**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ**

**ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ**

**ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ**

**ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ**

**ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ**

**ΟΜΑΔΙΚΗ ΕΡΓΑΣΙΑ**

**Ομάδα 36**

|  |  |  |
| --- | --- | --- |
| **Ονοματεπώνυμο:** | ΑΝΤΩΝΙΟΣ ΜΠΕΛΕΣΗΣ | ΚΩΝΣΤΑΝΤΙΝΟΣ ΑΝΑΓΝΩΣΤΑΚΗΣ |
| **Αριθμός Μητρώου:** | 1066445 | **1066439** |
| **Τμήμα:** | ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ | ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ |
| **Έτος φοίτησης:** | 6ο | 6ο |

Περιεχόμενα

[1 Εννοιολογικό Μοντέλο 2](#_Toc155778036)

[2 Σχεσιακό Μοντέλο 3](#_Toc155778037)

[3 Εφαρμογή για επικοινωνία με τη βάση δεδομένων 4](#_Toc155778038)

[4 Ευρετήρια 4](#_Toc155778039)

# Εισαγωγή

Ο σκοπός της παρούσας εργασίας ήταν ο σχεδιασμός βάσης δεδομένων για εφαρμογή αποθήκευσης και ανάκτησης άρθρων και επιστημονικών περιοδικών.

Ακολουθώντας τις τεχνικές που διδαχτήκαμε στο μάθημα ξεκινήσαμε με τον σχεδιασμό του εννοιολογικού μοντέλου (ERD) βάσει των απαιτήσεων που χρειάζεται να πληροί η εφαρμογή και συνεχίσαμε με τον μετασχηματισμό του σε σχεσιακό σχήμα. Έχοντας ορίσει τους απαραίτητους πίνακες προχωρήσαμε στον προγραμματισμό της βάσεις με εντολές SQL και έπειτα από εισαγωγή δεδομένων και έλεγχο ορθότητας των πινάκων σχεδιάσαμε την εφαρμογή με χρήση της γλώσσας **Python**. Σκοπός της εφαρμογής είναι η προσομοίωση λειτουργιών που προβλέπεται να εκτελεί η βάση μέσα από ένα απλό γραφικό περιβάλλον από την οπτική ενός χρήστη.

# Εννοιολογικό Μοντέλο

Ο μικρόκοσμος της βάσης είναι ο εξής: Η εφαρμογή αποτελείται από χρήστες οι οποίοι αναζητούν και διαβάζουν δημοσιεύσεις. Οι δημοσιεύσεις αυτές μπορεί να είναι είτε άρθρα ή τεύχη περιοδικού. Το κάθε τεύχος περιέχει προφανώς άρθρα, με την παραδοχή ότι ένα άρθρο μπορεί να εκδοθεί το πολύ σε ένα περιοδικό. Επίσης τα άρθρα μπορούν να αναφέρουν άλλα άρθρα. Μία επιπλέον παραδοχή είναι ότι τα περιοδικά έχουν ακριβώς έναν εκδότη ενώ τα άρθρα μπορούν να έχουν έναν ή περισσότερους συγγραφείς. Ο συγγραφέας καθώς και ο χρήστης μπορούν να συνεργάζονται με κάποιο ή περισσότερα ιδρύματα (Πανεπιστήμιο, Εταιρία ή Κολλέγιο).

Ο χρήστης πρέπει να μπορεί να αναζητήσει ένα άρθρο με βάση keywords, αναφορές από ή σε άλλα άρθρα, επιστημονικό πεδίο και συγγραφέα. Επίσης να μπορεί να βρει τεύχος περιοδικού που έχει εκδοθεί κάποιο άρθρο και όλα τα άρθρα που περιέχονται σε κάποιο τεύχος καθώς και το αντίστροφο, καθώς και να καταγράφονται οι αναφορές άρθρων σε άλλα άρθρα και ο συνολικός αριθμός αναφορών σε κάθε άρθρο. Επιπλέον η ημερομηνία έκδοσης του κάθε άρθρου αλλά και η ημερομηνία που εντάχθηκε στη βάση δεδομένων. Τέλος, να καταγράφεται η προβολή/ανάγνωση των άρθρων από τους χρήστες και ο χρήστης να μπορεί να αποθηκεύσει μία δημοσίευση για ανάγνωση αργότερα καθώς και να ακολουθήσει συγγραφείς.

Ο κάθε χρήστης θα έχει ένα Boolean γνώρισμα «verified» το οποίο αν είναι αληθές του δίνεται η δυνατότητα να ανεβάσει άρθρο και να ορίσει ο ίδιος συγγραφέα και όλα τα στοιχεία του άρθρου. Η παραδοχή για τα τεύχη των περιοδικών είναι ότι καταχωρούνται στη βάση σε εταιρικό επίπεδο και όχι σε επίπεδο εφαρμογής.

Εικόνα που περιέχει διάγραμμα, κύκλος

Περιγραφή που δημιουργήθηκε αυτόματα

Εικ1. Το διάγραμμα ERD της βάσης δεδομένων

Βασική απαίτηση της εφαρμογής είναι να επιτρέψει στον χρήστη την εύκολη περιήγηση ανάμεσα σε δημοσιεύσεις, άρθρα ή τεύχη μέσω των κλειδιών και γνωρισμάτων που αναφέρθηκαν παραπάνω. Εκτός αυτού πρέπει να εμφανίζονται τα ίδια τα γνωρίσματα αυτών αλλά και οι οντότητες που συνδέονται μαζί τους όπως συγγραφέας ή περιοδικό και εκδότης αντίστοιχα. Για ένα άρθρο συγκεκριμένα απαραίτητη είναι και η εμφάνιση άλλων άρθρων τα οποία αναφέρονται από αυτό, ενώ θα πρέπει να είναι δυνατή και η εύρεση του με την χρήση των keywords που το χαρακτηρίζουν. Για ένα τεύχος εκτός από το περιοδικό στο οποίο περιέχεται θα πρέπει να εμφανίζονται και τα άρθρα που ενδεχομένως περιλαμβάνει. Τέλος, θα πρέπει να καταγράφεται η προβολή και ανάγνωση των δημοσιεύσεων και να παρέχεται στον χρήστη η δυνατότητα αποθήκευσης δημοσιεύσεων ή συγγραφέων.

# Σχεσιακό Μοντέλο

Έπειτα έγινε η μετατροπή του εννοιολογικού μοντέλου οντοτήτων-συσχετίσεων σε σχεσιακό μοντέλο (Εικ. 2). Αυτό αποτελείται από 16 πίνακες συνολικά. Ο σχεδιασμός έγινε με την εφαρμογή dbdesigner και οι πίνακες με τα πεδία τους δημιουργήθηκαν σε SQLite με την εφαρμογή DB Browser.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, διάγραμμα, σχεδίαση

Περιγραφή που δημιουργήθηκε αυτόματα

Εικ2. Οι πίνακες του σχεσιακού μοντέλου της βάσης

Ίσως ο πιο βασικός πίνακας είναι ο πίνακας PUBLISHMENT (Δημοσίευση) ο οποίος έχει ως πρωτεύον κλειδί τη στήλη “id” ο οποίος είναι ένας τυχαίος ακέραιος. Στην αρχική υλοποίηση (ενδιάμεση παρουσίαση) είχε προταθεί ως ένα σύνθετο κλειδί ο συνδυασμός των γνωρισμάτων “url” και “file\_address” τα οποία λαμβάνουν εκ φύσεως μοναδικές τιμές, αλλά στην πράξη αποδείχτηκε μη-αποδοτικό καθώς θα έπρεπε να χρησιμοποιούμε και τις δύο τιμές σαν ξένα κλειδιά για κάθε πίνακα που τα χρησιμοποιεί, δηλαδή στους περισσότερους πίνακες.

Ο πίνακας επιπλέον έχει ως ξένα κλειδιά το όνομα του εκδότη (υποθέτουμε ότι κάθε εκδότης θα έχει μοναδικό όνομα) καθώς και τη διεύθυνση ηλεκτρονικού ταχυδρομείου του συγγραφέα του. Και τα δύο αυτά πεδία μπορεί να πάρουν την τιμή NULL εφόσον δεν έχουμε αυτά τα στοιχεία. Επιπλέον υπάρχει πεδίο για το ποιο πρότυπο γραφής ακολουθεί καθώς και την χρονική στιγμή την οποία ανέβηκε στη βάση το άρθρο.

Τέλος έχει την στήλη “views” η οποία αναφέρεται στον αριθμό των αναγνώσεων που έχει η δημοσίευση. Αυτή ενημερώνεται για κάθε νέα εισαγωγή στον πίνακα “reads” χρησιμοποιώντας ένα trigger. Ένα παρόμοιο Trigger έχουμε και για να μετράμε τον αριθμό αναφορών (citations) σε κάθε άρθρο χρησιμοποιώντας τον πίνακα CITES.

Όπως αναφέραμε και στο Εννοιολογικό Μοντέλο, οι δημοσιεύσεις χωρίζονται σε Άρθρα και σε τεύχη επιστημονικών περιοδικών. Η βασική τους διαφορά είναι ότι το άρθρο έχει ως ξένο κλειδί το πρωτεύον κλειδί του τεύχους καθώς κάθε τεύχος περιέχει από ένα έως πολλά άρθρα.

Τέλος, εισαγάγαμε ψεύτικα δεδομένα στην βάση μας για να ελέγξουμε την λειτουργία της. Υπάρχουν συνολικά περίπου 2500 δημοσιεύσεις εκ των οποίων οι 2300 είναι άρθρα και οι υπόλοιπες τεύχη περιοδικών. Τα ίδια τα περιοδικά είναι μόνο 50. Στη βάση καταγράφονται 300 συγγραφείς εκ των οποίων οι 200 περίπου συνεργάζονται με κάποιο ίδρυμα και περίπου 1000 χρήστες από τους οποίους ανήκουν σε κάποιο ίδρυμα οι 150. Για να είναι αληθοφανής ο αριθμός των «προβολών» σε κάθε άρθρο υπάρχουν συνολικά 800 χιλιάδες αναγνώσεις στην βάση. Ιδιαίτερο ενδιαφέρον έχει ο πίνακας KEYWORDS ο οποίος αποτελείται μόνο από περίπου 150 ξεχωριστά keywords. Ωστόσο, επειδή όλα τα άρθρα έχουν δεκάδες λέξεις κλειδιά, ο πίνακας αποτελείται από 50 χιλιάδες εγγραφές.

# Εφαρμογή για επικοινωνία με τη βάση δεδομένων

Η υλοποίηση του κώδικα έγινε με γλώσσα Python. Οι κύριες βιβλιοθήκες που χρησιμοποιήθηκαν είναι η *sqlite3* για επικοινωνία με την βάση και η *tkinter* για προγραμματισμό του γραφικού περιβάλλοντος της εφαρμογής.

Αρχικά, ορίζεται η global μεταβλητή για επικοινωνία με την βάση με χρήση της συνάρτησης *create\_connection()*. Αφού γίνει έλεγχος πως η σύνδεση ήταν επιτυχής ξεκινάει η λειτουργία της εφαρμογής

*login\_screen()*

Δημιουργεί το παράθυρο για την οθόνη login στην εφαρμογή με ένα πεδίο κειμένου για να εισάγει ο χρήστης το username. Παρέχεται και η επιλογή signup στην βάση σε περίπτωση που δεν υπάρχει ο χρήστης.

*login()*

Κλίνεται με το πάτημα του κουμπιού “Login” και ελέγχει αν το κείμενο που πληκτρολόγησε ο χρήστης στο πεδίο κειμένου είναι ίδιο με κάποιο από τα usernames στην βάση. Ο έλεγχος γίνεται με την δημιουργία ενός κέρσορα (cursor) που λαμβάνει ένα read query και με την χρήση της εντολής *cursor.fetchall()* επιστρέφονται τα δεδομένα που σύλλεξε από την βάση. Τα δεδομένα αυτά αποθηκεύονται σε μια μεταβλητή. Αρχικά, ελέγχεται αν η μεταβλητή αυτή είναι άδεια και συνεπώς αν ο χρήστης δεν είναι καταχωρημένος στην βάση. Σε περίπτωση που έχει τιμή ο χρήστης είναι καταχωρημένος στην βάση και τα δεδομένα καταχωρούνται σε λίστα για πιο εύκολη πρόσβαση τους. Η τεχνική αυτή χρησιμοποιείται σε κάθε ανάγνωση δεδομένων της βάσης.

*signup()*

Σε περίπτωση που ο χρήστης πατήσει το κουμπί “Sign Up” ανοίγει νέα φόρμα που ζητάει τα απαραίτητα στοιχεία για να καταγραφεί ο χρήστης στη βάση.

*on\_signup\_click()*

Αφού ο χρήστης συμπληρώσει όλα τα πεδία και πατήσει το κουμπί χρησιμοποιείται η μεταβλητή connection για την δημιουργία ενός κέρσορα (cursor) που θα χρησιμοποιηθεί για να εκτελεστεί το query που θα εισάγει τα δεδομένα του χρήστη στην βάση. Μόλις εκτελεστεί το query από τον κέρσορα γίνεται χρήση της εντολής connection.commit() για να οριστικοποιηθεί η αλλαγή στην βάση. Η μέθοδος αυτή επαναλαμβάνεται για κάθε περίπτωση που εισάγονται, διαγράφονται ή ανανεώνονται δεδομένα στην βάση. Παρουσιάζεται σφάλμα αν το email ή το username υπάρχουν ήδη στην βάση. Το πεδίο verified για τον χρήστη στη βάση λαμβάνει αυτόματα την τιμή false και η έγκρισή του πρέπει να γίνει μέσα στη βάση.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

*open\_main\_window()*

Γίνεται κλήση της αφού ο χρήστης κάνει επιτυχώς login ή signup και κλείνει τα προηγούμενα παράθυρα. Δημιουργεί το μενού με τις επιλογές για φιλτράρισμα, αποθήκευση άρθρων ή ανάρτηση άρθρων. Τα διαθέσιμα φίλτρα περιλαμβάνουν αναζήτηση δημοσιεύσεων, περιοδικών ή άρθρων βάσει συγγραφέων και keywords.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

filter\_options()

Έχει τον ρόλο συνάρτησης που μεσολαβεί για να αναθέσει τις κατάλληλες τιμές στις λίστες-μεταβλητές που θα λειτουργήσουν ως ορίσματα για την επόμενη συνάρτηση.

select\_filter()

Δημιουργεί ένα νέο παράθυρο βάσει των ορισμάτων που δόθηκαν από την προηγούμενη συνάρτηση. Στην συνάρτηση αυτή δημιουργούνται μεταβλητές που θα αποτελέσουν όρισματα του query ανάγνωσης που θα εκτελεστεί στη συνέχεια στη βάση. Στο πάνω μέρος του παραθύρου εμφανίζονται checkboxes. Σε αυτά αναγράφονται μερικά χαρακτηριστικά της οντότητας που επέλεξε ο χρήστης τα οποία έχει την επιλογή να μην του εμφανιστούν. Ακολουθούνται από ένα πεδίο στο οποίο ο χρήστης εισάγει κείμενο που περιέχεται στο ID ή στον τίτλο της οντότητας που αναζητά. Τέλος, εμφανίζονται radiobuttons τα οποία αν επιλεχθούν εμφανίζουν τα δεδομένα με τη σειρά που αναγράφεται. Οι εμφωλευμένες συναρτήσεις update\_options() και update\_order() χρησιμοποιούνται για να ενημερώσουν τις μεταβλητές ανάλογα με τις επιλογές checkbuttons και radiobuttons αντίστοιχα του χρήστη.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

*finalize\_filter()*

Ανάλογα με τον τύπο φίλτρου που επέλεξε ο χρήστης και τις τιμές από τις μεταβλητές της προηγούμενης συνάρτησης δημιουργείται το query που θα εκτελεστεί.

*print\_text\_window()*

Λαμβάνοντας το query της προηγούμενης συνάρτησης και μια λίστα που περιλαμβάνει τα επιθυμητά πεδία που θα εμφανιστούν στον χρήστη ανοίγει ένα νέο παράθυρο κειμένου μέσα στο οποίο καταγράφεται το αποτέλεσμα της αναζήτησης. Γίνεται καταγραφή του χρόνου που χρειάστηκε η αναζήτηση.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

*Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματαselect\_keywords()*

Αποτελεί ειδική περίπτωση φιλτραρίσματος καθώς αντίθετα με τις προηγούμενες χρειάζεται η δημιουργία ενός παραθύρου στην οποία αναγράφονται όλα τα keywords που υπάρχουν εντός της βάσης ως κουμπιά. Ο χρήστης έχει την δυνατότητα να επιλέξει πολλά από αυτά αποθηκεύοντας τις τιμές τους σε μια λίστα keywords. Η συνάρτηση *on\_select()* καθιστά τα keyword checkbuttons λειτουργικά ενώ η *add\_selected\_to\_list()* προσθέτει όσα keywords επιλέχθηκαν σε μια λίστα.

*create\_keyword\_query()*

Λαμβάνονται όλα τα keywords από την λίστα keywords της προηγούμενης συνάρτησης και δημιουργείται ένα query τέτοιο ώστε να εμφανίζονται τα άρθρα που περιέχουν όλα τα keywords που επιλέχθηκαν. Έπειτα γίνεται κλήση της print\_text\_window() για να εμφανίσει το ID και τον τίτλο από όσα άρθρα βρέθηκαν.

*save\_publishment()*

Αν από το κύριο παράθυρο γίνει επιλογή του κουμπιού “Save Article” εμφανίζεται ένα παράθυρο με ένα πλαίσιο κειμένου στο οποίο ο χρήστης μπορεί να εισάγει το ID μιας Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματαδημοσίευσης που υπάρχει καταχωρημένη στη βάση. Παρέχονται οι επιλογές Save, Remove, Show my collection.

*save() / remove()*

Λαμβάνεται το ID που έχει εισάγει ο χρήστης καθώς και το email του χρήστη και προστίθεται η αφαιρείται αντίστοιχα από τον πίνακα SAVES της βάσης η σειρά που περιέχει και τις δύο αυτές τιμές σε περίπτωση που πράξη δεν έχει ήδη ολοκληρωθεί.

*show\_saved()*

Λαμβάνεται το email του χρήστη που είναι συνδεδεμένος και με από τον πίνακα SAVES εμφανίζονται όλες οι δημοσιεύσεις που ταιριάζουν με αυτό.

*upload\_article()*

Καλείται από το κύριο παράθυρο μόνο εάν ο χρήστης που είναι συνδεδεμένος είναι verified. Σε αυτή την περίπτωση ανοίγει μια φόρμα προκειμένου ο χρήστης να αναρτήσει άρθρο. Ζητείται να εισάγει τον τίτλο του, την ημερομηνία έκδοσης, το επιστημονικό πεδίο και σε περίπτωση που υπάρχουν το ID άρθρων που κάνει cite. Το κουμπί “Add writer(s)” οριστικοποιεί τα στοιχεία του άρθρου και κάνει κλήση στην επόμενη συνάρτηση.

*add\_writer()*

Εάν συμπληρωθούν όλα τα απαραίτητα πεδία κλείνει την φόρμα για το άρθρο και ανοίγει καινούρια προκειμένου να εισαχθούν οι συγγραφείς του άρθρου και εάν συνεργάζονται με κάποιο ίδρυμα. Η δυνατότητα να προστεθούν πολλοί συγγραφείς δίνεται από το κουμπί “Add Writers” το οποίο με κλήση της εμφωλευμένης συνάρτησης add\_to\_dictionary() προσθέτει τα στοιχεία του συγγραφέα σε ένα λεξικό κάθε φορά που πατιέται. Μόλις ολοκληρωθεί η προσθήκη όλων των συγγραφέων με το κουμπί “Upload” γίνεται κλήση της επόμενης συνάρτησης για εισαγωγή των στοιχείων στη βάση.

*finish\_upload()*

Λαμβάνει όλα τα δεδομένα που εισήγαγε ο χρήστης στις δύο προηγούμενες φόρμες. Αρχικά, δημιουργείται η δημοσίευση στον πίνακα PUBLISHMENT με ένα καινούριο μοναδικό id, url και file path βάσει αυτού, μηδενικά views, την ημερομηνία τη στιγμής της ανάρτησης και ένα σταθερό πρότυπο. Έπειτα, προστίθενται τα δεδομένα στον πίνακα ARTICLE και τον CITES αν έχουν εισαχθεί άρθρα που γίνονται cited. Στη συνέχεια, γίνεται έλεγχος για τους συγγραφείς που εάν δεν υπάρχουν ήδη στην βάση προστίθενται στον πίνακα WRITER. Η ίδια διαδικασία γίνεται και για το INSTITUTION. Tέλος, προστίθενται να πρωτεύοντα κλειδιά στους πίνακες που συσχετίζουν τις παραπάνω οντότητες. Συγκεκριμένα writer mail, article id στον πίνακα WORKS\_ON και inst\_name, writer mail στον πίνακα COOPERATES.

*generate\_id()*

Δημιουργεί ένα μοναδικό οκταψήφιο αριθμό για την δημοσίευση ελέγχοντας πως δεν υπάρχει ήδη στην βάση.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

# Ευρετήρια

Ένα πολύ σημαντικό στοιχείο των βάσεων δεδομένων, ειδικά αυτών με σημαντικό πλήθος δεδομένων, είναι τα ευρετήρια. Σε αυτό το κεφάλαιο εξετάζουμε την επιλογή ευρετηρίων που έγινε στο σχεδιασμό της βάσης μας.

Για να δούμε τα αποτελέσματα δημιουργήθηκαν 4 αντίγραφα της βάσης, τα οποία διαφέρουν μονάχα στο πλήθος των εγγραφών PUBLISHMENT, καθώς είχαν 10 χιλιάδες, 100 χιλιάδες, 500 χιλιάδες 1 εκατομμύριο και 1.5 εκατομμύρια εγγραφές αντίστοιχα, μαζί με την αρχική μας υλοποίηση η οποία έχει 2500 εγγραφές. Όλες οι νέες εγγραφές PUBLISHMENT αντιστοιχούν σε νέες τόσες εγγραφές και στο πεδίο ARTICLE.

## Θεωρητική απόδοση

Αρχικά, εξετάστηκε το παρακάτω query:

*SELECT id*

*FROM PUBLISHMENT*

*WHERE views = (SELECT MAX(views)*

*FROM PUBLISHMENT)*

το οποίο αναζητά τη δημοσίευση με τις περισσότερες αναγνώσεις στην βάση. Έγινε χρονομέτρηση με ευρετήριο union index στις στήλες “view” και “id” (με αυτή την σειρά) του πίνακα PUBLISHMENT. Αυτή η επιλογή έγινε καθώς η βάση αναζητά τον κωδικό id της δημοσίευσης ανάλογα με τις προβολές.

Εικ.3: Γράφημα χρόνου εύρεσης δημοσίευσης με τις περισσότερες προβολές με ευρετήρια (κόκκινο) και χωρίς ευρετήρια (μπλε). Ο χρόνος είναι σε δευτερόλεπτα και ο άξονας των x (αριθμός δημοσιεύσεων) είναι σε λογαριθμική κλίμακα.

Πράγματι, ενώ ο χρόνος αναζήτησης για μη-ύπαρξη ευρετηρίου αυξάνεται γραμμικά, ο χρόνος για το συγκεκριμένο ευρετήριο πρακτικά δεν αυξάνεται, καθώς απαιτείται μόνο η εύρεση της μεγαλύτερης τιμής των προβολών.

## Πρακτική απόδοση

Ωστόσο, η εφαρμογή μας δεν εκτελεί τη συγκεκριμένη εντολή στη βάση. Αντιθέτως, τα πιο συχνά queries που πραγματοποιεί, είναι queries ταξινόμησης δημοσιεύσεων. Έτσι, έγινε η μέτρηση του χρόνου εμφάνισης όλων των δημοσιεύσεων, του τίτλου και του αριθμού αναγνώσεων, ταξινομημένες κατά φθίνοντα αριθμό προβολών. Το query που εξετάστηκε ήταν το παρακάτω:

*SELECT id, COALESCE(title, journ\_title) AS title,views*

*FROM PUBLISHMENT*

*LEFT JOIN ARTICLE ON id = article\_id*

*LEFT JOIN ISSUE i ON id = i.issue\_id*

Έγιναν οι ίδιες χρονομετρήσεις με προηγουμένως, για ίδιο αριθμο δεδομένων, ωστόσο λόγω της σύνθετης μορφής του query στις μετρήσεις με ευρετήρια χρησιμοποιήσαμε περισσότερα από ένα.

Εικόνα που περιέχει γραμμή, γράφημα, διάγραμμα, πλαγιά

Περιγραφή που δημιουργήθηκε αυτόματαΣυγκεκριμένα χρησιμοποιήσαμε ευρετήρια πρωτεύοντων κλειδιών για όλους τους πίνακες που εμπλέκονται (ARTICLE, ISSUE και PUBLISHMENT) καθώς και ένα ευρετήριο για τη στήλη “title” των άρθρων, καθώς ο χρήστης μπορεί ανά πάσα στιγμή να εκτελέσει αναζήτηση με βάση τον τίτλο. Τέλος, δημιουργήθηκε ένα union ευρετήριο όπως στο προηγούμενο πείραμα, που περιέχει τις στήλες “views” και “article\_id” για ίδιους λόγους με πριν.

Εικ.4: Γράφημα χρόνου ταξινόμησης όλων των δημοσιεύσεων ανάλογα με τον αριθμό αναγνώσεων σε γραμμική και λογαριθμική κλίμακα.

Λόγω της σύνθετης μορφής του query, η διαφορά με τη χρήση ευρετηρίων δεν είναι όσο αποτελεσματική όσο προηγουμένως, καθώς και πάλι απαιτείται ανάγνωση όλων των πεδίων, απλά καθίσταται πιο γρήγορη η πράξη της ταξινόμησης. Η βελτίωση του χρόνου με χρήση των ευρετηρίων είναι της τάξης του 150%. Η διαφορά της απόδοσης είναι πρακτικά ίδια για τις περισσότερες πράξεις που εκτελεί η εφαρμογή μας.

Η αρχική ιδέα ήταν η δοκιμή να γίνει με την αναζήτηση μέσω keywords, ωστόσο αποδείχτηκε ότι ο χρόνος πρακτικά δεν άλλαζε με τη χρήση ευρετηρίων καθώς παρόλο που ο αριθμός των άρθρων (άρα και το πλήθος των τιμών στον πίνακα KEYWORDS) αυξανόταν, εφόσον τα ίδια τα keywords σαν ξεχωριστά string παρέμεναν στον ίδιο αριθμό (περίπου 150), ο χρόνος του sorting ήταν πρακτικά ίδιος καθώς ο αριθμός των άρθρων με τα ίδια keywords προφανώς ήταν αρκετά μικρός. Επομένως, εφόσον ο αριθμός των keywords δεν αναμένεται να ξεπεράσει ακόμη και κάποια χιλιάδες σε περιβάλλον ρεαλιστικής εφαρμογής, επιλέχθηκε να μην εισαχθεί ευρετήριο για τα keywords.