Εφαρμογή ζωολογικού κήπου

Διαχείριση βάσης δεδομένων σχετικά με την υγεία και την φροντίδα των ζώων

Ομάδα 47

Λυδία Εξάρχου ΑΜ: 1072721
Ιωάννης Κατής ΑΜ: 1072735

Περίληψη

Η βάση δεδομένων για έναν ζωολογικό κήπο διαχειρίζεται τις καταγραφές δεδομένων των ζώων, βοηθώντας το προσωπικό να παρακολουθεί τις εξελίξεις στην υγεία των ζώων και να κρατάει ιστορικό για την φροντίδα τους. Η βάση δεδομένων περιέχει πληροφορίες σχετικά με τη φροντίδα των κελιών των ζώων, την διατροφής τους, το ιστορικό ασθενείας τους και τις επισκέψεις σε κτηνίατρο. Η εφαρμογή επιτρέπει την αναζήτηση πληροφοριών με βάση οποιοδήποτε στοιχείο όπως για παράδειγμα το είδος ζώων, τα στοιχεία των υπαλλήλων ή την ημερομηνία μίας ενέργειας. Μπορεί επίσης να γίνει προσθήκη νέων ζώων στη βάση δεδομένων ή καταγραφή νέων πληροφοριών για την φροντίδα τους. Το σημαντικότερο κομμάτι στην δημιουργία της βάσης ήταν αυτή να περιοριστεί έτσι ώστε να αφορά την φροντίδα των ζώων, δηλαδή η εφαρμογή να χρησιμοποιείται αποκλειστικά εκ των έσω, ώστε να περιλαμβάνει όσο το δυνατόν πιο λεπτομερώς, ολοκληρωμένα και οργανωμένα τα δεδομένα. Το πιο ενδιαφέρον κομμάτι αυτής της ιδέας είναι η σύνδεση όλων των σύνθετων πληροφοριών μεταξύ τους, που θα δίνει στον χρήστη την ευελιξία εύκολα και γρήγορα να αναζητά τα παλιά δεδομένα και να καταγράφει τα νέα.

1 Μεθοδολογία

Η ανάπτυξη της εφαρμογής που διαχειρίζεται τα δεδομένα ενός ζωολογικού κήπου είναι σύνθετη διαδικασία, η οποία μπορεί να διαχωριστεί σε δύο βασικά μέρη. Σε πρώτη φάση, το κυρίαρχο ζήτημα είναι η σχεδίαση και η υλοποίησης της βάσης σε SQL κώδικα, που τρέχει πάντα πίσω από την εφαρμογή μας, και σε δεύτερη η δημιουργία μίας εφαρμογής σε γλώσσα python, η οποία αλληλοεπιδρά με την βάση που βρίσκεται στο παρασκήνιο, αλλά δημιουργεί ένα ευχάριστο και εύχρηστο περιβάλλον για τον χρήστη. Θα γίνει ξεχωριστά ανάλυση της μεθοδολογίας κάθε μέρους.

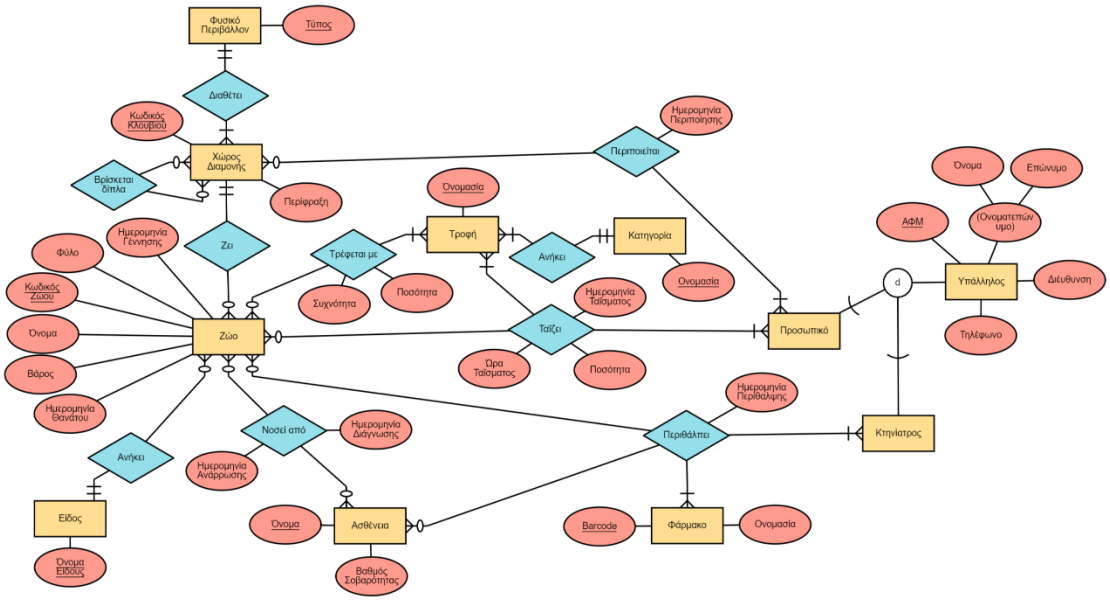
1.1 Βάση δεδομένων

Η δημιουργία της βάσης δεδομένων απαίτησε αρκετή μελέτη και χρόνο, καθώς και συγκεκριμένα στάδια έως την ολοκλήρωσή της. Πρώτα έπρεπε να γίνει η θεωρητική σχεδίαση άρα η περιγραφή του μικρόκοσμου, έπειτα η πραγματική υλοποίηση SQL με ιδιαίτερη προσοχή στην αναφορική ακεραιότητα και σε άλλες λεπτομέρειες και τέλος η προσθήκη τυχαίων δεδομένων στην βάση με σκοπό να αναδεικνύεται η υποθετική χρήση της.

1.1.1 Σχεδίαση μικρόκοσμου και εννοιολογικό μοντέλο (ERD)

Αρχικά έπρεπε να γίνει μελέτη ως προς το τι είναι απαραίτητο να περιέχει μία βάση δεδομένων ενός ζωολογικού κήπου, δηλαδή εξοικείωση με το θέμα, τις ανάγκες και τις ιδιαιτερότητες του ζητήματος. Η πρώτη απόφαση που πάρθηκε ήταν ο προσανατολισμός της βάσης στις ανάγκες των ζώων και των εργαζομένων, δηλαδή η βάση έχει το προφίλ ενός λεπτομερειακού ψηφιακού ιστορικού καταγραφής των γεγονότων και των πληροφοριών του κήπου. Για τον ζωολογικό κήπο είναι σημαντικό να διατηρεί αρχείο με τα ζώα που διαθέτει και τις ανάγκες τους καθώς και το σημείο όπου βρίσκεται το καθένα. Πιο συγκεκριμένα οφείλει να γνωρίζει τι τροφές καταναλώνει κάθε ζώο καθώς και το ιστορικό των γευμάτων του. Ακόμη είναι χρήσιμο να καταγράφει λεπτομερώς τον χώρο που ζει κάθε είδος ζώου, δηλαδή απαιτούνται βασικά χαρακτηριστικά κάθε χώρου (θέση, είδος κλουβιού κ.ά.), το ιστορικό φροντίδας του από το προσωπικό καθώς και με ποια άλλα ζώα βρίσκεται κοντά. Ο ζωολογικός κήπος χρειάζεται επιπλέον πλήρη λίστα με τους εργαζόμενους του. Τέλος, στον ζωολογικό κήπο υπάρχουν και κτηνίατροι οι οποίοι είναι υπεύθυνοι για την περίθαλψη των ζώων, την διατήρηση ιστορικού τους σε περίπτωση ασθένειας όπως επίσης και για την χορήγηση φαρμάκων.

Για τον μικρόκοσμο αυτό κατασκευάσαμε το παρακάτω εννοιολογικό μοντέλο με χρήση κατάλληλης εφαρμογής [1].



Εικόνα 1: ERD βάσης δεδομένων

Αποφασίσαμε να δημιουργήσουμε τις παρακάτω 11 οντότητες:

- Ζώο, η βασικότερη οντότητα της βάσης, αφορά ξεχωριστά το κάθε ζώο που ζει στον κήπο
- Είδος (Ζώου), σχετίζεται συνολικότερα με το κάθε είδος ζώου
- Χώρος διαμονής, το κλουβί όπου διαμένουν ένα ή περισσότερα είδη του κήπου
- Φυσικό περιβάλλον, περιγράφει το περιβάλλον που προσομοιάζει το κλουβί
- Τροφή, κάθε είδος βρώσιμής ουσίας που καταναλώνεται από τα ζώα
- Κατηγορία (Τροφής), προσδίδει στην τροφή την κατηγορία στην οποία ανήκει
- Ασθένεια, κάθε είδος νοσήματος από το οποίο μπορεί να πάσχει ένα ζώο του κήπου
- Φάρμακο, κάθε φάρμακο που δίνεται στα ζώα από τους κτηνιάτρους
- Υπάλληλος, όλοι οι εργαζόμενοι του κήπου
- Προσωπικό, οι υπάλληλοι που φροντίζουν τα κλουβιά ή ταΐζουν τα ζώα
- Κτηνίατροι, οι υπάλληλοι στον τομέα υγείας των ζώων

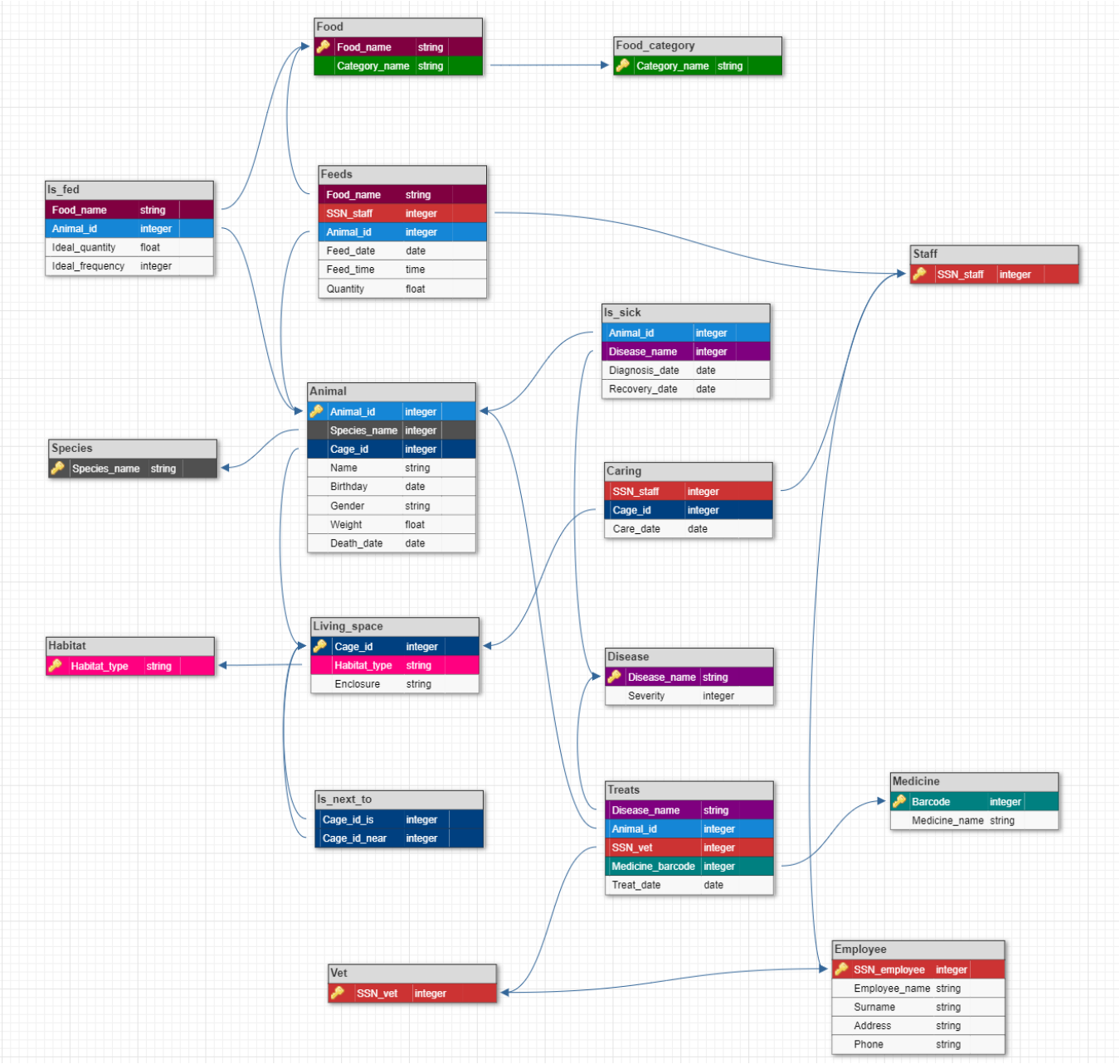
Παρακάτω περιγράφουμε κάποια από τα ιδιαίτερα στοιχεία που συναντά κανείς στο συγκεκριμένο ERD:

- Η αυτοσυσχέτιση «Βρίσκεται δίπλα» στην οντότητα Χώρος Διαμονής μας δίνει την σχετική θέση των κλουβιών στον κήπο, δηλαδή ως προς τα άλλα κλουβιά.
- Η σχέση «Τρέφεται με» λειτουργεί ως πίνακας καταγραφής όλων των τροφών τα οποία δύναται να καταναλώσει κάθε ζώο.
- Η σχέση «Ταΐζει» λειτουργεί ως ιστορικό καταγραφής της καθημερινής διατροφής των ζώων, λαμβάνοντας υπ' όψιν και τον υπάλληλο που τάισε το ζώο.
- Η σχέση «Νοσεί από» λειτουργεί ως ιστορικό καταγραφής νόσησης του κάθε ζώου, με ημερομηνία νόσησης και ημερομηνία ανάρρωσης.
- Η σχέση «Περιθάλπει» λειτουργεί ως ιστορικό όλων των περιστατικών που εξετάζει ο κτηνίατρος, είτε αυτά είναι σχετικά με κάποια ασθένεια είτε χρειάζονται χορήγηση φαρμάκου.
- Η σχέση «Περιποιείται» λειτουργεί ως ιστορικό του καθαρισμού και της περιποίησης των κελιών από τους υπαλλήλους.

Με βάση το συγκεκριμένο διάγραμμα, προχωρήσαμε στην κατασκευή του λογικού σχεσιακού μοντέλου

1.1.2 Δημιουργία λογικού σχεσιακού μοντέλου

Με βάση τους κανόνες μετατροπής του ERD σε λογικό μοντέλο, σχεδιάσαμε το παρακάτω διάγραμμα με χρήση της εφαρμογής [2].



Εικόνα 2: Λογικό σχεσιακό μοντέλο

Κατασκευάσαμε τους πίνακες των 11 οντοτήτων και έπειτα 6 ακόμη πίνακες που αφορούν την καταγραφή ασθενειών, την περίθαλψη από κτηνίατρο, την αυτοσυσχέτιση “βρίσκεται δίπλα”, την καταγραφή ταΐσματος, τα στοιχεία διατροφής και την περιποίηση κελιού. Αυτές οι σχέσεις προστέθηκαν στο διάγραμμα ως πίνακες γιατί είναι σχέσεις 1 ή πολλά προς πολλά, όπως φαίνεται αναλυτικά στο erd, άρα χρειάζονται έναν παραπάνω πίνακα για να δοθούν ολοκληρωμένα. Προσθέτουμε όλα τα απαραίτητα foreign keys, περνώντας έτσι τις συσχετίσεις του ERD στο διάγραμμα.

1.1.3SQL

Μέσω της εφαρμογής που δημιουργήσαμε το λογικό σχεσιακό μοντέλο, μας δόθηκε η δυνατότητα να κάνουμε εξαγωγή τον αντίστοιχο κώδικα σε SQLite και δουλέψαμε πάνω σε αυτόν. Ο κώδικας αυτός περιείχε τις εντολές CREATE TABLE για όλους τους πίνακες που είχαμε δημιουργήσει, με τις αντίστοιχες ιδιότητες και τους τύπους τους. Για παράδειγμα για το Table animal πήραμε τον παραπάνω κώδικα:

SQLite 1: Animal table

```
CREATE TABLE IF NOT EXISTS "Animal" (
	"Animal_id"	integer,
	"Species_name"	integer,
	"Cage_id"	integer,
	"Name"	string,
	"Birthday"	date,
	"Gender" string,
	"Weight" float,
	"Death_date"	date
);
```

Σε αυτόν τον κώδικα προσθέσαμε τις εντολές που ορίζουν τα κύρια κλειδιά, τα ξένα κλειδιά και που αναφέρονται, αλλά και τους προσδιορισμούς αναφορικής ακεραιότητας. Στην συγκεκριμένη περίπτωση ο πίνακας Animal έχει ως ξένο κλειδί το είδος ζώου, το οποίο δεν διαγράφεται ποτέ από την βάση ούτε ενημερώνεται, άρα δεν υπήρχε λόγος να προσδιοριστεί η συμπεριφορά της βάσης σε μία τέτοια περίπτωση. Ακόμη, έχει ως ξένο κλειδί τον αριθμό κλουβιού που διαμένει το ζώο, το οποίο σε περίπτωση διαγραφής (δηλαδή κατάργησης του προκείμενου κλουβιού) γίνεται NULL και σε περίπτωση ενημέρωσης του αριθμού του προκείμενου κλουβιού γίνεται CASCADE, άρα αυτόματα αλλάζει και ο αριθμός στο animal. Τελικά ο κώδικας παίρνει αυτή την μορφή:

SQLite 2: Animal table

```
CREATE TABLE IF NOT EXISTS "Animal" (
	"Animal_id"	integer,
	"Species_name"	integer,
	"Cage_id"	integer,
	"Name"	string,
	"Birthday"	date,
	"Gender" string,
	"Weight" float,
	"Death_date"	date,
	PRIMARY KEY("Animal_id"),
	FOREIGN KEY("Species_name") REFERENCES "Species"("Species_name"),
	FOREIGN KEY("Cage_id") REFERENCES "Living_space"("Cage_id") ON DELETE SET NULL ON UPDATE CASCADE
    );
```

Αυτές οι προσθήκες έγιναν και για τα 17 tables που έχουμε δημιουργήσει στην SQLite, συνδέοντας έτσι όλα τα tables μεταξύ τους, όπως είχαμε ορίσει στα αρχικά διαγράμματα. Με την ίδια λογική ορίσαμε και την συμπεριφορά της βάσης σε διάφορες περιπτώσεις διαγραφών και ανανεώσεων, γνωρίζοντας πάντα ότι σε κάποιους πίνακες δεν θα υπάρχει δυνατότητα διαγραφής των δεδομένων, όπως στον πίνακα που περιέχει όλες τις τροφές ή στον πίνακα που περιέχει όλα τα φάρμακα.

1.1.4 Δεδομένα βάσης

Το επόμενο στάδιο στην δημιουργία της βάσης ήταν να δώσουμε κάποια ενδεικτικά δεδομένα που θα γεμίσουν την βάση, ώστε να προχωρήσουμε στην δημιουργία ενδεικτικών ερωτήσεων που θα απαντάει η βάση μας. Για την δημιουργία των δεδομένων χρησιμοποιήσαμε random generator [3], το οποίο μας έδωσε σε αρχείο csv τους πίνακες που χρειαζόμασταν για την βάση. Για όσα δεδομένα που απαιτούσε η βάση τα generator αυτό δεν μας κάλυπτε, χρησιμοποιώντας κώδικα python, γράψαμε σύντομα script για να μπορέσουμε γρήγορα να δημιουργήσουμε τον μεγάλο αριθμό δεδομένων που χρειαζόμασταν. Η προσθήκη των δεδομένων στην βάση έγινε μέσω εφαρμογής [4], που χρησιμοποιήσαμε γενικά για την διαχείριση της βάσης όσο ήταν σε μορφή SQL, από την οποία όταν κάναμε export τον κώδικά SQLite, έκανε autogenerate τις αντίστοιχες εντολές. Για παράδειγμα στην προσθήκη μίας νέας τροφής με χαρακτηριστικό το όνομα και το είδος της, ο τελικός κώδικας περιείχε την εντολή:

```
INSERT INTO "Food" VALUES ('tomato','fruit');
```

1.1.5 Ενδεικτικές τυπικές αναζητήσεις (queries)

Για να επιδείξουμε την ορθή λειτουργία της βάσης καταγράψαμε κάποια τυπικά και κάποια πιο σύνθετα ερωτήματα τα οποία μπορεί να δίνει ο χρήστης στην βάση και να λαμβάνει τα σωστά δεδομένα. Παρακάτω αναφέρονται ένα-ένα τα διάφορα ερωτήματα που αφορούν την βάση μας.

1. Ποιο κλουβί έχει τα περισσότερα διπλανά κλουβιά;

Query 1: SQLite κώδικας

```
select Cage_id_is, count(Cage_id_near) as maxim
FROM Is_next_to
group by Cage_id_is
having maxim= (SELECT MAX(next)
        FROM (  SELECT Cage_id_is, count(cage_id_near) as next
            FROM Is_next_to
            GROUP by Cage_id_is
))
```

2. Τι είδη υπάρχουν στον κήπο και πόσα ζώα υπάρχουν από το κάθε ένα;

Query 2: SQLite κώδικας

```
SELECT Species_name, COUNT(Species_name)
FROM Animal
GROUP BY Species_name
```

3. Ποιος είναι ο μέσος όρος ηλικίας των ζωντανών ζώων;

Query 3: SQLite κώδικας

```
Select AVG (date("now")-Birthday)
from Animal
where Death_date is NULL
```

4. Ποια ζώα που έχουν πεθάνει είχαν αρρωστήσει πάνω από 2 φορές(εκείνο τον χρόνο);

Query 4: SQLite κώδικας

```
Select Animal_id, Species_name,
from animal natural join Is_sick
where Death_date is not Null and strftime("%Y", Death_date)=strftime("%Y", Diagnosis_date)
group by Animal_id
HAVING count (Animal_id)>1
```

5. Ποιοι υπάλληλοι ταΐζουν τα ζώα περισσότερο από το επιθυμητό;

Query 5: SQLite κώδικας

```
Select Employee_name, Surname, SSN_staff
from Animal NATURAL JOIN Feeds NATURAL join Is_fed join Employee on SSN_staff=SSN_employee
where Ideal_quantity<Quantity
group by SSN_staff
```

6. Πόσο τρώνε κατά μέσο όρο ανά ημέρα τα ζώα ανά είδος;

Query 6: SQLite κώδικας

```
Select avg (Quan), Species_name
    from (SELECT Species_name,avg(Quantity) as Quan, *
```

```
        FROM Feeds NATURAL JOIN Animal
    GROUP by Feed_date
        ORDER by Species_name)
    group by Species_name
```

---

7. Ποιο είδος ζώου μένει σε κλουβί χωρίς να συμβιώνει με άλλο είδος;

Query 7: SQLite κώδικας

```
select Species_name
from (Select *
        FROM Animal
        GROUP by Species_name)
GROUP by Cage_id
HAVING count (Cage_id) = 1
```

---

8. Ανάμεσα στα μακροβιότερα ζώα του κήπου, ποιες τροφές είναι οι πιο συνηθισμένες;

Query 8: SQLite κώδικας

```
select Food_name, count(Food_name) as Fr
from Is_fed
where Animal_id in
                    (select Animal_id
                    from (Select Animal_id, date("now")-Birthday as age
                            from Animal
                            where Death_date is NULL
                            UNION
                            select Animal_id, Death_date-Birthday
                            from Animal
                            where Death_date is not NULL)
                    where age =(select max(age) as old
                                from (Select date("now")-Birthday as age
                                        from Animal
                                        where Death_date is NULL
                                        UNION
                                        select Death_date-Birthday
                                        from Animal
                                        where Death_date is not NULL
                                        order by age desc)))
group by Food_name
having Fr>1
order by Fr Desc
```

1.2 Εφαρμογή

Σε επόμενη φάση, αφού είχαμε ολοκληρώσει την SQL, έπρεπε να δημιουργήσουμε την εφαρμογή μας χρησιμοποιώντας κώδικα python. Αποφασίσαμε αρχικά τις βασικές λειτουργίες της εφαρμογής μας. Επιλέξαμε να έχει ο χρήστης της ευελιξία είτε να βλέπει και να αναζητά στα υπάρχοντα δεδομένα, είτε να προσθέτει νέα δεδομένα. Για αυτό τον σκοπό βάλαμε δύο κουμπιά στο παράθυρο έναρξης της εφαρμογής, με τις δύο αυτές λειτουργίες. Ακόμη, προσθέσαμε ένα κουμπί όπου ο χρήστης έχει την δυνατότητα να εκτελέσει κάποια από τα προαναφερθέντα queries και να λάβει αποτελέσματα. Όλα τα κουμπιά αποτελούν εικονίδια που δημιουργήθηκαν με χρήση ειδικής πλατφόρμας[5].

1.2.1 Υπάρχουσες καταχωρήσεις και αναζήτηση

Με την επιλογή του κουμπιού εμφάνισης των υπαρχόντων δεδομένων ανοίγει ένα νέο παράθυρο που δίνει την δυνατότητα στον χρήστη να επιλέξει μια από τις κατηγορίες δεδομένων και να εμφανίσει τα περιεχόμενά της που υπάρχουν εκείνη την στιγμή στην βάση. Με την επιλογή μιας από τις κατηγορίες, που κάθε μια αποτελεί και ένα

ξεχωριστό κουμπί, ανοίγει ένα νέο αναδυόμενο παράθυρο το οποίο εμφανίζει τα δεδομένα της επιλεγμένης κατηγορίας και δίνει ακόμη την δυνατότητα αναζήτησης με βάση συγκεκριμένο γνώρισμα και όνομα, ομαδοποίησης ανάλογα των επιθυμιών του χρήστη και ταξινόμησης τους με φθίνουσα σειρά.

1.2.2 Προσθήκη δεδομένων

Με την επιλογή του κουμπιού προσθήκης δεδομένων, ο χρήστης αντιμετωπίζει ένα νέο παράθυρο που του δίνει την δυνατότητα να επιλέξει το είδος της προσθήκης που θέλει να κάνει, ουσιαστικά σε ποιον πίνακα της βάσης θα γίνει η προσθήκη του δεδομένου. Κάθε είδος προσθήκης αποτελεί ένα κουμπί, το οποίο εάν πατηθεί καλεί την αντίστοιχη συνάρτηση, που εμφανίζει ένα ακόμη αναδυόμενο παράθυρο, με την κατάλληλη φόρμα συμπλήρωσης. Ο χρήστης καλείται να συμπληρώσει την φόρμα με τα στοιχεία και να την υποβάλλει. Η υποβολή καλεί ένα παραμετροποιημένο query τύπου insert ή update με ορίσματα τα στοιχεία που έδωσε ο χρήστης, που εκτελείται στην βάση και το δεδομένο προστίθεται. Ο χρήστης κατά την εισαγωγή των στοιχείων μπορεί να κάνει 3 ειδών λάθη, τα οποία αναγνωρίζονται από το σύστημα και διακόπτουν την διαδικασία ενημερώνοντας τον χρήστη. Αυτά είναι:

- Κάποιο πεδίο να είναι κενό
- Κάποιο πεδίο να μην είναι στην σωστή μορφή (πχ αλφαριθμητικό σε πεδίο εισαγωγής ημερομηνίας)
- Κάποιο πεδίο να μην ικανοποιεί τις συνθήκες ξένων κλειδιών της βάσης

1.2.3 Queries

Με την επιλογή του κουμπιού των Queries ανοίγει ένα νέο παράθυρο που εμφανίζει μερικά από τα προαναφερθέντα queries και δίνει την δυνατότητα στον χρήστη επιλογής τους. Έπειτα με το κουμπί της επιλογής εμφανίζεται ένα νέο αναδυόμενο παράθυρο με τα αποτελέσματα του query. Στην περίπτωση επιλογής του κουμπιού ακύρωσης το παράθυρο κλείνει και ο χρήστης συνεχίζει την πλοήγηση του στην εφαρμογή.
Στο σημείο αυτό αξίζει να σημειωθεί ότι στον κώδικα της εφαρμογής και συγκεκριμένα στο κομμάτι των queries έχει υλοποιηθεί η δημιουργία ευρετηρίου αλλά και η χρονομέτρηση εκτέλεσης της εντολής με και χωρίς την χρήση ευρετηρίου. Τα συμπεράσματα της σύγκρισης αυτής βρίσκονται στο παράρτημα.

2 Αξιολόγηση εφαρμογής

Ως κριτήρια επιτυχίας του project-εφαρμογή μας χρησιμοποιήσαμε τα εξής:

- Περιεκτική περιγραφή του προβλήματος
- Σωστή εφαρμογή θεωρίας βάσεων δεδομένων-Κανονικοποίηση της βάσης
- Εύκολη χρήση εφαρμογής-Χρήση γραφικής διεπαφής
- Πλήρης και αδιάκοπη λειτουργία-Διαχείριση σφαλμάτων
- Ικανοποιητικό δείγμα τυχαίων δεδομένων

3 Ολοκλήρωση του project

3.1 Διαχωρισμός ενεργειών

Η υλοποίηση του Project έγινε με πλήρη συνεργασία και των δύο μελών της ομάδας. Αρχικά συζητήσαμε το πρόβλημα που μας ανατέθηκε και καταλήξαμε σε μια πιθανή λύση του. Στην συνέχεια δημιουργήσαμε από κοινού το ERD και το λογικό σχεσιακό μοντέλο και ξεκινήσαμε να προσθέτουμε μαζί καταχωρήσεις στην βάση μας. Ύστερα η υλοποίηση του κώδικα έγινε με την κοινή χρήση κώδικα μέσω του Visual Studio Code και χωρίστηκε κατά βάση ως εξής :

1. Τις λειτουργίες εμφάνισης των υπαρχόντων δεδομένων ανέλαβε ο Κατής Ιωάννης
2. Τις λειτουργίες για την προσθήκη νέων δεδομένων τις ανέλαβε η Εξάρχου Λυδία

Σημειώνεται ότι στον διαχωρισμό αυτόν, από κοινού έγιναν η υλοποίηση των συναρτήσεων error handling και της αναζήτησης στα δεδομένα. Αντίστοιχα και τα queries έγιναν από κοινού.

3.2 Χρονοδιάγραμμα ενεργειών

Χρονοδιάγραμμα Υλοποίησης	
4η-6η εβδομάδα	Δημιουργία μικρόκοσμου, ERD και προετοιμασία ενδιάμεσης παρουσίασης
7η εβδομάδα	Λογικό σχεσιακό μοντέλο
8η-9η εβδομάδα	Επεξεργασία SQL και φόρτωση δεδομένων στην βάση
10η-12η εβδομάδα	Υλοποίηση κώδικα python

Χρονοδιάγραμμα Υλοποίησης	
13η εβδομάδα	Υλοποίηση αναφοράς, δημιουργία παρουσίασης και παράδοση project

Βιβλιογραφία

[1] https://erdmaker.com/
[2] https://www.dbdesigner.net/
[3] https://www.mockaroo.com/
[4] https://sqlitebrowser.org/dl/
[5] https://www.canva.com/

A Παράρτημα

A.1 Οδηγίες εγκατάστασης

Απαραίτητες προϋποθέσεις για την επιτυχή εκτέλεσης της εφαρμογής είναι οι εξής:
1. Εγκατάσταση της βιβλιοθήκης PySimpleGui
2. Ο κώδικας, το db αρχείο και ο φάκελος «icons» να βρίσκονται στον ίδιο φάκελο
Η βιβλιοθήκη PySimpleGui εγκαθίστανται με την εντολή «pip install pysimplegui» στο command line με path τον φάκελο εγκατάστασης της python.

A.2 Παραδείγματα χρήσης της εφαρμογής

Παρακάτω θα γίνει επίδειξη για δύο τυπικές χρήσης της εφαρμογής. Σε κάθε περίπτωση η επεξήγηση ξεκινάει από την παρακάτω οθόνη έναρξης:

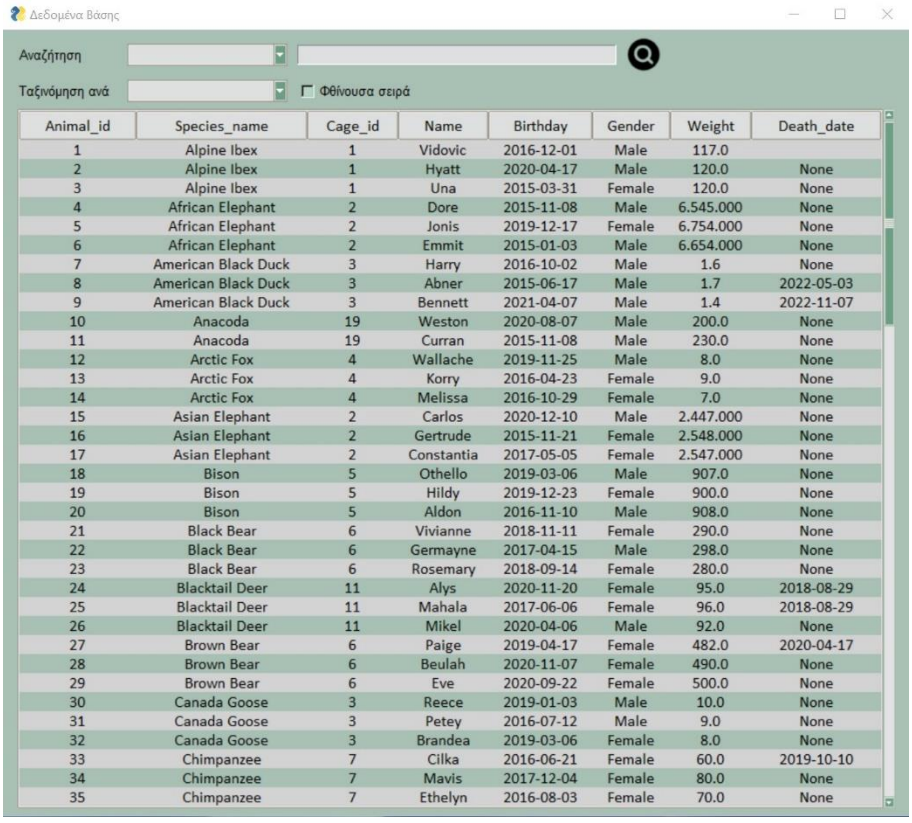
Εικόνα 3: Οθόνη έναρξης

A.2.1 Προβολή υπαρχόντων καταχωρήσεων

Έστω ότι θέλουμε να εμφανίσουμε τα υπάρχοντα δεδομένα της βάσης του ζωολογικού κήπου και πιο συγκεκριμένα τα ζώα που υπάρχουν σε αυτόν. Επιλέγουμε αρχικά το κουμπί «Υπάρχουσες καταχωρήσεις» στην οθόνη έναρξης της εφαρμογής. Από το νέο παράθυρο επιλέγουμε το κουμπί «Δεδομένα Ζώων Κήπου».

Εικόνα 4: Επιλογή κατηγορίας δεδομένων

Στην συνέχεια μπορούμε να δούμε τα υπάρχοντα δεδομένα της βάσης εκείνη την στιγμή, να πραγματοποιήσουμε μια αναζήτηση στην συγκεκριμένη κατηγορία αλλά και να ομαδοποιήσουμε και να ταξινομήσουμε σε φθίνουσα σειρά τα αποτελέσματα. Για την επανεμφάνιση των αρχικών δεδομένων μετά από μία αναζήτηση απλά ξαναπατάμε το κουμπί της αναζήτησης ή enter.



Animal_id	Species_name	Cage_id	Name	Birthday	Gender	Weight	Death_date
1	Alpine Ibex	1	Vidovic	2016-12-01	Male	117.0	
2	Alpine Ibex	1	Hyatt	2020-04-17	Male	120.0	None
3	Alpine Ibex	1	Una	2015-03-31	Female	120.0	None
4	African Elephant	2	Dore	2015-11-08	Male	6.545.000	None
5	African Elephant	2	Jonis	2019-12-17	Female	6.754.000	None
6	African Elephant	2	Emmit	2015-01-03	Male	6.654.000	None
7	American Black Duck	3	Harry	2016-10-02	Male	1.6	None
8	American Black Duck	3	Abner	2015-06-17	Male	1.7	2022-05-03
9	American Black Duck	3	Bennett	2021-04-07	Male	1.4	2022-11-07
10	Anacoda	19	Weston	2020-08-07	Male	200.0	None
11	Anacoda	19	Curran	2015-11-08	Male	230.0	None
12	Arctic Fox	4	Wallache	2019-11-25	Male	8.0	None
13	Arctic Fox	4	Korry	2016-04-23	Female	9.0	None
14	Arctic Fox	4	Melissa	2016-10-29	Female	7.0	None
15	Asian Elephant	2	Carlos	2020-12-10	Male	2.447.000	None
16	Asian Elephant	2	Gertrude	2015-11-21	Female	2.548.000	None
17	Asian Elephant	2	Constantia	2017-05-05	Female	2.547.000	None
18	Bison	5	Othello	2019-03-06	Male	907.0	None
19	Bison	5	Hildy	2019-12-23	Female	900.0	None
20	Bison	5	Aldon	2016-11-10	Male	908.0	None
21	Black Bear	6	Vivianne	2018-11-11	Female	290.0	None
22	Black Bear	6	Germayne	2017-04-15	Male	298.0	None
23	Black Bear	6	Rosemary	2018-09-14	Female	280.0	None
24	Blacktail Deer	11	Alys	2020-11-20	Female	95.0	2018-08-29
25	Blacktail Deer	11	Mahala	2017-06-06	Female	96.0	2018-08-29
26	Blacktail Deer	11	Mikel	2020-04-06	Male	92.0	None
27	Brown Bear	6	Paige	2019-04-17	Female	482.0	2020-04-17
28	Brown Bear	6	Beulah	2020-11-07	Female	490.0	None
29	Brown Bear	6	Eve	2020-09-22	Female	500.0	None
30	Canada Goose	3	Reece	2019-01-03	Male	10.0	None
31	Canada Goose	3	Petey	2016-07-12	Male	9.0	None
32	Canada Goose	3	Brandea	2019-03-06	Female	8.0	None
33	Chimpanzee	7	Cilka	2016-06-21	Female	60.0	2019-10-10
34	Chimpanzee	7	Mavis	2017-12-04	Female	80.0	None
35	Chimpanzee	7	Ethelyn	2016-08-03	Female	70.0	None

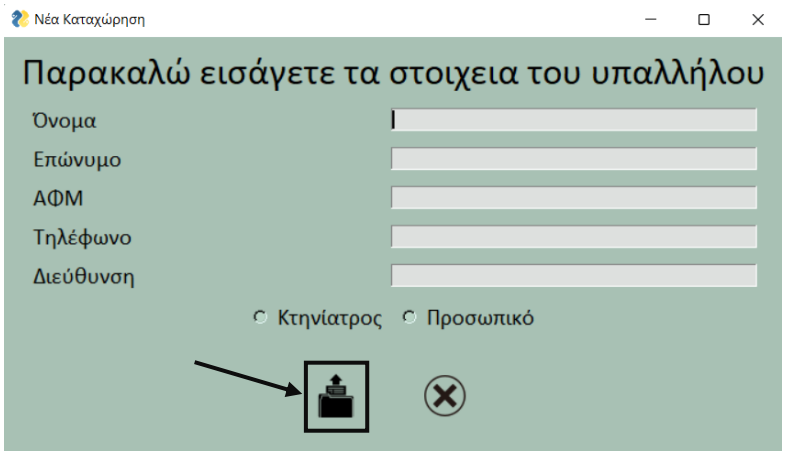
Εικόνα 5: Εμφάνιση Δεδομένων

Α.2.2 Εισαγωγή νέας καταχώρησης

Έστω ότι θέλουμε να κάνουμε μία νέα καταχώριση στην βάση του ζωολογικού κήπου, στην προκείμενη περίπτωση θέλουμε να προσθέσουμε ένα νέο υπάλληλο. Αρχικά επιλέγουμε το κουμπί νέα καταχώρηση στην οθόνη έναρξης της εφαρμογής. Από το αναδυόμενο παράθυρο επιλέγουμε το κουμπί προσθήκη υπαλλήλου.

Εικόνα 6: Επιλογή επιθυμητής προσθήκης

Στην συνέχεια συμπληρώνουμε τα ζητούμενα στοιχεία στο παράθυρο που προκύπτει, σημειώνοντας την κατάλληλη επιλογή, εάν ο νέος υπάλληλος είναι κτηνίατρος ή προσωπικό, και καταχωρούμε τα στοιχεία πατώντας το αντίστοιχο εικονίδιο.

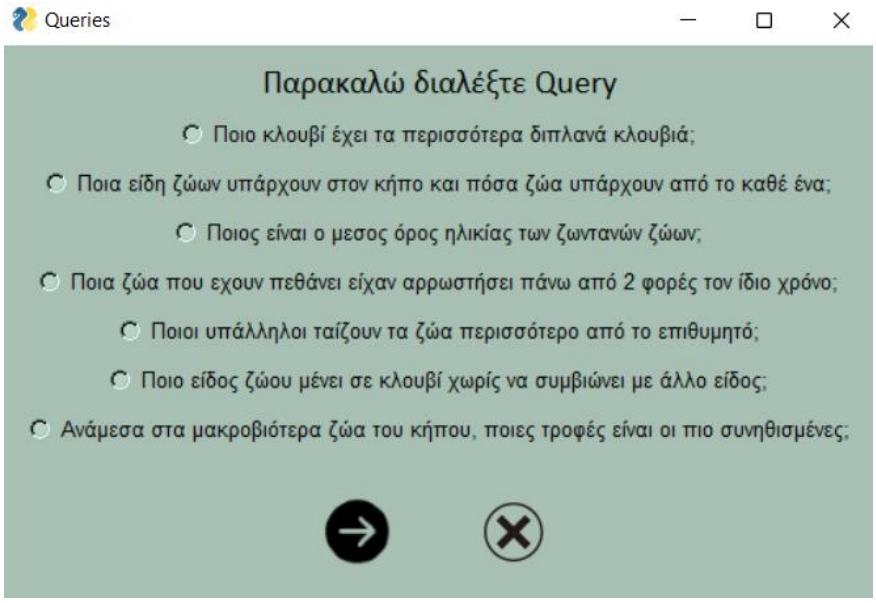


Εικόνα 7: Καταχώρηση στοιχείων

Εάν γίνει κάποιο λάθος στα στοιχεία που εισάγονται και η προσθήκη στην βάση δεν μπορεί να πραγματοποιηθεί, θα υπάρξει ανάλογο μήνυμα σε αναδυόμενο παράθυρο, ενημερώνοντας τον χρήστη.

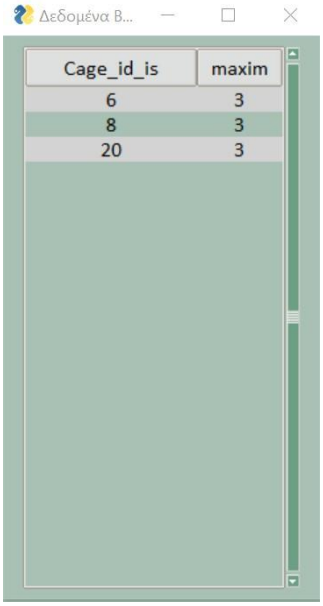
Α.2.3 Queries

Έστω ότι θέλουμε να τρέξουμε κάποιο query, τότε επιλέγουμε το αντίστοιχο κουμπί από την οθόνη έναρξης και στην συνέχεια το query που επιθυμούμε να τρέξουμε ώστε να εμφανιστούν τα αντίστοιχα αποτελέσματα.



Εικόνα 8: Επιλογή Query

Αν για παράδειγμα διαλέξουμε το πρώτο query και μετά το πάτημα του κουμπιού επιλογής, μας εμφανίζονται τα αποτελέσματα της ερώτησης σε ένα νέο αναδυόμενο παράθυρο όπως φαίνεται στην εικόνα 9 παρακάτω.



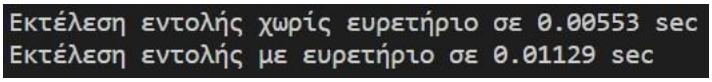
Εικόνα 9: Αποτελέσματα 1ου Query

Παρακάτω φαίνεται και η χρονική σύγκριση του ίδιου query χωρίς και με την χρήση ευρετηρίου (index).

```
Εκτέλεση εντολής χωρίς ευρετήριο σε 0.00233 sec
Εκτέλεση εντολής με ευρετήριο σε 0.00059 sec
```

Εικόνα 10: Χρονική σύγκριση με χρήση index

Αυτό που παρατηρούμε είναι ότι ενώ θεωρητικά θα περιμέναμε πολύ μεγαλύτερη χρονική διαφορά στην εκτέλεση της εντολής του query, οι χρόνοι είναι αρκετά κοντά στο συγκεκριμένο παράδειγμα. Αυτό πιθανώς να οφείλεται στον μικρό αριθμό δεδομένων που έχει η βάση και πιο συγκεκριμένα το table στο οποίο ψάχνει το query. Με εκτέλεση του 5ου query για παράδειγμα που περιλαμβάνει τον μεγαλύτερο σε μέγεθος πίνακα της βάσης μας (5000 δεδομένα) παίρνουμε τους χρόνους που φαίνονται παρακάτω στην εικόνα 11.



Εικόνα 11: Χρονική Σύγκριση 5ου Query

Παρατηρούμε ότι ο χρόνος εκτέλεσης με ευρετήριο είναι μεγαλύτερος από αυτόν χωρίς. Καταλήγουμε επομένως στο συμπέρασμα ότι η χρήση ευρετηρίου χρησιμεύει περισσότερο στην αναζήτηση μεγάλου όγκου δεδομένων, πιθανώς μεγαλύτερο από 10000.