δυνατότητα να ανατρέξει σε προηγούμενα έτη και να αντλήσει χρήσιμη πληροφορία η οποία θα τον βοηθήσει να λάβει τις καλύτερες αποφάσεις τόσο κατά την διάρκεια όσο και κατά την πρόληψη μίας πυρκαγιάς. Ένας κτηνοτρόφος μπορεί να αντλήσει πληροφορία σχετικά με τις τρέχουσες συνθήκες σε μία περιοχή για να λάβει απόφαση αν θα πρέπει να μετοικήσει ή όχι. Άρα παρατηρούμε ότι τα δεδομένα αυτά αποτελούν μια χρήσιμη πηγή πληροφορίας τόσο ως πρόσφατη πληροφορία όσο και ως ιστορική πληροφορία.

3.3.1 Αποθήκευση δεδομένων της πρόσφατης πληροφορίας

Όπως και στην αποθήκευση της μετεωρολογικής πρόγνωσης υπάρχει και εδώ το δίλημμα να χρησιμοποιηθεί μία διαφορετική συλλογή για κάθε μετεωρολογική παράμετρο που υπάρχει στα δεδομένα. Έτσι θα είχαμε μία συλλογή για την θερ-



Σχήμα 3.8: Δομή αποθήκευσης δεδομένων αισθητήρων

μοκρασία, μία για τον άνεμο, μία για την υγρασία μία για την κάθε άλλη μετεωρολογική παράμετρο που εμφανίζεται στα δεδομένα των αισθητήρων. Σε περίπτωση που θέλουμε να αντλήσουμε πληροφορία για έναν σταθμό πρέπει να εξάγουμε δεδομένα από κάθε συλλογή δεδομένων. Θα πρέπει να κάνουμε ένα ερώτημα σε κάθε συλλογή που έχουμε τα δεδομένα. Για να αποφευχθεί αυτό έχουμε δημιουργήσει μία συλλογή στην οποία αποθηκεύουμε τα πρόσφατα δεδομένα για κάθε μέρα. Έτσι αυτή η συλλογή έχει την δομή του σχήματος 3.8.

- Sid : το αναγνωριστικό του σταθμού
- Mid : το αναγνωριστικό της κάθε μετεωρολογική παράμετρος

Η δομή των δεδομένων σε αυτή την συλλογή διαφέρει ως προς την μετεωρολογική πρόγνωση στην οποία είχαμε για κάθε σημείο όλη την πληροφορία. Ο λόγος που δεν υιοθετούμε το ίδιο μοντέλο και σε αυτή την συλλογή είναι διότι δεν έχουμε τόσα πολλά διαφορετικά σημεία για αποθήκευση οπότε απλοποιούμε την δομή μας. Οπότε αν γίνει κάποιο ερώτημα σχετικά με τις πρόσφατες τιμές ενός σταθμού αρκεί να ψάξουμε με βάση το sid σε αυτή την βάση και στην συνέχεια να κάνουμε ταξινόμηση ως προς την χρονική στιγμή για να πάρουμε τα δεδομένα με σωστή χρονολογική σειρά.

3.3.2 Αποθήκευση δεδομένων ιστορικού χαρακτήρα

Όσον αφορά το ιστορικό των δεδομένων από τους αισθητήρες των σταθμών αυτό που υιοθετούμε είναι το ίδιο με την μετεωρολογική πρόγνωση μόνο που σε αυτή την συλλογή η αποθήκευση γίνεται σε επίπεδο ημέρας και όχι μήνα. Και αυτό διότι εφόσον αποθηκεύουμε την πρόσφατη πληροφορία σε επίπεδο ημέρας όταν θα χρειαστεί να την αποθηκεύσουμε σε συλλογή του ιστορικού μας εξυπηρετεί να αποθηκεύσουμε τα δεδομένα πάλι σε επίπεδο ημέρας. Μας επιτρέπεται να γίνει αυτό και για τον λόγο ότι όπως και πριν τα διαφορετικά σημεία των αισθητήρων είναι πολύ λιγότερα από της πρόγνωσης. Επομένως για το ιστορικό έχουμε την εξής δομή:

Για κάθε μετεωρολογική παράμετρο που εμφανίζεται στους αισθητήρες των σταθμών έχουμε μία συλλογή με την εξής δομή:

- Sid : αναγνωριστικό του σταθμού προέλευσης των δεδομένων
- Ημέρα των μετρήσεων
- Δεδομένα σε μορφή πίνακα ταξινομημένα με χρονολογική σειρά

Η παραπάνω δομή είναι απλή στη σχεδίαση παρόλα αυτά είναι μία αρκετά ισχυρή δομή ως προς την εξαγωγή δεδομένων από την δομή αυτή. Συνήθως σε ιστορικά δεδομένα κάνουμε ερωτήματα πάνω σε έναν σταθμό και μία μετεωρολογική παράμετρο. Οπότε, αυτό που απαιτείται είναι να ψάξουμε στην εκάστοτε συλλογή-μετεωρολογική παράμετρο και να αναζητήσουμε με βάση των σταθμό της επιλογής μας. Η πληροφορία θα πρέπει να είναι ταξινομημένη με βάση την χρονολογική σειρά κάθε ημέρας.

3.3.3 Ευρετηριοποίηση των συλλογών αποθήκευσης δεδομένων από αισθητήρες

Όπως και στην μετεωρολογική πρόγνωση έτσι και στις συλλογές όπου αποθηκεύουμε δεδομένα από τους αισθητήρες απαιτείται να χρησιμοποιηθούν ευρετήρια τα οποία θα βελτιστοποιήσουν την ταχύτητα εξαγωγής των δεδομένων. Έτσι λοιπόν στην συλλογή για τα πρόσφατα δεδομένα έχουμε δημιουργήσει ένα ευρετήριο πάνω στην χρονολογική σειρά οπότε οποιαδήποτε ταξινόμηση γίνεται πάνω στην ημερομηνία, επιστρέφει τα δεδομένα με βάση το ευρετήριο αυτό. Επίσης,

δημιουργούμε ένα ευρετήριο βάσει του μετεωρολογικού σταθμού οπότε τα δεδομένα ενός σταθμού βρίσκονται κοντά μέσα στο ευρετήριο αυτό.

Όσον αφορά το ιστορικό μιας και έχουμε απλοποιημένη την δομή με βάση κάθε μετεωρολογική παράμετρο δημιουργούμε ένα ευρετήριο για το αναγνωριστικό του σταθμού και ένα ευρετήριο σχετικό με την ημερομηνία κάθε ημέρας που αποθηκεύουμε. Με αυτόν τον τρόπο έχουμε βέλτιστη απόδοση πάνω στην ταχύτητα εξαγωγής των δεδομένων και από την συλλογή του ιστορικού.

3.3.4 Αντίγραφα ασφαλείας των δεδομένων από τους αισθητήρες των μετεωρολογικών σταθμών

Ένα κρίσιμο ερώτημα που προκύπτει είναι το εξής: σε περίπτωση που αστοχήσει οποιαδήποτε λειτουργία σχετικά με την αποθήκευση των ιστορικών δεδομένων από την πρόσφατη πληροφορία τότε θα δημιουργηθεί ένα κενό στα δεδομένα του ιστορικού. Για να αποφευχθεί αυτό το κακό σενάριο έχουμε σκεφτεί τα δεδομένα να αποθηκεύονται και σε μία επιπλέον αποθήκη δεδομένων ως απλά δεδομένα αποθηκευμένα με την εξής δομή:

- Sid: σταθμός
- Mid: μετεωρολογική παράμετρο
- Χρονική στιγμή
- Τιμή μετεωρολογική παράμετρος

Κάθε φορά λοιπόν που τρέχει κάποια από τις λειτουργίες εισαγωγής δεδομένων από τις πηγές κάνουμε και μία επιπλέον εισαγωγή σε αυτή την συλλογή σε περίπτωση που κάτι πάει στραβά σε κάποιο επόμενο βήμα τακτοποίησης των δεδομένων. Επομένως, με αυτό τον τρόπο όπως θα δούμε και αργότερα επιτυγχάνουμε μέγιστη ασφάλεια στην ολοκλήρωση και την ακεραιότητα των δεδομένων μας.

ΚΕΦΑΛΑΙΟ 4

Υλοποιηση

4.1 Υλοποίηση αποθήκευσης δεδομένων για την πρόγνωση

4.2 Υλοποίηση τελικής εφαρμογής για την πρόγνωση

4.3 Υλοποίηση αποθήκευσης δεδομένων μετεωρολογικών σταθμών

4.4 Υλοποίηση τελικής εφαρμογής για τους μετεωρολογικούς σταθμούς

Σε αυτό το κεφάλαιο θα παρουσιάσουμε την υλοποίηση όλων των δομικών και σχεδιαστικών τεχνικών που αναφέρθηκαν παραπάνω καθώς και την συσχέτιση τους με τη τελική εφαρμογή χρήστη. Καθοριστικό σημείο για την υλοποίηση αποτελεί η κατανόηση ότι τα αποθηκευμένα δεδομένα έχουν άμεση σχέση με την τελική εφαρμογή. Επομένως, το συγκεκριμένο κεφάλαιο το αποσυνδέουμε σε τέσσερις υποενότητες: την υλοποίηση της αποθηκευτικής δυνατότητας για την πρόγνωση, την υλοποίηση της εφαρμογής για την πρόγνωση, την υλοποίηση της αποθηκευτικής δυνατότητας για τους σταθμούς και την υλοποίηση της τελικής εφαρμογής για τους μετεωρολογικούς σταθμούς. Κάθε μία από τις υλοποιήσεις αυτές έρχεται σε παραλληλισμό με τις σχεδιαστικές αρχές που δόθηκαν στην προηγούμενη ενότητα.

4.1 Υλοποίηση αποθήκευσης δεδομένων για την πρόγνωση

Αρχικά θα πρέπει να υπενθυμίσουμε ότι σημαντική αρχή για την υλοποίηση κάθε διαδικασίας είναι η ταχύτητα απόκρισης και η ακεραιότητα των δεδομένων.

Για την αποθήκευση των δεδομένων χρησιμοποιούμε την μηχανή βάσεων δεδομένων MongoDB όπως περιγράψαμε με λεπτομέρεια στην ενότητα 2. Η μηχανή αυτή μας επιτρέπει να διαχειριζόμαστε επαρκώς χρονοχωρικά δεδομένα.

Πιο συγκεκριμένα, για την αποθήκευση των δεδομένων της πρόγνωσης δημιουργήσαμε δύο βάσεις δεδομένων για κάθε πεδίο της πρόγνωσης μία για την Ήπειρο και μία για την Ελλάδα.

Μέσα σε καθεμιά από τις βάσεις δεδομένων αποθηκεύουμε όπως σχεδιάστηκε στην προηγούμενη ενότητα τους πίνακες.

```
> show dbs
Coordinates            0.002GB
ForecastEpirus         0.230GB
ForecastGreece         0.159GB
WeatherForecast        87.443GB
```

Σχήμα 4.1: Κύρια διάκριση βάσεων δεδομένων

Στο σχήμα 4.1 δείχνουμε την βασική διάκριση που κάνουμε στις βάσεις δεδομένων για τα δύο πεδία των περιοχών ενδιαφέροντος. Μία βάση δεδομένων ανήκει στο πεδίο της Ηπείρου, μία στην περιοχή ολόκληρου του Ελλαδικού χώρου για τις μετρήσεις της κάθε εκτέλεσης του μοντέλου και μία για την αποθήκευση

```
> show tables
Forecast.LastDate
Forecast_coordinates
Forecast_cprecip
Forecast_filters
Forecast_hcloud
…..
```

Σχήμα 4.2: Εσωτερική σχεδίαση της βάσης δεδομένων για την περιοχή της Ηπείρου

των ιστορικών δεδομένων της περιοχής της Ηπείρου η οποία μας ενδιαφέρει επιχειρησιακά πιο πολύ. Επίσης, στην δεξιά πλευρά διακρίνουμε την πληροφορία σχετικά με τον χώρο.

Στο Σχήμα 4.2 βλέπουμε εσωτερικά την υλοποίηση της δομής της βάσης δεδομένων για την περιοχή της Ηπείρου. Η εντολή show tables μας εμφανίζει τους πίνακες-συλλογές μέσα στη βάση δεδομένων. Παρατηρούμε ότι έχουμε μία συλλογή για κάθε μετεωρολογική παράμετρο και επιπλέον μία συλλογή στην οποία αποθηκεύουμε την χρονοσφραγίδα της τελευταίας ανανέωσης των δεδομένων (Forecast.LastDate), μία συλλογή για τις συντεταγμένες που ανήκουν στο πεδίο της Ηπείρου (Forecast_coordinates) και μία συλλογή στην οποία αποθηκεύουμε την δομή για τις πληροφορίες κάθε σημείου ξεχωριστά για κάθε χρονική στιγμή (Forecast_filters).

```
> db.Forecast_temp2.findOne()
{
        "_id" : ObjectId("611abfbe5abee74fef16ee6c"),
        "date" : ISODate("2021-08-16T00:00:00Z"),
        "loc" : {
                "type" : "Point",
                "coordinates" : [
                        18.9460754,
                        38.2129364
                ]
        },
        "daily_values" : [….]
}
```

Σχήμα 4.3: Εσωτερική σχεδίαση της συλλογής θερμοκρασίας για την περιοχή της Ηπείρου

Στο Σχήμα 4.3 παρουσιάζουμε την εσωτερική δομή για κάθε πίνακα στην βάση δεδομένων για την περιοχή της Ηπείρου. Αποθηκεύουμε την ημερομηνία, την τοποθεσία και τις τιμές για τις οποίες αναφέρεται η πρόγνωση. Για παράδειγμα, για τις 16/08/2020, για το σημείο [18.946074, 38.2129364] έχουμε στον πίνακα daily_values 36 τιμές μία για κάθε ώρα τις πρόγνωσης. Όπως είπαμε και στην

ενότητα της σχεδίασης η έννοια των χρονοσειρών επειδή είναι στατική δεν απαιτείται να βάζουμε σε κάθε τιμή και την χρονοσφραγίδα της. Γνωρίζουμε ότι η

```
> db.Forecast_filters.findOne()
{
        "_id" : ObjectId("611abffa0d7057ac1b8e7382"),
        "loc" : {
                "type" : "Point",
                "coordinates" : [
                        18.9460754,
                        38.2129364
                ]
        },
        "time" : 1,
        "temp2" : 28.65,
        "is_land" : false,
        "winds10" : 3.04,
        "windd10" : 231.45,
        "humid2" : 64.69,
        "slvp" : 1013.2,
        "hcloud" : 0,
        "mcloud" : 0,
        "lcloud" : 0,
        "tprecip" : 0,
        "cprecip" : 0
}
```

Σχήμα 4.4: Εσωτερική σχεδίαση της συλλογής φ' για την περιοχή της Η-πείρου

πρώτη τιμή του πίνακα αντιστοιχεί στις 02:00, η δεύτερη στις 04:00 κ.ο.κ.

Στο Σχήμα 4.4 φαίνεται ίσως ο πιο σημαντικός πίνακας σε όλη την υλοποίηση της εφαρμογής της πρόγνωσης από την πλευρά της εμπειρίας χρήστη και την λειτουργία των φίλτρων. Όπως είπαμε κατά την σχεδίαση η αρχική ιδέα ήταν να υπήρχε ένας πίνακας για κάθε μετεωρολογική παράμετρο γεγονός το οποίο θα είχε κόστος κάθε φορά που θα εκτελούσαμε ένα ερώτημα για ένα σημείο. Οπότε, στον πίνακα Forecast_filters παρατηρούμε ότι για κάθε σημείο έχουμε την τιμή

```
> db.Forecast_filters.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_",
                "ns" : "Forecast.Forecast_filters"
        },
        {
                "v" : 2,
                "key" : {
                        "loc" : 1
                },
                "name" : "loc_1",
                "ns" : "Forecast.Forecast_filters"
        },
        {
                "v" : 2,
                "key" : {
                        "time" : 1
                },
                "name" : "time_1",
                "ns" : "Forecast.Forecast_filters"
        },
        {
                "v" : 2,
                "key" : {
                        "loc" : "2dsphere"
                },
                "name" : "loc_2dsphere",
                "ns" : "Forecast.Forecast_filters"
```

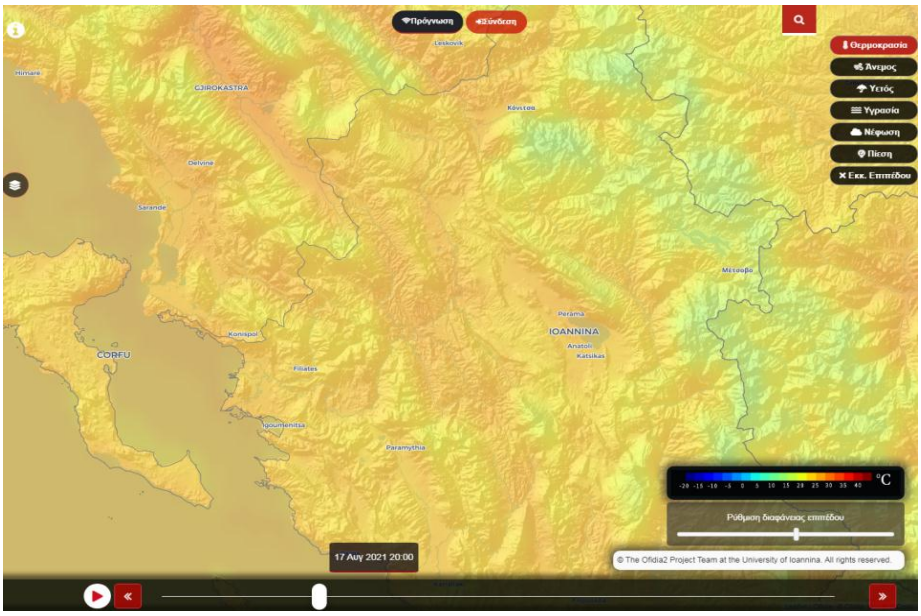
Σχήμα 4.5: Ευρετήρια για την συλλογή των φίλτρων

της χρονοσφραγίδας (time), την τοποθεσία του σημείου (location), για κάθε μετεωρολογική παράμετρο την τιμή της (temp2, windd10,windz10, hcloud ...) και αν το σημείο είναι στεριά ή όχι (is_land).

Στο Σχήμα 4.5 παρουσιάζουμε τα ευρετήρια που έχουμε δημιουργήσει για τον πίνακα Forecast_filters. Τα δύο σημαντικότερα εξ' αυτών είναι το χωρικό ευρετήριο και το χρονικό ευρετήριο τα οποία όπως είπαμε κατά την σχεδίαση παίζουν πολύ σημαντικό ρόλο στην ταχύτητα απόκρισης των λειτουργιών της εφαρμογής.

4.2 Υλοποίηση τελικής εφαρμογής για την πρόγνωση

Σε αυτή την υποενότητα θα παρουσιάσουμε την τελική εφαρμογή η οποία ανήκει σε ένα ευρύτερο πλαίσιο εφαρμογών για υπηρεσία πρόληψης πυρκαγιών στην περιοχή της Ηπείρου.



Σχήμα 4.6: Στιγμιότυπο από την εφαρμογή απεικόνισης της πρόγνωσης

Στο Σχήμα 4.6 φαίνεται ένα τμήμα της τελικής εφαρμογής που έχουμε αναπτύξει. Το πιο σημαντικό είναι ότι παρουσιάζεται με τον πιο ευδιάκριτο τρόπο η αποτύπωση του γεωγραφικού χώρου μέσω του χάρτη καθώς και η αποτύπωση του χρόνου μέσω της οριζόντιας γραμμής επιλογής χρονοσφραγίδας.

Επιπλέον, σε αυτό το στιγμιότυπο της εφαρμογής δείχνουμε την απεικόνιση της μετεωρολογικής παραμέτρου της θερμοκρασίας για την περιοχή ενδιαφέροντος. Να τονιστεί ότι με παρόμοιο τρόπο γίνεται και η απεικόνιση των υπόλοιπων μετρικών που μας ενδιαφέρουν. Για να γίνει αυτή η απεικόνιση εφικτή απαιτούνται οι εξής προϋποθέσεις:

- Χρήση ενός χάρτη ουδέτερου χρώματος για την σωστή αποτύπωση του επιπέδου της μετεωρολογική παράμετρος
- Παραγωγή των επιπέδων για κάθε χρονική στιγμή όχι απευθείας αλλά μέσω ενός script αφού γίνει η εισαγωγή των νέων δεδομένων από την εκτέλεση του μετεωρολογικού μοντέλου

Για την παραγωγή των απεικονίσεων κάθε μετεωρολογική παράμετρος χρησιμοποιούμε τον πίνακα Forecast_{όνομα_μετεωρολογική παράμετρος}

```
//Ανάκτηση των δεδομένων παραμέτρου για την χρονική στιγμή time
client = MongoClient('localhost', 27017)
db = client.ForecastEpirus

counter=0
latitudesInList = []
longitudesInList= []
valuesInList= []

temp = db.Forecast_temp2 // για την θερμοκρασία
cursor = temp.find()

for doc in cursor:
        latitudesInList.append(doc["loc"]["coordinates"][1])
        longitudesInList.append(doc["loc"]["coordinates"][0])
        valuesInList.append(doc["daily_values"][time])

……..

// Δημιουργία των απεικονίσεων
cmap = plt.get_cmap(cmap1)
levels = MaxNLocator(nbins=15).tick_values(low_value, high_value)
norm = BoundaryNorm(levels, ncolors=cmap.N, clip=True)
for i in range(0,162):
     for j in range(0,135):
         if(vv[i,j]<low_value or vv[i,j]>high_value):
                vv[i,j]=np.nan
plt.pcolor(yy, xx, vv, cmap=cmap ,vmin=low_value,vmax=high_value, ra
sterized=True, snap=True)
main_path = "/home/gbakos/Forecast/Epirus/"
if(os.path.exists(main_path+folder)):

    fig.savefig(main_path+folder+'/'+str(time)+'.png', dpi=100,trans
```

Σχήμα 4.7: Υλοποίηση δημιουργίας απεικονίσεων δεδομένων

Στο Σχήμα 4.7 δείχνουμε τα δύο πιο σημαντικά τμήματα του κώδικα. Το ένα είναι η ανάκτηση των δεδομένων από την βάση δεδομένων που έχουμε δημιουργήσει και το άλλο είναι η δημιουργία του επιπέδου μέσω της συνάρτησης pcolor της βιβλιοθήκης matplotlib της python.

Σημαντικά στοιχεία που έχουμε προσθέσει στην υλοποίηση της δημιουργίας του επιπέδου είναι τα εξής:

- Η κάθε απεικόνιση τοποθετείται στον φάκελο με το όνομα της μετεωρολογικής παραμέτρου και σαν όνομα στο κάθε αρχείο ορίζουμε τη τιμή της χρονοσφραγίδας (1.png).

Σημαντικά στοιχεία που έχουμε προσθέσει στην υλοποίηση της δημιουργίας του επιπέδου είναι τα εξής:

- Το χρωματικό μοντέλο αποτελείται από 15 χρώματα και η διαβαθμίσεις γίνονται με βάση την ελάχιστη και μέγιστη τιμή την οποία καθορίζουμε για κάθε μετεωρολογική παράμετρο
- Σε περίπτωση που δεν υπάρχει κάποιος φάκελος στον οποίο πρέπει να αποθηκευτεί μία απεικόνιση τον δημιουργούμε εκείνη τη στιγμή

Σχήμα 4.8: Εμφάνιση πληροφοριών με ένα κλικ

Στο Σχήμα 4.8 παρουσιάζουμε την κυριότερη λειτουργία που έχουμε αναπτύξει στην εφαρμογή της πρόγνωσης που δεν είναι άλλη από την εμφάνιση των πληροφοριών για ένα σημείο ενδιαφέροντος όποτε πατάμε το κλικ στο σημείο

στον χάρτη. Μόλις ο χρήστης πατήσει οπουδήποτε στον χάρτη σε ένα σημείο αυτό που γίνεται είναι το εξής:

- Αναζήτηση με χρήση του ευρετηρίου του πιο κοντινού σημείου που αποθηκεύουμε στον πίνακα Forecast_filters.
- Αναζήτηση με χρήση λογισμικού geolocation των πληροφοριών της περιοχής όπως χώρα, περιφέρεια, δήμος, κοινότητα

```
first_location = db['Forecast_filters'].find({
    'loc'=>
        {'$near'=>
            {
                  '$geometry'=>
                    {'type'=>'Point','coordinates'=>[params[:long].t
o_f,params[:lat].to_f]},
                   '$maxDistance'=>1413
               }
```

Σχήμα 4.9: Κώδικας ερωτήματος για τα φίλτρα

Στο Σχήμα 4.9 με χρήση του προγραμματιστικού περιβάλλοντος Ruby-on-Rails έχουμε δημιουργήσει της μεθόδους που θα επιστρέφουν τις απαιτούμενες πληροφορίες στην εφαρμογή χρήστη. Η συγκεκριμένη υλοποίηση έχει την λογική API κλήσεων όπου κάθε μέθοδος επιστρέφει δεδομένα σε μορφή JSON. Στον κώδικα παρουσιάζεται η αναζήτηση του κοντινότερου σημείου σε αυτό που έχουμε επιλέξει μέσω του χάρτη (params[:long].to_f, params[:lat].to_f]), (γεωγραφικό μηκος, γεωγραφικό πλάτος). Με χρήση της παραμέτρου $near της mongodb αναζητάμε σε μια περιοχή με ακτίνα 1413μ το κοντινότερο σημείο που έχουμε αποθηκευμένο. Η παράμετρος $near λειτουργεί μόνο σε περίπτωση που έχει δημιουργηθεί το ευρετήριο για την τοποθεσία. Ο λόγος που χρησιμοποιούμε αυτή την απόσταση είναι διότι τα σημεία μεταξύ τους απέχουν 2 χιλιόμετρα, οπότε η μέγιστη απόσταση στην οποία μπορεί να εμφανιστεί ένα σημείο είναι sqrt(2) = 1,412... . σε αυτή την υλοποίηση υπάρχει μία ιδιαιτερότητα. Ενδέχεται ένα σημείο που έχουμε επιλέξει με βάση το κριτήριο της ακτίνας που τέθηκε να είναι κοντά σε δύο σημεία για τα σημεία δημιουργούν κυκλικές περιοχές οι οποίες τέμνονται.

Οπότε, κάνουμε μία επιπλέον διαδικασία μέσω της οποίας βρίσκουμε το πιο κοντινό από τα δύο που ενδέχεται να προκύπτουν.

Σημαντικό στοιχείο στην παραπάνω υλοποίηση είναι η χρήση της συνάρτησης sort με βάση την χρονοσφραγίδα. Επειδή έχουμε δημιουργήσει το ευρετήριο για την χρονοσφραγίδα τα δεδομένα ανακτώνται ταξινομημένα μέσω αυτής της συνάρτησης.

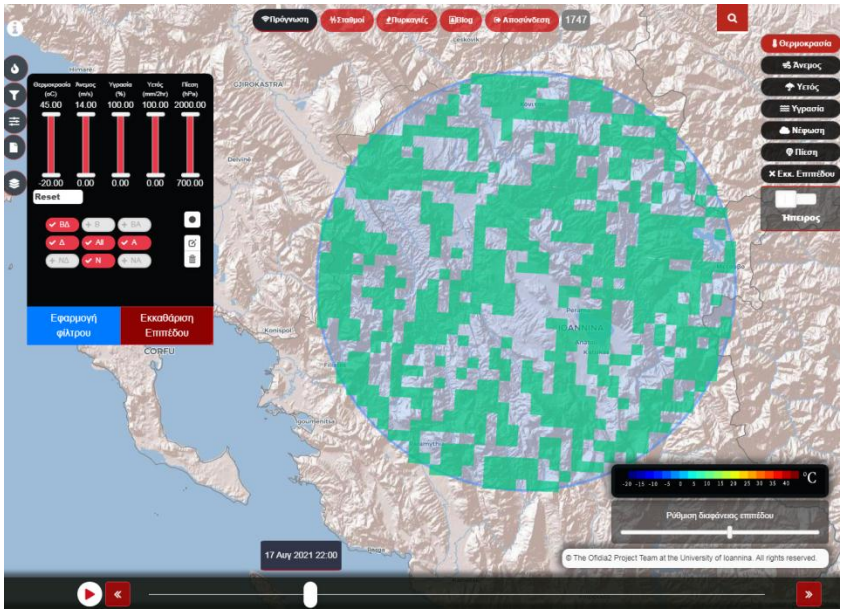

```
city: "Κουκλιοί"
data: {temp: Array(36), winds10: Array(36), windd10: Array(36),
precip: Array(36), humid: Array(36), …}
indexes: {Canadian_FWI: Array(1), Canadian_FWI_BUI: Array(1), Cana-
dian_FWI_DC: Array(1), Canadian_FWI_DMC: Array(1), Canadi-
an_FWI_FFMC: Array(1), …}
windLevels: {Forecast_winddz1: Array(1), Forecast_winddz2: Array(1),
Forecast_winddz3: Array(1), Forecast_winddz4: Array(1), Fore-
cast_winddz5: Array(1), …}
```


Σχήμα 4.10: Δεδομένα επιστροφής από κλήση API

Στο σχήμα 4.10 παρουσιάζουμε τις πληροφορίες που επιστρέφονται μετά την εκτέλεση της μεθόδους για το σημείο που επιλέχθηκε. Διακρίνουμε τις εξής πληροφορίες:

- Το σημείο που επιλέχθηκε με χρήση λεκτικού
- Τα δεδομένα για κάθε μετεωρολογική παράμετρο
- Τα δεδομένα για την επικινδυνότητα της πυρκαγιάς στο σημείο αυτό
- Τα δεδομένα για τα διαφορετικά επίπεδα του ανέμου

Στο Σχήμα 4.11 βλέπουμε την λειτουργία των φίλτρων με βάση την επιλογή του χρήστη με γεωγραφικό και χρονικό κριτήριο. Με την βοήθεια αυτής της λειτουργίας ένας χρήστης μπορεί να επιλέξει ένα εύρος από σημεία και κάποιες από τις συνθήκες που θέλει να επαληθεύονται και θα λάβει άμεσα το αποτέλεσμα. Το αποτέλεσμα περιέχει όλα τα σημεία εκείνα για τα οποία επαληθεύονται τα κριτήρια του χρήστη. Στα κριτήρια μπορεί να είναι οποιαδήποτε μετεωρολογική παράμετρο από τις υπάρχουσες στην συλλογή δεδομένων forecast_filters.



Σχήμα 4.11: Απεικόνιση χειροκίνητων φίλτρων φίλτρων κινδύνου πυρκαγιάς

Υπάρχει μόνο ένας περιορισμός σε αυτή τη λειτουργία ο οποίος έχει να κάνει με την βιβλιοθήκη που χρησιμοποιούμε σχετικά με την απεικόνιση των σημείων στον χάρτη. Ο περιορισμός που τίθεται με βάση την εμπειρία είναι ότι το μέγιστο πλήθος σημείων που μπορεί να εφαρμοστεί το φίλτρο είναι γύρω στα 4000 σημεία. Ενώ ο χρόνος που εκτελείται το ερώτημα είναι στα πλαίσια του επιτρεπτού, απεικόνιση καθυστερεί πάρα πολύ αν υπάρχει μεγάλος αριθμός σημείων οπότε από την πλευρά της εφαρμογής έχουμε θέσει ένα άνω όριο το οποίο δεν χρειάζεται να το υπερβεί.

Στο Σχήμα 4.12 δείχνουμε τον κώδικα που εκτελούμε ούτως ώστε να επιστρέφονται τα φιλτραρισμένα δεδομένα με βάση τα κριτήρια του χρήστη. Και σε αυτή την περίπτωση χρησιμοποιούμε την εντολή $near με παραμέτρους την ακτίνα του κύκλου πού έχει σχηματίσει ο χρήστης και κέντρο την επιλογή του χρήστη.

```
data = db['ForecastEpirus.filters'].find({
'$and'=>[
      {'temp2'=>{'$gte' => params[:tempDown].to_f,'$lte' => params[:
tempUp].to_f}},

       {'winds10'=>{'$gte' => params[:windDown].to_f,'$lte' => param
s[:windUp].to_f}},
      {'hmid2'=>{'$gte' => params[:humidDown].to_f,'$lte' => params[
:humidUp].to_f}},
      {'cprec'=>{'$gte' => params[:precipDown].to_f,'$lte' => params
[:precipUp].to_f}},
      {'slvp'=>{'$gte' => params[:pressureDown].to_f,'$lte' => param
s[:pressureUp].to_f}},
      {'time'=>{'$eq' => params[:time].to_i}},
      {'loc'=>{
         '$near'=>{
               '$geometry'=>{
                    'type'=>'Point',
                     'coordinates'=>
                               [params[:centerAreaLng].to_f,
                               params[:centerAreaLat].to_f]},
                               '$maxDistance'=>params[:radiusArea].
to_f
                          }
                     }},
```

Σχήμα 4.12: Ερωτήματα για φιλτράρισμα με βάση τα δοθέντα κριτήρια

4.3 Υλοποίηση αποθήκευσης δεδομένων μετεωρολογικών σταθμών

Σε αυτή την ενότητα θα παρουσιάσουμε την διαχείριση της αποθηκευτικής δυνατότητας για την εφαρμογή των μετεωρολογικών σταθμών με βάση την σχεδίαση που προτείναμε στην αντίστοιχη ενότητα. Όπως είπαμε και στο κεφάλαιο της σχεδίασης για την διαχείριση των λειτουργιών των σταθμών έχουμε αναπτύξει διαφορετικές τεχνικές οι οποίες έχουν να κάνουν με την πολυπλοκότητα των λειτουργιών που θέλουμε να υποστηρίξουμε.

4.3.1 Αποθήκευση πρόσφατων δεδομένων μετρήσεων

Όπως είπαμε και στην ενότητα της σχεδίασης, για να αποφύγουμε να έχουμε όλα τα δεδομένα σε μία συλλογή όπως έρχονται τα δεδομένα από τους αισθητήρες , κάνουμε αποσύνδεση των δεδομένων με βάση τις λειτουργίες.

```
> show tables
meteo.humidity
meteo.pressure
meteo.temperature
ofidiaJS.Tsoil_10
ofidiaJS.Tsoil_20
ofidiaJS.Tsoil_30
……..
ofidiaJS.Tsoil_60
ofidiaJS.UVA
……

stations.stationsOfidia.data
stations.stationsMeteo.data
```

Σχήμα 4.13: Διαχωρισμός αποθήκευσης συλλογών δεδομένων μετεωρολογικών σταθμών

Στο Σχήμα 4.13 παρουσιάζουμε την δομή των συλλογών στην βάση δεδομένων των σταθμών. Παρατηρούμε ότι έχουμε μία συλλογή για κάθε μετεωρολογική παράμετρο και τύπο σταθμού. Επίσης, έχουμε δημιουργήσει και δύο συλλογές οι οποίες για κάθε τύπο σταθμού αποθηκεύουμε μόνο τα πρόσφατα δεδομένα από

τους σταθμούς (stations.stationsMeteo.dataRecent, stations.ofidia.dataRecent). επιπλέον, σε άλλες δύο συλλογές αποθηκεύουμε όλα τα δεδομένα που λαμβάνουμε από τους σταθμούς σε περίπτωση που κάτι δεν πάει καλά να υπάρχει εφικτός τρόπος για ανάκτηση των δεδομένων (stations.stationsOfidia.data,..).

```
> db.stations.stationsMeteo.data.findOne()
{
        "_id" : ObjectId("5cf7857099f6102eeacc39a5"),
        "temperature" : 21,
        "sid" : 231,
        "humidity" : 70,
        "pressure" : 1014,
        "time" : ISODate("2019-06-05T11:20:00Z"),
        "windD" : "NA",
        "wind" : 1
}
```

Σχήμα 4.14: Εσωτερική σχεδίαση της συλλογής αποθήκης όλων των δεδομένων τρίτων σταθμών

Στο Σχήμα 4.14 παρουσιάζουμε την εσωτερική δομή της μόνιμης αποθήκης όλων των δεδομένων των σταθμών που προέρχονται από 2 από τις 3 πηγές. Παρατηρούμε ότι τα δεδομένα τοποθετούνται με βάση την χρονοσφραγίδα. Το sid προσδιορίζει με κάποιο τρόπο το χωρικό στοιχείο αφού κάθε σταθμός αντιστοιχεί σε ένα σημείο στον χώρο. Αυτή την συλλογή την χαρακτηρίζουμε ως βοηθητική μιας και όπως είπαμε πριν γίνεται χρήση της μόνο αν πάει κάτι στραβά στην εκτέλεση των λειτουργιών.

Κατά αντιστοιχία έχουμε υλοποιήσει και την αποθήκευση των μόνιμων δεδομένων ασφαλείας για τους σταθμούς του δεύτερου τύπου. Επίσης, με παρόμοιο τρόπο αλλά μόνο με τα πρόσφατα δεδομένα, έχει υλοποιηθεί και η σχεδίαση για την αποθήκευση δεδομένων της τελευταίας ημέρας.

4.3.2 Αποθήκευση ιστορικών δεδομένων μετεωρολογικών σταθμών

Σε αυτή την υποενότητα θα παρουσιάσουμε την υλοποίηση των συλλογών σχετικών με τις λειτουργίες απεικόνισης ιστορικών στοιχείων. Όπως είπαμε και στην ενότητα της σχεδίασης έχουμε αποσυνδέσει την λειτουργία των ιστορικών στοιχείων από τα πρόσφατα. Κυρίως αυτή τη σχεδίαση την χαρακτηρίζει η ομαδοποίηση των δεδομένων με βάση κάποια ολότητα όπως είναι η μέρα ή ο μήνας. Εμείς επιλέξαμε την ολότητα της ημέρας.

```
> db.meteo.temperature.findOne()
{
        "_id" : ObjectId("5e80873d286ddbd79edd3c75"),
        "date" : ISODate("2019-06-05T23:50:00Z"),
        "sid" : 231,
        "values" : [
                {
                        "time" : ISODate("2019-06-05T11:20:00Z"),
                        "value" : 21
                },
            …….
      ]
      "min" : 15,
        "max" : 22,
        "avg" : 18.384615384615383
}
```

Σχήμα 4.15: Σχεδίαση της συλλογής αποθήκευσης των δεδομένων μίας παραμέτρου

Στο Σχήμα 4.15 παρατηρούμε την εσωτερική δομή της συλλογής για τα ιστορικά στοιχεία των σταθμών. Ο πίνακας values περιέχει για μία μέρα τις τιμές για κάθε χρονοσφραγίδα σε ταξινομημένη σειρά ούτως ώστε να ανακτώνται ήδη ταξινομημένα. Επίσης, για να μην έχουμε μία πρόσθετη καθυστέρηση στην εφαρμογή υπολογίζοντας την μέγιστη τιμή, ελάχιστη τιμή και τον μέσο όρο κάνουμε τον υπολογισμό του σε χρόνο που φτιάχνουμε όλη τη δομή.

Στο Σχήμα 4.16 παρατηρούμε άλλη μία εκδοχή για την υλοποίηση της συλλογής των ιστορικών στοιχείων η οποία είναι λειτουργιο-κεντρική. Γι' αυτό το λόγο

```
> db.meteoJS.temperature.findOne()
{
        "_id" : ObjectId("5e8215b1ba739bd607bbbab4"),
        "date" : ISODate("2019-06-05T23:50:00Z"),
        "sid" : 231,
        "values" : [
                "[Date.UTC(2019,5,5,11,20),21],",
                "[Date.UTC(2019,5,5,11,50),21],",
                "[Date.UTC(2019,5,5,12,20),21],",
      ],
        "min" : 15,
        "max" : 22,
        "avg" : 18.384615384615383
}
```

Σχήμα 4.16: Σχεδίαση της συλλογής αποθήκευσης δεδομένων μίας παραμέτρου (δεύτερη εκδοχή)

δεν αποτελεί μια ευέλικτη μορφή αποθήκευσης παρά μόνο για την απεικόνισης γραφημάτων που έχουμε εισάγει και χρησιμοποιούμε στην εφαρμογή μας. Σε περίπτωση που χρησιμοποιηθεί άλλη βιβλιοθήκη δεν είναι βέβαιο ότι η υλοποίηση αυτή θα λειτουργήσει με τα αναμενόμενα αποτελέσματα.

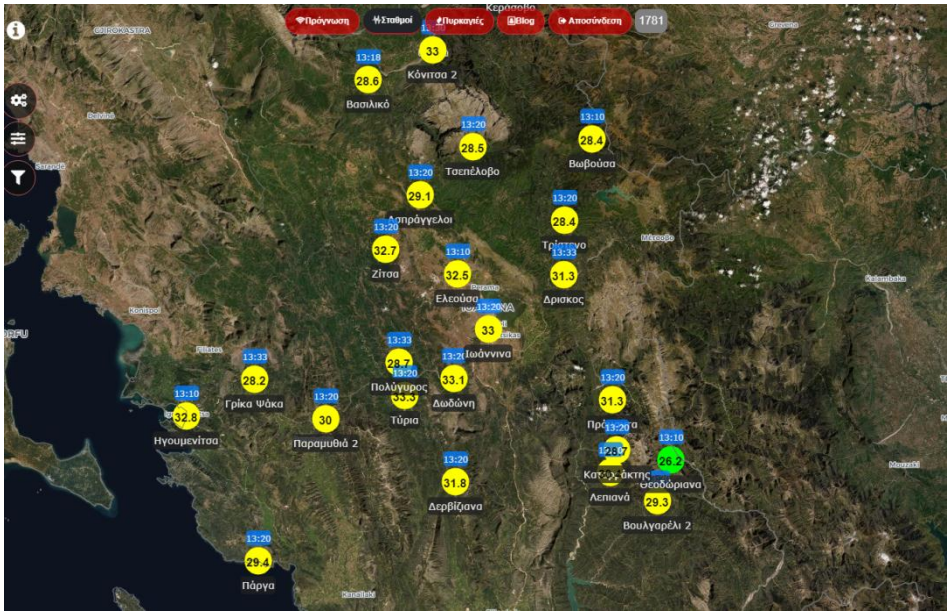
4.4 Υλοποίηση τελικής εφαρμογής για τους μετεωρολογικούς σταθμούς

Στην υποενότητα αυτή θα παρουσιάσουμε την τελική εφαρμογή χρήστη για την διαχείριση δεδομένων από μετεωρολογικούς σταθμούς οι οποίοι είναι σημεία στο χώρο και τα δεδομένα τους χρονοσειρές με τιμές. Θα δούμε πως η τελική εφαρμογή επηρεάζει την σχεδίαση των διαφόρων συλλογών που σχεδιάστηκαν. Αν και οι περισσότερες συλλογές που σχεδιάστηκαν είναι γενικού σκοπού σε πο-

λύ μεγάλο βαθμό σχεδιάστηκαν για να ταιριάζουν στα δεδομένα που απαιτεί η τελική εφαρμογή.

4.4.1 Υλοποίηση λειτουργιών για τις πρόσφατες μετρήσεις

Η υλοποίηση της εφαρμογής για την απεικόνιση των πρόσφατων μετρήσεων βασίζεται στην απλή ιδέα: όλοι οι σταθμοί να είναι ορατοί και να φαίνεται η πρόσφατη τιμή τους.



Σχήμα 4.17: Εφαρμογή απεικόνισης μετεωρολογικών σταθμών

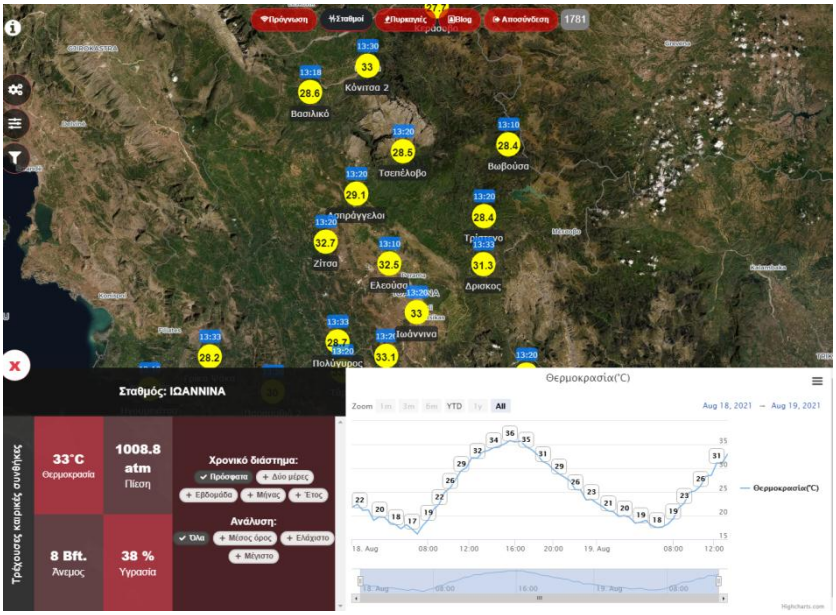
Στο Σχήμα 4.17 φαίνεται με τον καλύτερο τρόπο η απεικόνιση της λειτουργίας αυτής. Κάθε σταθμός-σημείο στον χάρτη εμφανίζεται σαν μία κουκίδα με την τιμή και την χρονοσφραγίδα.

Για να γίνει αυτό εφικτό απαιτείται να γίνει κλήση της μεθόδου η οποία επιστρέφει τα δεδομένα στην εφαρμογή.

```
db= Mongo::Client.new(['10.7.2.39:27017'], :database => "Stations")
data=db['stations.stationsMeteo.dataRecent'].find()
puts data
render :json => {"data": data}
```

Σχήμα 4.18: Εκτέλεση ερωτήματος για πρόσφατα δεδομένα

Στο Σχήμα 4.18 φαίνεται η εκτέλεση του απλού ερωτήματος το οποίο επιστρέφει τα δεδομένα από την συλλογή των πρόσφατων δεδομένων stations.stationsMeteo.dataRecent. Ο λόγος που το ερώτημα αυτό είναι τόσο απλό



Σχήμα 4.19: Αποθήκευση πρόσφατων πληροφοριών σταθμού

έγκειται στην στοχευμένη υλοποίηση της δομής της συλλογής των πρόσφατων δεδομένων. Εφόσον τα δεδομένα που επιστρέφονται είναι ταξινομημένα για να εμφανίσουμε την τελευταία τιμή απλά εμφανίζουμε το τελευταία στοιχείο των τιμών και χρονοσφραγίδων.

Με τον ίδιο τρόπο και με την κλήση της ίδιας μεθόδου εμφανίζουμε τα στοιχεία για ένα σταθμό από την ίδια συλλογή. Η διαφορά είναι ότι υπάρχει το κριτήριο του αναγνωριστικού του σταθμού.

4.4.2 Υλοποίηση λειτουργιών για την απεικόνιση ιστορικών δεδομένων

Όπως φαίνεται στο Σχήμα 4.19 ο χρήστης μπορεί να επιλέξει ένα εύρος διαθέσιμων λειτουργιών. Μπορεί να χρειαστεί να ανακτήσει και να ερευνήσει ιστορικά στοιχεία της τελευταίας εβδομάδας, μήνα, έτους κ.α. καθώς και παραμετροποιημένα φίλτρα. Για να γίνει αυτό εφικτό γίνονται οι κατάλληλες κλήσεις σε μεθόδους που ανακτούν τα δεδομένα από τις συλλογές των ιστορικών στοιχείων.

Στο Σχήμα 4.20 φαίνεται η κλήση της μεθόδου ανάκτησης των ιστορικών δεδομένων από την αντίστοιχη συλλογή. Η επιλογή της συλλογής καθορίζεται από την επιλογή της μετεωρολογικής παραμέτρου που απαιτεί ο χρήστης να ανακτήσει δεδομένα για παράδειγμα την θερμοκρασία. Τα υπόλοιπα κριτήρια που θέτει ο χρήστης τοποθετούνται ως παράμετροι στο ερώτημα που εκτελείται. Επειδή έχουμε δημιουργήσει ευρετήριο χρονικό η εκτέλεση που γίνεται μας επιτρέπει να έχουμε τα δεδομένα ταξινομημένα ανά ημερομηνία. Επίσης, αν κάποιος χρήστης επιλέξει ως κριτήριο να ανακτηθούν μέσοι όροι, μέγιστες τιμές και ελάχιστες αυτό είναι εφικτό μιας και η ίδια η συλλογή έχει έτοιμα προς ανάκτηση αυτά τα δεδομένα. να θυμηθούμε ότι έχουμε εισάγει στο σχήμα μας την ελάχιστη τιμή, την μέγιστη τιμή και τον μέσο όρο για να ανακτώνται πολύ γρήγορα χωρίς καμία επεξεργασία.

```
mids_to_metrics = {8=>"temperature",10=>"humidity",11=>"wind",14=>"p
ressure"}
mids_to_metrics_gr = {8=>"Θερμοκρασία(C)",10=>"Υγρασία(%)",11=>"Άνεμ
ος",14=>"Πίεση(hPa)"}
analysis_to_en = {1 => "all" ,2=>"average" ,3=>"min", 4=>"max"}
analy-
sis_to_el = {1 => "Όλα" ,2=>"Μέσος όρος" ,3=>"Ελάχιστές τιμές", 4=>"
Μέγιστες τιμές"}

today = DateTime.now()
start_day = DateTime.new(today.year,today.month,today.day)

db = Mongo::Client.new(['10.7.2.39:27017'], :database => "StationsNe
w")
collection = "meteo.".concat(mids_to_metrics[mid])
data=db[collection].find({'$and'=>[
        {'sid'=>{'$eq'=>station.to_i}},
```

Σχήμα 4.20: Κλήση μεθόδου για ιστορικά δεδομένα σταθμών

ΚΕΦΑΛΑΙΟ 5

Πειραματα Αξιολογηση

5.1 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή της πρόγνωσης

5.2 Αποτελέσματα από την χρήση της εφαρμογής πρόγνωσης

5.3 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή των μετεωρολογικών σταθμών

5.4 Αποτελέσματα από την χρήση της εφαρμογής των μετεωρολογικών σταθμών

Σε αυτό το σημείο θα παρουσιάσουμε μερικά πειράματα που πραγματοποιήσαμε πάνω στις διάφορες λειτουργίες της βάσης δεδομένων και της εφαρμογής γενικότερα συμπεράσματα πάνω στην εκτέλεση της εφαρμογής που δημιουργήσαμε. Τα πειραματικά αποτελέσματα προκύπτουν από την χρήση της εφαρμογής μέσω ενός γνωστού περιηγητή ιστού σε Windows 10 με σύνδεση στα 1-10Mbps (σχετικά αργή σύνδεση) σε μηχάνημα με επεξεργαστή Intel Core i5 στα 2,7GHz με μνήμη Ram 8GB και με έναν δίσκο SSD 480GB. Επίσης, έγιναν πειρά-

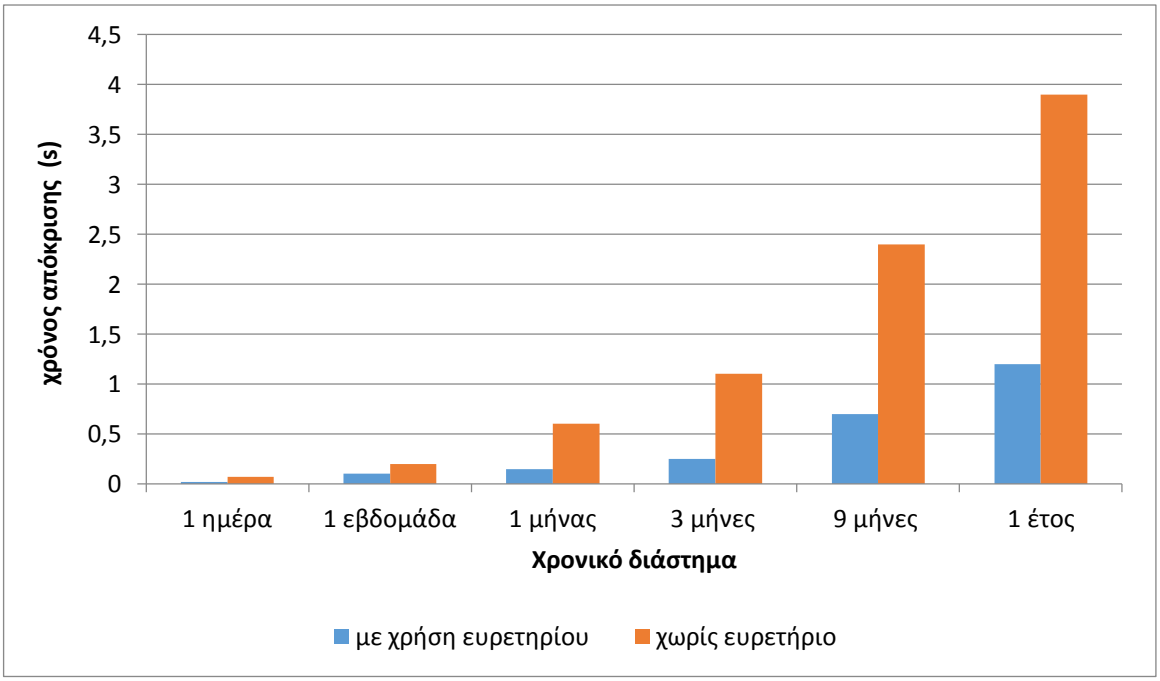
ματα πάνω σε μηχάνημα με λειτουργικό σύστημα Linux (Debian10), τετραπύρηνο επεξεργαστή στα 2.50 GHz, μνήμη RAM στα 6 GB RAM, και με έναν δίσκο 1 x 250 GB 7.2KRPM SATA. Το μηχάνημα αυτό είχε εγκατεστημένοο το σύστημα της βάσης δεδομένων και τον διακομιστή της εφαρμογής. Τα πειράματα και τα ερωτήματα έγιναν πάνω στην διασύνδεση που προσφέρει η βιβλιοθήκη PyMongo η οποία αναλαμβάνει να διαχειρίζεται συνδέσεις και ερωτήματα με τον εξυπηρετητή της MongoDB. Επίσης, έγιναν μετρήσεις κατευθείαν πάνω στην βάση δεδομένων με χρήση των ερωτημάτων που προσφέρει η MongoDB.

5.1 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή της πρόγνωσης

Για τις λειτουργίες της εφαρμογής της πρόγνωσης έγιναν μια σειρά από μετρήσεις που αφορούν τις σχεδιαστικές προτάσεις που έχουμε εισάγει στο σύστημά μας.

5.1.1 Πείραμα 1: Χρήση του ευρετηρίου

Στο σημείο αυτό παρουσιάζουμε το πρώτο πείραμα που κάνουμε το οποίο ελέγχει την απόδοση του συστήματος με κριτήριο την ύπαρξη ή μη ευρετηρίων πάνω στην χρονική διάσταση των δεδομένων. Πιο συγκεκριμένα, εκτελούμε λειτουργίες να δούμε κατά πόσο το ευρετήριο χρονοσειρών που έχουμε προσθέσει



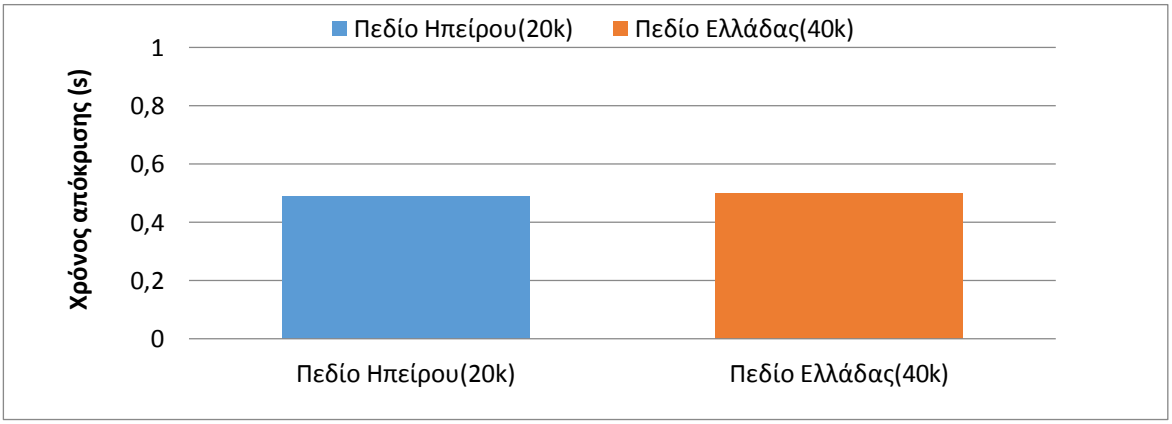
Σχήμα 5.1: Μετρήσεις πειραμάτων στην εφαρμογή ιστορικού πρόγνωσης πάνω σε δύο συλλογές με ευρετήριο και χωρίς αντίστοιχα

στη συλλογή με το ιστορικό πολλών δεδομένων παίζει κάποιο σημαντικό ρόλο. Πιο συγκεκριμένα, εκτελούμε το ερώτημα στην συλλογή δεδομένων της θερμοκρασίας για παραμετροποιήσιμο χρονικό εύρος για σημείο ενδιαφέροντος [39,12875, 20,3491].

Στο Σχήμα 5.1 παρατηρούμε το γράφημα με τους χρόνους εκτέλεσης ερωτημάτων ιστορικών δεδομένων πάνω στην εφαρμογή της πρόγνωσης. Υπάρχει μία ιδιαιτερότητα στην επεξεργασία των δεδομένων στις δύο περιπτώσεις. Στην πρώτη περίπτωση με την χρήση ευρετηρίου όταν εκτελείται ένα ερώτημα με χρονικό εύρος και απαιτείται χρονική ταξινόμηση, η λύση είναι τετριμμένη και άμεση διότι η χρήση του ευρετηρίου επιτρέπει να έρθουν τα δεδομένα ήδη ταξινομημένα. Στην δεύτερη περίπτωση η εγγύηση αυτή δεν υφίσταται, οπότε απαιτείται επιπλέον υπολογιστικό κόστος να ελεγχθεί και να ταξινομηθεί το αποτέλεσμα του ερωτήματος. Το επιπλέον κόστος που προκύπτει κάνει μη εφικτή την λύση χωρίς ευρετήριο οπότε παρατηρούμε ότι χρήση του ευρετηρίου είναι υποχρεωτική.

5.1.2 Πείραμα 2: Απόκριση ανάλογα με το πλήθος επιστρεφόμενων τιμών

Στο δεύτερο πείραμα που παρουσιάζουμε εξετάζουμε τους χρόνους εκτέλεσης των ερωτημάτων πάνω στην λειτουργία της πρόσφατης πληροφορίας για ένα σημείο στον χάρτη. Η λειτουργία αυτή, να θυμήσουμε, ανακτά πληροφορίες από την δομή δεδομένων ***Forecast_filters.*** Μας ενδιαφέρει να ελέγξουμε κατά πόσο θα επηρεαστεί ο χρόνος εκτέλεσης του ερωτήματος αυτού αν το πλήθος των δεδομένων που αποθηκεύονται σχεδόν διπλασιαστεί.



Σχήμα 5.2: Μετρήσεις πειραμάτων στην εφαρμογή ιστορικού πρόγνωσης πάνω σε δύο συλλογές διαφορετικών μεγεθών

Αυτό θα γίνει με τον εξής απλό τρόπο: μιας και απαιτείται μία συλλογή διπλάσια σε μέγεθος από την συλλογή που χρησιμοποιούμεθα χρησιμοποιήσουμε ως σύγκριση την συλλογή που προκύπτει από την αποθήκευση του δεύτερου πεδίου (Ελλάδα).

Στο Σχήμα 5.2 παρατηρούμε τα αποτελέσματα που λάβαμε από πειράματα όσον αφορά το πλήθος το σημείων που έχουμε να αποθηκεύσουμε. Δοκιμάζουμε μετρήσεις στις 2 συλλογές που απαντώνται στο σύστημα μας με τη μία να έχει διπλάσιο μέγεθος από την άλλη. Μετά από 100 επαναλήψεις πάνω σε διαφορετικά σημεία κάθε φορά παρατηρούμε ότι η διαφορά στον χρόνο εκτέλεσης είναι πολύ-πολύ μικρή – στο συγκεκριμένο παράδειγμα μπορεί να αγγίζει και το όριο του σφάλματος εκτέλεσης. Η διαφορά που παρατηρούμε είναι της τάξης του 0.1 δευτερολέπτου ενώ έχουμε διπλασιασμό της συλλογής. Αυτό που συμπεραίνουμε είναι ότι το μέγεθος της συλλογής δεν παίζει κάποιο σημαντικό ρόλο στην απόδοση των ερωτημάτων. Θα είχε ενδιαφέρον αν δημιουργούσαμε συλλογές που αντί για 20k και 40k να είχαν 500k σημεία, αλλά στα πλαίσια της εφαρμογής μας είναι αρκετή μια τέτοια διερεύνηση.

5.1.3 Πείραμα 3: Χρόνος απόκρισης της διαδικτυακής εφαρμογής σε σχέση με το ερώτημα

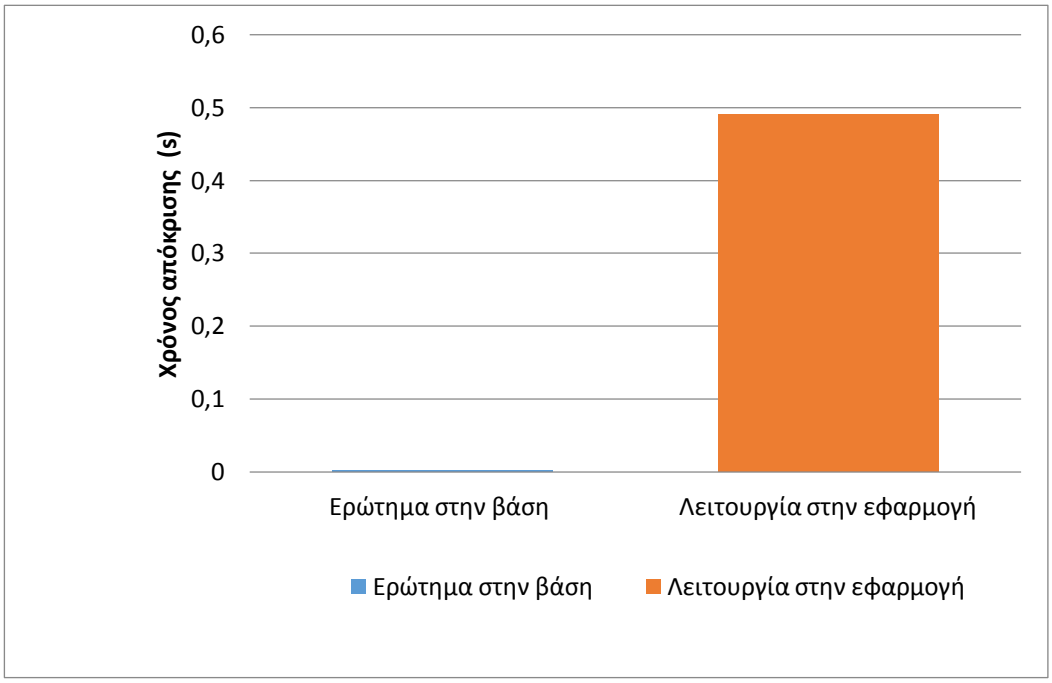
Σε αυτό το πείραμα θα δείξουμε πώς επηρεάζεται ο χρόνος απόδοσης των λειτουργιών από την επιπλέον επιβάρυνση που εισάγει η απεικόνιση των πληροφοριών μέσα από την διαδικτυακή εφαρμογή. Για να γίνει αυτό εφικτό κάνουμε την εξής διαδικασία:

```
db.ForecastEpirus.filters.find({ loc: {
      $near: {
          $geometry:{
               type: "Point", coordinates: [20.14,39.15]
          },
          $maxDistance:1413
       }
}}).sort({time:1}).explain("executionStats").executionStats.executio
```

Σχήμα 5.3: Ερώτημα για την ανάκτηση των πρόσφατων πληροφοριών σημείου

- Μέτρηση 1: εκτελούμε ένα ερώτημα μέσω της διεπαφής απευθείας στην βάση δεδομένων
- Μέτρηση 2: εκτελούμε μία ενέργεια που αφορά το ίδιο ερώτημα αλλά μέσα από την τελική εφαρμογή αυτή τη φορά.

Η πρώτη μέτρηση αφορά το ερώτημα στο σχήμα 5.3.



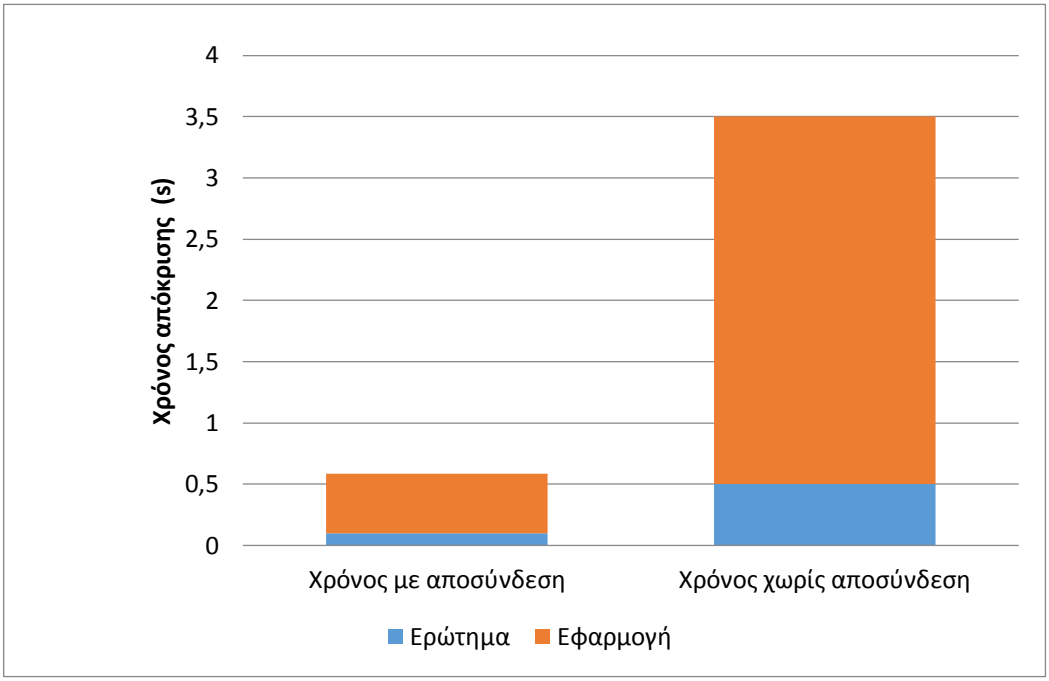
Σχήμα 5.4: Ανάκτηση πρόσφατων δεδομένων σημείου από την εφαρμογή ή απευθείας από την βάση δεδομένων

Στο Σχήμα 5.4 παρατηρούμε ο χρόνος εκτέλεσης του ερωτήματος κατευθείαν πάνω στη βάση είναι 0,002s ενώ ο χρόνος απεικόνισης στον χρήστη είναι 0,49s.

Συμπεραίνουμε ότι η μεγαλύτερη επιβάρυνση στην εφαρμογή μας προκύπτει από την επεξεργασία των πληροφοριών και το πώς θέλουμε να εμφανίζονται στην εφαρμογή και από την ταχύτητα του δικτύου. Επίσης, η επεξεργασία γίνεται από κάποιες επιμέρους βιβλιοθήκες επεξεργασίας γεωγραφικών δεδομένων οι οποίες επιβάλλουν το επιπλέον κόστος.

5.1.4 Πείραμα 4: αποσύνδεση των συλλογών των πρόσφατων και ιστορικών δεδομένων

Στο πείραμα αυτό θα προσπαθήσουμε να εξετάσουμε την σχεδίαση που προτείνουμε με την αποσύνδεση των συλλογών στα πρόσφατα και στα ιστορικά δεδομένα. Θα προσπαθήσουμε να δούμε την διαφορά στους χρόνους εκτέλεσης που θα είχε η εφαρμογή σε περίπτωση που δεν κάναμε την αποσύνδεση που προτείνουμε.

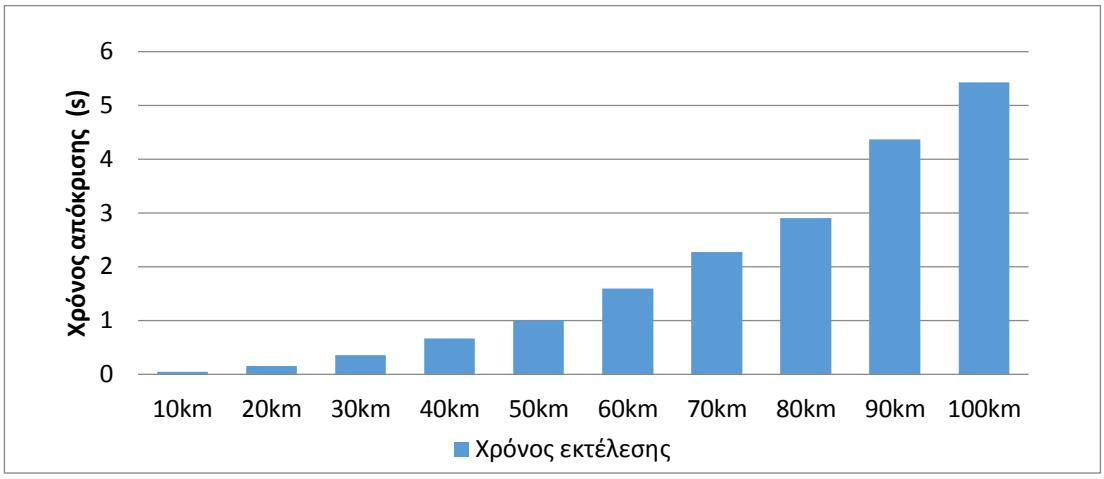


Σχήμα 5.5: Ανάκτηση πρόσφατων δεδομένων σε αποσυνδεδεμένη σχεδίαση αποθήκευσης

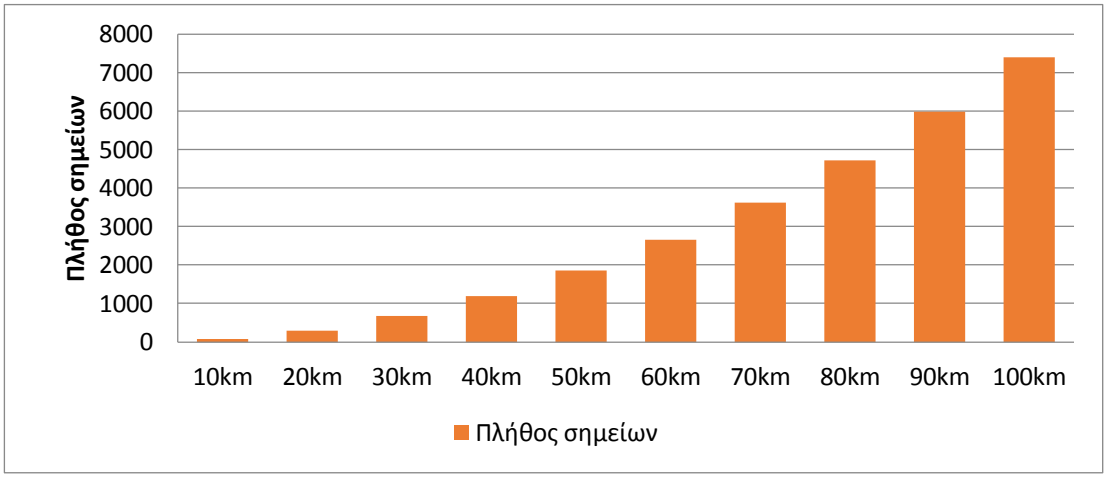
Στο Σχήμα 5.5 παρατηρούμε ότι οι χρόνοι εκτέλεσης της ίδιας λειτουργίας χρησιμοποιώντας την σχεδίαση της αποσυνδεδεμένης δομής είναι κατά πολύ μικρότεροι από το να μην χρησιμοποιούσαμε αυτή την τεχνική. Παρατηρούμε ότι ο χρόνος εκτέλεσης του ερωτήματος είναι αρκετά μικρός στην περίπτωση της αποσυνδεδεμένης τεχνικής αλλά και στην δεύτερη εκδοχή παραμένει μικρός. Παρόλα αυτά η επιβάρυνση που θέτει η τελική εφαρμογή κάνει την δεύτερη τεχνική μη αποδεκτή όσον αφορά την άμεση απόκριση και την ελάχιστη καθυστέρηση που παρατηρεί ο χρήστης.

5.1.5 Πείραμα 5: Διερεύνηση πλήθους σημείων φίλτρου με βάση την ακτίνα επιλογής

Στο πείραμα αυτό θα εξετάσουμε την απόδοση των χειροκίνητων φίλτρων κινδύνου πυρκαγιάς που έχουμε εισάγει στην γενικότερη εφαρμογή της πρόγνωσης. Μας απασχολεί όπως σε κάθε λειτουργία να έχουμε άμεση απόκριση στην τελική εφαρμογή. γι' αυτό διενεργούμε αυτό το πείραμα για να δούμε πιο είναι εκείνο το πλήθος σημείων το οποίο μπορεί να θεωρηθεί ανώφλι στην εφαρμογή μας.



Σχήμα 5.6: Χρόνος εκτέλεσης για την ανάκτηση των δυναμικών δεδομένων με βάση την ακτίνα επιλογής



Σχήμα 5.7: Πλήθος σημείων που ανακτώνται από την χρήση των δυναμικών δεδομένων με βάση την ακτίνα επιλογής

Στα Σχήματα 5.6 και 5.7 παρατηρούμε τα αποτελέσματα από τις μετρήσεις

σχετικά με την εκτέλεση των δυναμικών φίλτρων και των ερωτημάτων που σχετίζονται με αυτά. Παρατηρούμε ότι όσο μεγαλύτερη ακτίνα επιλέγουμε στα δυναμικά φίλτρα τόσο περισσότερο καθυστερεί και η ανάκτηση των δεδομένων.

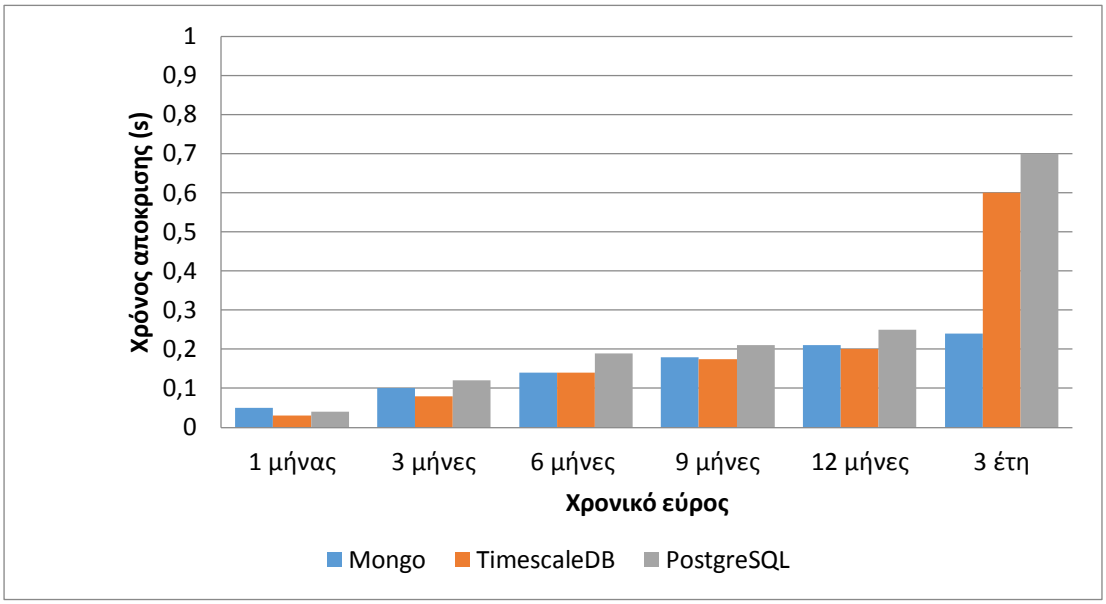
Με βάση το πείραμα αυτό αποφασίστηκε να υπάρξει ένα άνω όριο όσον αφορά τα σημεία που μπορούμε να ανακτήσουμε με μία επιλογή,. Το άνω όριο αυτό είναι τα 80km ακτίνα, επιλογή η οποία μας δίνει μέγιστο αριθμό σημείων περίπου 5000 και χρόνο εκτέλεσης 3 δευτερολέπτων.

5.2 Πειραματική ανάλυση των λειτουργιών για την εφαρμογή των μετεωρολογικών σταθμών

Στην υποενότητα αυτή θα παρουσιάσουμε κάποια πειράματα που έγιναν πάνω στις δομές δεδομένων που έχουμε προτείνει και σχεδιάσει όσον αφορά την λειτουργία των μετεωρολογικών σταθμών.

5.2.1 Πείραμα 1: Μετρήσεις πάνω στην συλλογή με ίδια δομή αλλά σε διαφορετική μηχανή αποθήκευσης

Το συγκεκριμένο πείραμα που περιγράφουμε σε αυτή την παράγραφο αποτελεί μία σημαντική αφετηρία για την επιλογή της αποθηκευτικής μηχανής που θέλουμε να αποθηκεύσουμε τα δεδομένα μας. Η συγκεκριμένη πειραματική υλοποίηση έγινε στο πλαίσιο της συνεργασίας με τους κ. Θεοφάνη Μπαλάσκα και Ευάγγελο Δημουλή.



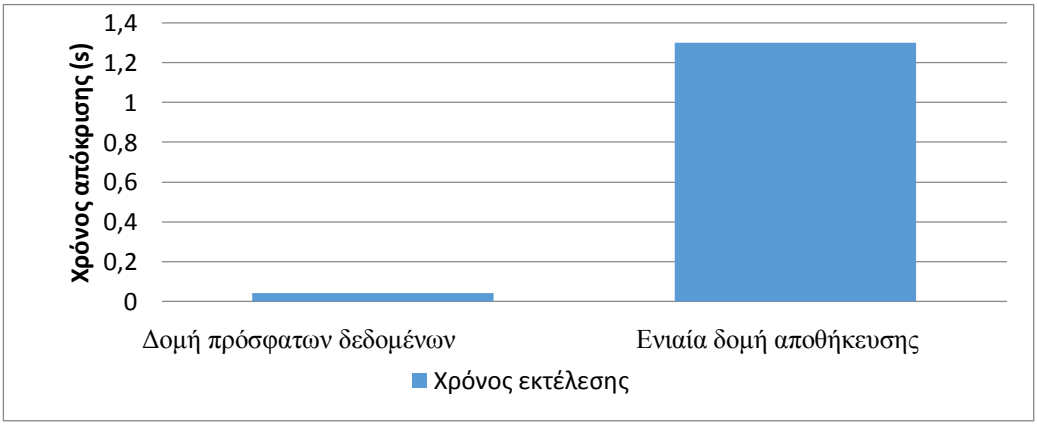
Σχήμα 5.8: Μετρήσεις πειραμάτων στην εφαρμογή μετεωρολογικών σταθμών με διαφορετικές μηχανές αποθήκευσης

Στο σχήμα 5.8 παρατηρούμε ότι για το ιστορικό των μετεωρολογικών σταθμών για 1 μέχρι 12 μήνες η TimescaleDB έχει ελαφρώς καλύτερη απόδοση αλλά πολύ μικρή διαφορά σε σχέση με την MongoDB. Στην συγκεκριμένη υλοποίηση μας ενδιέφερε η γρήγορη ανάκτηση ακόμα και στην χειρότερη περίπτωση που θα απαιτήσει ο χρήστης να ανακτηθεί σχετικά γρήγορα. Έτσι, όταν κάποιος απαιτή-

σει δεδομένα 3 ετών η MongoDB έχει καλύτερη απόδοση. Οι παραπάνω χρόνοι είναι οι χρόνοι εκτέλεσης των ερωτημάτων από κάθε διεπαφή του ίδιου του συστήματος βάσεων δεδομένων και όχι από την εφαρμογή. Τονίζουμε ότι για την απεικόνιση από την εφαρμογή απαιτείται και ένα επιπλέον κόστος το οποίο συνήθως μπορεί να είναι διπλάσιο ή τριπλάσιο από τον χρόνο εκτέλεσης στη βάση δεδομένων.

5.2.2 Πείραμα 2: Μελέτη χρόνων εκτέλεσης πρόσφατων δεδομένων

Στο συγκεκριμένο πείραμα θα εξετάσουμε κατά πόσο παίζει ρόλο η αποσύνδεση που κάνουμε στην σχεδίαση των δομών δεδομένων ως προς τις επιμέρους λειτουργίες σχετικά με την ανάκτηση των πρόσφατων δεδομένων.



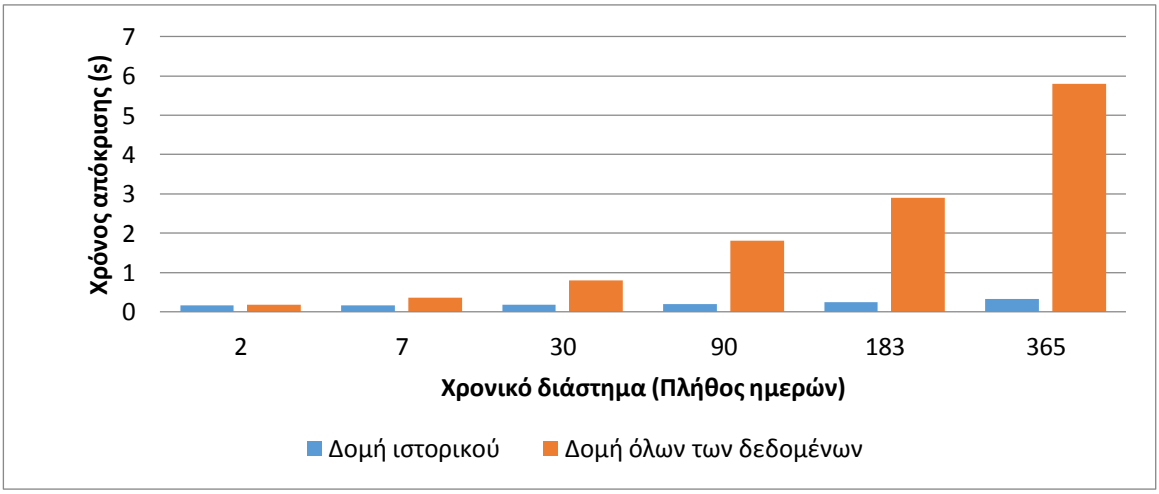
Σχήμα 5.9: Χρόνος εκτέλεσης ερωτημάτων πρόσφατων δεδομένων με χρήση των διαφορετικών εκδοχών

Στο Σχήμα 5.9 παρατηρούμε την μεγάλη διαφορά στον χρόνο εκτέλεσης στις δύο διαφορετικές περιπτώσεις που έχουμε χρήση ή μη της αποσυνδεδεμένης λογικής που προτείναμε. Ο λόγος της μεγάλης καθυστέρησης που προκύπτει από την χρήση της πρώτης απλής εκδοχής είναι διότι για κάθε αίτηση που κάνουμε θα πρέπει να ανακτώνται δεδομένα από διαφορετικές συλλογές της βάσης δεδομένων. Οι χρόνοι αυτοί αποτελούν τους χρόνους εκτέλεσης με ερωτήματα χωρίς περαιτέρω επεξεργασία που θα γίνει στην τελική εφαρμογή.

5.2.3 Πείραμα 3: Μελέτη χρόνου εκτέλεσης ιστορικών δεδομένων

Στο πείραμα αυτό θα εξετάσουμε την εκτέλεση των επιμέρους λειτουργιών των ιστορικών δεδομένων ενός σταθμού μέσα από το πλαίσιο των δύο εκδοχών

που προτείνουμε. Η πρώτη εκδοχή είναι να γίνει εκτέλεση με βάση τη νέα σχεδίαση όπου τα ιστορικά δεδομένα ομαδοποιούνται με βάση την ημέρα και η δεύτερη εκδοχή είναι να χρησιμοποιηθούν τα δεδομένα από την συλλογή τη οποίας η δομή μοιάζει με αυτή των πρόσφατων δεδομένων.



Σχήμα 5.10: Χρόνος εκτέλεσης ερωτημάτων πρόσφατων δεδομένων με χρήση των διαφορετικών εκδοχών

Στο σχήμα 5.10 παρατηρούμε την χρονική εξέλιξη της απόδοσης των ερωτημάτων χρονικού εύρους για έναν σταθμό για τον οποίο θέλουμε να ανακτήσουμε πληροφορίες. Στο συγκεκριμένο παράδειγμα, αιτούμαστε δεδομένα της θερμοκρασίας αέρα για πολλά χρονικά εύρη. Ξεκινάμε από χρονικό εύρος 2 ημερών μέχρι 1 έτος. Αυτό που παρατηρούμε είναι ότι με την χρήση της ανεξάρτητης δομής για τα ιστορικά δεδομένα σχετικά με την θερμοκρασία η απόδοση που πετυχαίνουμε ακόμα και στην χειρότερη περίπτωση του ενός έτους είναι αρκετά μικρή, πολύ μικρότερη του 1s. Σε αντίθεση, αν και εδώ δεν είχαμε κάνει την αποσύνδεση των ιστορικών δεδομένων σε νέα δομή αλλά χρησιμοποιούσαμε την δομή στην οποία αποθηκεύονται όλα τα δεδομένα τότε η απόδοση θα ήταν μη αποδεκτή.

ΚΕΦΑΛΑΙΟ 6

ΕΠΙΛΟΓΟΣ

Σε αυτό το κεφάλαιο θα συνοψίσουμε τα αποτελέσματα σχετικά με τις μετρήσεις που προέκυψαν από τα πειράματα του κεφαλαίου 5. Θα δούμε κατά πόσο τα αποτελέσματα από τα πειράματα ικανοποίησαν τους αρχικούς στόχους οι οποίοι είχαν τεθεί.

Η πιο κρίσιμη λειτουργία της εφαρμογής της πρόγνωσης είναι η άμεση απόκριση δεδομένων που αφορούν ένα σημείο το οποίο επιλέγει ο χρήστης. Για αυτή την λειτουργία η εφαρμογή ιστού αποκρίνεται σε περίπου 500ms. Ο χρόνος αυτός είναι ικανός να θεωρηθεί άμεσα αποκρίσιμη η λειτουργία αυτή.

Επίσης, για λειτουργίες ιστορικών δεδομένων πάνω σε ένα σημείο ο χρόνος είναι λίγο μικρότερος του 1s. Παρόλο τον μεγάλο όγκο των δεδομένων που απαιτείται σε περίπτωση που ζητηθεί να απεικονιστεί το ιστορικό, ο χρόνος είναι αρκετά μικρός για να μην χάσει ο χρήστης το ενδιαφέρον του από την εφαρμογή. Ο λόγος που έχουμε αυτή την ταχύτατη απόκριση σε αυτές τις δύο διαφορετικές λειτουργίες είναι διότι κάναμε αποσύνδεση των δύο λειτουργιών από τη βάση τους, γεγονός το οποίο απέδωσε ικανοποιητικά.

Όσον αφορά την εφαρμογή της διαχείρισης των μετεωρολογικών σταθμών τα αποτελέσματα από τα πειράματα δείχνουν ότι η σχεδίαση και η υλοποίηση οδη-

γεί στους επιθυμητούς στόχους που έχουμε θέσει. Πιο συγκεκριμένα, η πρώτη λειτουργία που παρατηρεί ο χρήστης μόλις ανοίξει την εφαρμογή των μετεωρολογικών σταθμών είναι η απεικόνιση των πιο πρόσφατων μετρήσεων για την μετεωρολογική παράμετρο της θερμοκρασίας. Για αυτή τη λειτουργία ο χρόνος εκτέλεσης της είναι περίπου 400ms. Ενώ, για την λειτουργία απεικόνισης της πρόσφατης πληροφορίας για έναν σταθμό ο χρόνος απόκρισης είναι 100ms.

Όσον αφορά τις πιο απαιτητικές λειτουργίες και σε αυτή την περίπτωση ο χρόνος είναι αρκετά μικρός. Για παράδειγμα, αν ζητηθεί να απεικονιστούν τα δεδομένα για ένα έτος ενός σταθμού ο χρόνος απόκρισης είναι περίπου 3s. Ο χρόνος αυτός είναι μικρός αν σκεφτεί κανείς ότι τα δεδομένα που επιστρέφουν είναι περίπου 20000 μετρήσεις.

Τέλος, όπως και στην εφαρμογή της πρόγνωσης ο λόγος που η ταχύτητα απόκρισης είναι αρκετά μεγάλη είναι διότι έχουμε αποσυνδέσει τα δεδομένα και την διαχείριση τους με βάση την λειτουργία που απαιτείται.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] MongoDB, https://www.mongodb.com.

[2] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, Theo Vassilakis, Dremel: Interactive Analysis of Web-Scale Datasets, International Conference on Very Large Data Bases, 2010, Singapore, pp 330-339

[3] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Bigtable: A Distributed Storage System for Structured Data, USENIX Symposium on Operating Systems Design and Implementation, 2006, Seattle USA, pp. 205-218

[4] Ilari Shafer, Raja R. Sambasivan, Anthony Rowe, Gregory R. Ganger, 'Specialized Storage for Big Numeric Time Series', USENIX Workshop on Hot Topics in Storage and File Systems, 2013, San Jose CA

[5] Anand Padmanabha Iyer, Ion Stoica, A Scalable Distributed Spatial Index for the Internet-of-Things, USENIX Symposium on Cloud Computing, 2017, Santa Clara CA, pp 548-560

[6] Qifan Pu, Shivaram Venkataraman, Ion Stoica, Shuffling, Fast and Slow: Scalable Analytics on Serverless Infrastructure, USENIX Symposium on Networked Systems Design and Implementation, 2019, Boston USA, pp 193-206

[7] Matplotlib, https://matplotlib.org/

[8] Windy, *windy.com*

[9] Ajla Kirlić, Muhedin Hadžić, Big data and time series: A literature review paper

[10] Timescale, https://www.timescale.com/

[11] Leaflet: an open-source JavaScript library for mobile-friendly interactive maps, https://leafletjs.com/

[12] Windfinder.com, *https://www.windfinder.com*

[13] WRF (Weather Research and Forecasting Model), *https://www.mmm.ucar.edu/weather-research-and-forecasting-model*

[14] TimescaleDB, *https://www.timescale.com/*

Σύντομο Βιογραφικο

Από το 2018 είναι μέλος του Εργαστηρίου Συστημάτων Υπολογιστών στο Τμήμα Μηχανικών Η/Υ & Πληροφορικής του Πανεπιστημίου Ιωαννίνων. Το 2018 εισήχθη στο ΠΜΣ του τμήματος. Το 2017 αποφοίτησε από το τμήμα Τμήμα Μηχανικών Η/Υ & Πληροφορικής του Πανεπιστημίου Ιωαννίνων με βαθμό «Λίαν καλώς». Από τότε ασχολείται με την δημιουργία προγραμματιστικών υποδομών τόσο ερευνητικών όσο και εφαρμοσμένων που έχουν σκοπό την καλύτερη εμπειρία του τελικού χρήστη. Κύρια ενδιαφέροντα του αποτελούν η σχεδίαση αποδοτικών εφαρμογών, η υλοποίηση αποθηκευτικών υποδομών που φιλοξενούν μεγάλα δεδομένα, η δημιουργία τεχνικών για την καλύτερη αξιοποίηση των διαθέσιμων πόρων των υπολογιστικών υποδομών.