Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Βάσεις Δεδομένων Εξαμηνιαία Εργασία

Ακ. έτος 2021-2022

Τίτλος:	**Εξαμηνιαία Εργασία**
Μάθημα:	**Βάσεις Δεδομένων**
Εξάμηνο:	**6ο**
Διδάσκοντες:	**Δημήτριος Τσουμάκος,Βασιλική Καντερέ, Μάριος Κόνιαρης**
Ομάδα:	**93**
Ο/επώνυμα:	**Σαμπάνης Αθανάσιος – ΑΜ: 03119054** **Φρουδάκης Ευάγγελος – ΑΜ: 03119108** **Χατζής Ιωάννης Κωνσταντίνος – ΑΜ: 03119923**

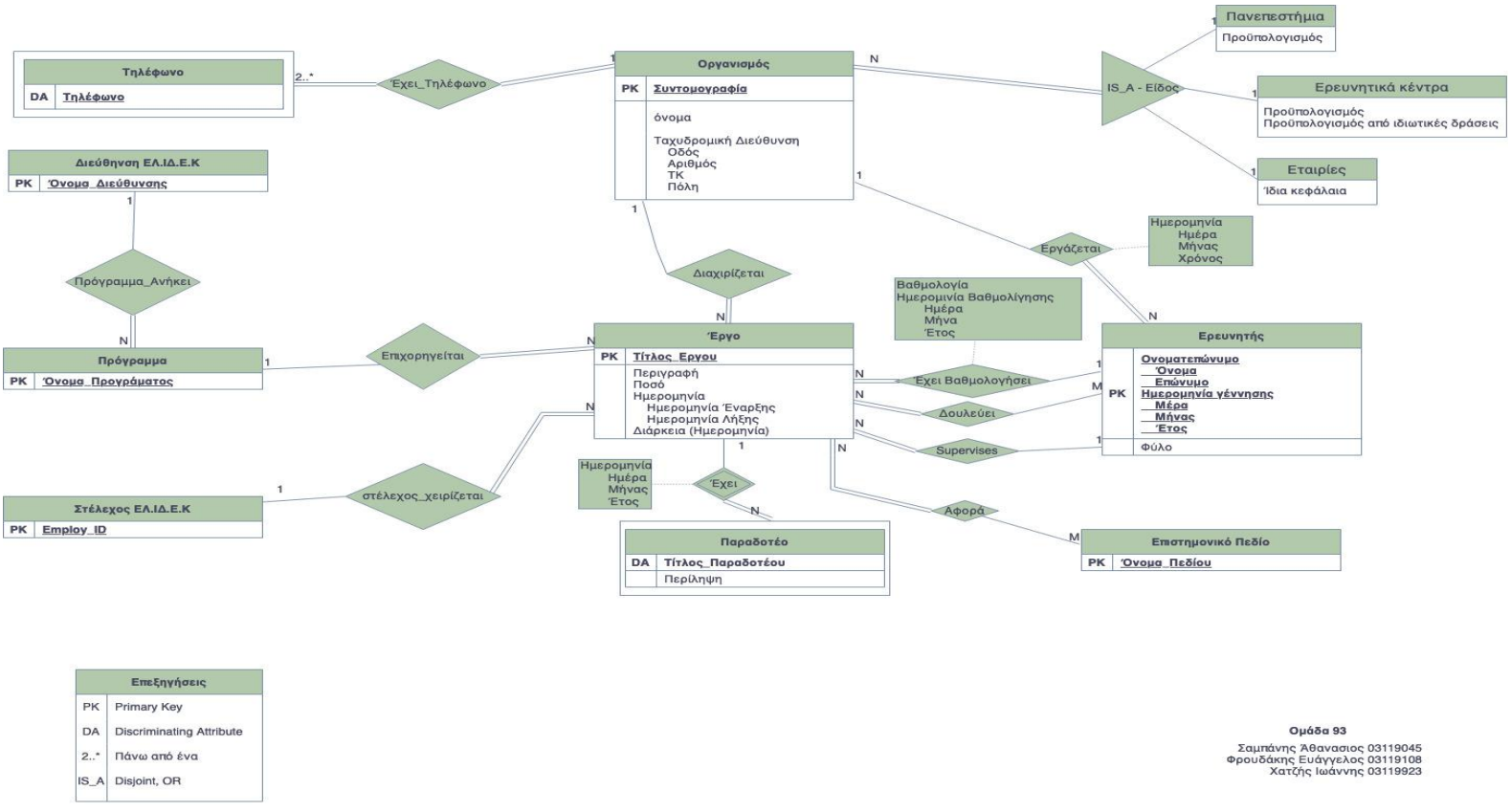
Ημερομηνία Παράδοσης: 05/06/2022

Εισαγωγή

Στην εργασία αυτή ζητούμενο ήταν η δημιουργία βάσης δεδομένων για το Ελληνικό Ίδρυμα Έρευνας και Καινοτομίας - ΕΛ.ΙΔ.Ε.Κ. το οποίο είναι σημαντικός χρηματοδότης της ακαδημαϊκής έρευνας στην Ελλάδα. Σε αυτό συμμετέχουν οργανισμοί (Πανεπιστήμια, Ερευνητικά Κέντρα ή Εταιρείες) οι οποίοι αναλαμβάνουν την διεκπεραίωση έργων. Αυτά τα έργα καταγράφονται στην βάση δεδομένων του ΕΛ.ΙΔ.Ε.Κ και βάσει της αξιολόγησης που λαμβάνουν, χρηματοδοτούνται ανάλογα. Πέραν της βάσης κληθήκαμε να δημιουργήσουμε κατάλληλο κατάλληλο user interface, όπου ένας κοινός χρήστης μπορεί με ευκολία να αναζητήσει δεδομένα βάσει ορισμένων κριτηρίων. Τα κριτήρια αυτά ορίζονται μέσω κατάλληλων queries υλοποιημένων σε SQL και η αναζήτηση των δεδομένων διευκολύνεται έχοντας ορίσει κατάλληλα indexes όπου χρειάζεται. Επιπλέον, ο χρήστης έχει την δυνατότητα να προσθέτει και να αφαιρεί δεδομένα μέσα στην Βάση και όπου αυτό δεν είναι δυνατό λόγο περιορισμό λαμβάνει σωστό μήνυμα για να τηρούνται οι κανόνες δημιουργίας της εργασίας.

Διαγράμματα

Παρακάτω φαίνεται το ER διάγραμμα που αποτελεί μέρος του πρώτου παραδοτέου της εργασίας:

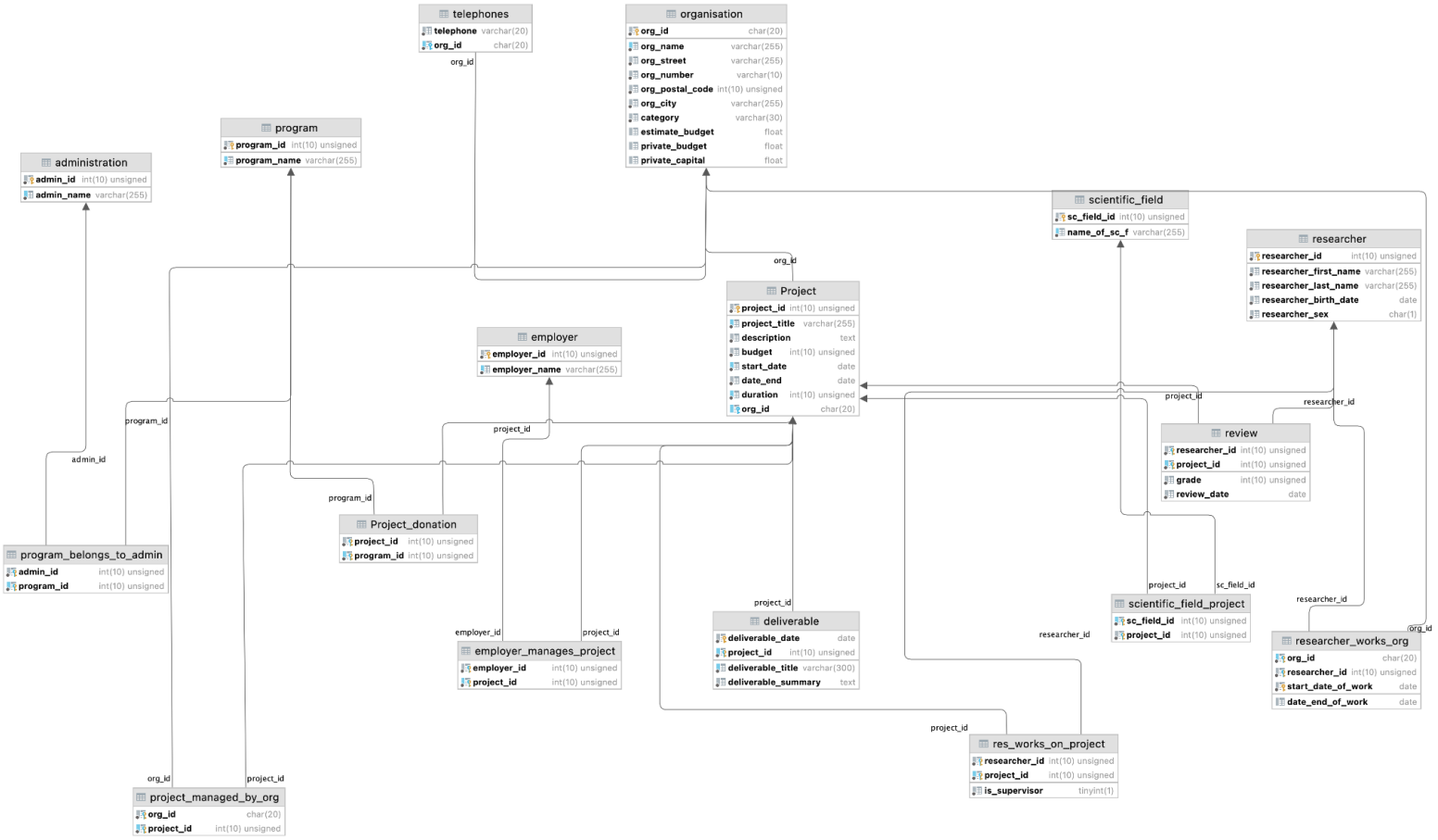


Επεξηγήσεις	
PK	Primary Key
DA	Discriminating Attribute
2..*	Πάνω από ένα
IS_A	Disjoint, OR

Ομάδα 93

Σαμπάνης Άθανασιος 03119045
Φρουδάκης Ευάγγελος 03119108
Χατζής Ιωάννης 03119923

Παρακάτω φαίνεται το Relational διάγραμμα που προέκυψε από την δική μας λύση του ER diagram (Παραδοτέο 1) και από την προτεινόμενη λύση που δόθηκε:



Το σχεσιακό διάγραμμα που φαίνεται παραπάνω έχει βασιστεί σε όλες τις σχέσεις του ER και τις αντίστοιχες οντότητες. Κάθε πίνακας έχει ορισμένα Primary Key και Foreign Key Constraints. Ωστόσο, για να καταλήξουμε σε αυτήν την μορφή έγινε σύμπτυξη ορισμένων πινάκων για την σωστή κανονικοποίηση και δημιουργήθηκαν τα κατάλληλα keys που συμβολίζονται με “ ”_ID, που τελικά οδηγούν στην βελτιστοποίηση της αναζήτησης των δεδομένων μέσα στην βάση. Επίσης, στον πίνακα telephones φαίνεται να μην υπάρχει primary key, κάτι που τελέστηκε για την δημιουργία του index telephones, με το οποίο τα τηλέφωνα αναζητούνται ευκολότερα μέσω του οργανισμού που ανήκουν.

Δημιουργία Βάσεις και Πινάκων

Στο σημείο αυτό φαίνεται το DDL και snippet του DML της βάσης. Παρακάτω φαίνονται τα Queries για την δημιουργία της βάσης και των πινάκων, ενδεικτικά check constraints triggers και indexes, data inserts και τέλος μέρος των Queries των ερωτημάτων που είναι ζητούμενα για το User Interface.

Query για δημιουργία της Βάσης:

```
create database DB_Project_ELIDEK;
```

Για την δημιουργία της βάσης χρησιμοποιούμε στο DBMS client και IDE που χρησιμοποιούμε την παραπάνω γραμμή κώδικα. Στη συνέχεια δημιουργούμε τους πίνακες της βάσης με την σειρά που που παρουσιάζονται κάτω.

Δημιουργία του πίνακα organisation:

```
CREATE TABLE organisation (
    org_id char(20) not null,
    org_name varchar(255) not null,
    org_street varchar(255) not null,
    org_number varchar(10) not null,
    org_postal_code int unsigned not null,
    org_city varchar(255) not null,
    category varchar(30) not null,
    estimate_budget float,
    private_budget float,
    private_capital float,
    CONSTRAINT PK_ORGANISATION PRIMARY KEY (org_id),
    CONSTRAINT UN_ORGANISATION UNIQUE (org_name),
    CONSTRAINT CHK_ORG_CATEGORY CHECK (category in ('UNIVERSITY', 'RESEARCH
CENTER', 'COMPANY'))
);
```

Αρχικά για την δημιουργία του πίνακα organisation, χρησιμοποιούμε το org_id σαν primary key, οπου αποτελεί μοναδικό char value. Στη συνέχεια, συμπληρώνουμε το όνομα του οργανισμού, την διεύθυνση και την κατηγορία του οργανισμού. Τα columns αυτά ορίζονται ως not null, καθώς πρέπει ο χρήστης υποχρεωτικά να τοποθετήσει στοιχεία στα πεδία αυτά για να καταχωρηθεί ο οργανισμός. Το ίδιο ισχύει για το column category το οποίο ωστόσο ελέγχεται από check constraint, ώστε να είναι ένα από τα τρία είδη που πρέπει να υπάρχουν. Για το είδος budget υπάρχει κατάλληλο trigger που ελέγχει αν ο χρήστης εισάγει σωστό είδος και ποσό budget, π.χ. αν ο χρήστης έχει δηλώσει την κατηγορία UNIVERSITY, και εισάγει σε λάθος κατηγορία ποσό διάφορο του μηδενός, τότε η εισαγωγή του διαγράφεται και του επιστρέφεται κατάλληλο error message.

Δημιουργία του πίνακα Project:

```
CREATE TABLE Project(
    project_id int unsigned not null AUTO_INCREMENT,
    org_id char(20) null,
      project_title varchar(255) not null,
    description text not null,
    budget int unsigned not null,
    start_date date not null,
    date_end date not null,
    duration int unsigned not null,
    CONSTRAINT PK_PROJECT PRIMARY KEY (project_id),
    CONSTRAINT UN_PROJECT_TILE UNIQUE (project_title),
    CONSTRAINT FK_ORG_ID_OF_PROJECT FOREIGN KEY (org_id) REFERENCES
organisation(org_id) ON DELETE SET NULL,
    CONSTRAINT CHK_BUDGET CHECK (budget >= 100000 and budget <= 1000000),
    CONSTRAINT CHK_date_end CHECK (date_end is null or (date_end >=
start_date + INTERVAL 1 YEAR and date_end <= start_date + INTERVAL 4 YEAR))
    );
```

Για την δημιουργία του πίνακα project, χρησιμοποιούμε αντίστοιχα ένα project_id primary key. Σε όποιο column χρειάζεται χρησιμοποιούμε την διευκρίνιση not null. Χρησιμοποιήθηκαν επίσης CHECK CONSTRAINTS για να ελέγχεται το σωστό εύρος τιμών για την χρηματοδότηση του έργου, αλλά και της διάρκειας του. Ακόμη δηλώνουμε το FOREIGN KEY CONSTRAINT για το org_id, και του προσθέτουμε την ιδιότητα σε περίπτωση διαγραφής κάποιου οργανισμού, να γίνεται ON DELETE SET NULL. Αυτό σε περίπτωση που ο χρήστης θέλει να διαγράψει κάποιον οργανισμό, να μην διαγράφονται τα έργα του, από τα logs του Ιδρύματος.

Δημιουργία του πίνακα program:

```
CREATE TABLE program(
    program_id int unsigned not null AUTO_INCREMENT,
    program_name varchar(255) not null,
    CONSTRAINT PK_PROGRAM PRIMARY KEY (program_id),
    CONSTRAINT UN_PROGRAM_NAME UNIQUE (program_name)
);
```

Δημιουργούμε τον πίνακα program ο οποίος παίρνει σαν columns ένα μοναδικό id του και το UNIQUE όνομα του. Θέτουμε το id ως PRIMARY KEY, και αυτά τα δύο μαζί με το UNIQUE CONSTRAINT αυτόματα θεωρούνται indexes της βάσης, διευκολύνοντας έτσι την αναζήτηση δεδομένων.

Query για δημιουργία του πίνακα Project_donation:

```
CREATE TABLE Project_donation(
       project_id int unsigned not null,
    program_id int unsigned not null,
    CONSTRAINT PK_PROJECT_DONATION PRIMARY KEY(project_id, program_id),
    CONSTRAINT FK_PROJDON_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id),
    CONSTRAINT FK_PROJDON_PROGRAM_ID FOREIGN KEY (program_id) REFERENCES
program(program_id));
```

Χρησιμοποιούμε τον παραπάνω πίνακα project donation για την υλοποίηση της σχέσης program χρηματοδοτεί κάποιο project. Επειδή, η σχέση είναι 1..* δηλαδή ένα project χρηματοδοτείται από ένα μόνο project, ενώ το program χρηματοδοτεί μη ορισμένο αριθμό projects, χρησιμοποιούμε συνδυαστικό CONSTRAINT PRIMARY KEY που περιέχει (project_id, program_id).

Δημιουργία του πίνακα administration:

```
CREATE TABLE administration(
    admin_id int unsigned not null AUTO_INCREMENT,
      admin_name varchar(255) not null,
    CONSTRAINT PK_ADMINISTRATION PRIMARY KEY (admin_id),
    CONSTRAINT UN_ADMINISTRATION UNIQUE (admin_name));
```

Με πιο απλό τρόπο ορίζουμε τον πίνακα administration ο οποίος περιέχει PRIMARY KEY το μοναδικό admin_id και το UNIQUE όνομα της διεύθυνσης.

Δημιουργία του πίνακα deliverable:

```
CREATE TABLE deliverable(
    deliverable_title varchar(300) not null,
    deliverable_date date not null,
    project_id int unsigned not null,
    deliverable_summary text not null,
    CONSTRAINT PK_DELIVERABLE PRIMARY KEY (deliverable_date, project_id),
    CONSTRAINT UN_DELIVERABLE UNIQUE (deliverable_title),
    CONSTRAINT FK_DELIV_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id)
);
```

Για τον πίνακα των παραδοτέων δημιουργούμε τα κατάλληλα deliverable_title που είναι UNIQUE και deliverable_date που με CONSTRAINT ορίζονται ως PRIMARY KEY του πίνακα, μιας και δίνουν την μοναδικότητα του στοιχείου αυτού για την βάση μας. Επίσης, Γίνεται κατάλληλη διευκρίνιση του project_id ως FOREIGN KEY αφού προκύπτει από το ήδη υπάρχον project_id που έχει ήδη εισαχθεί στον πίνακα Project.

Δημιουργία του πίνακα employer:

```
CREATE TABLE employer(
    employer_id int unsigned not null AUTO_INCREMENT,
    employer_name varchar(255) not null,
    CONSTRAINT PK_EMPLOYER PRIMARY KEY (employer_id),
    CONSTRAINT UN_EMPLOYER_MANAGERS_PROJECT UNIQUE (employer_name)
);
```

Στον πίνακα employer δηλώνουμε τα στελέχη ΕΛ.ΙΔ.Ε.Κ. Στο σημείο αυτό παρόμοια δημιουργούμε employer_id και employer_name (δηλώνεται UNIQUE column), ωστόσο χρειάζεται να σχολιάσουμε την δήλωση `AUTO_INCREMENT` με την οποία σε περίπτωση που ο χρήστης δεν δώσει κάποιο δικό του ID, η βάση δίνει αυτόματα κάποιο ID.

Δημιουργία του πίνακα employer_manages_project:

```
CREATE TABLE employer_manages_project (
    employer_id int unsigned not null,
    project_id int unsigned not null,
    CONSTRAINT PK_EMPLOYER_MANAGERS_PROJECT PRIMARY KEY (employer_id, project_id),
    CONSTRAINT FK_EMPLOYER_ID FOREIGN KEY (employer_id) REFERENCES
employer(employer_id),
    CONSTRAINT FK_EMP_MAN_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id)
);
```

Στην σχέση στέλεχος διαχειρίζεται έργο,δημιουργούμε τον αντίστοιχο πίνακα ο οποίος περιέχει δύο FOREIGN KEY `project_id,employer_id` τα οποία αποτελούν και συνδυαστικά και PRIMARY KEY του TABLE.

Δημιουργία του πίνακα program_belongs_to_admin:

```
CREATE TABLE program_belongs_to_admin(
    admin_id int unsigned not null,
    program_id int unsigned not null,
    CONSTRAINT PK_PROGRAM_BELONGS_TO_ADMIN PRIMARY KEY (admin_id, program_id),
    CONSTRAINT FK_PRO_BEL_TO_ADMIN_ADMIN_ID FOREIGN KEY (admin_id) REFERENCES
administration(admin_id),
    CONSTRAINT FK_PRO_BEL_TO_ADMIN_PROGRAM_ID FOREIGN KEY (program_id) REFERENCES
program(program_id)
);
```

Αντίστοιχα δημιουργείται και ο πίνακας program_belongs_to_admin που δείχνει την σχέση ότι κάθε πρόγραμμα επιχορήγησης ανήκει σε μία διεύθυνση του ΕΛ.ΙΔ.Ε.Κ. Εδώ υπάρχουν FOREIGN KEY `admin_id, program_id` που με `CONSTRAINT PRIMARY KEY` ορίζονται σαν PRIMARY KEY του TABLE.

Δημιουργία του πίνακα project_managed_by_org:

```
create table project_managed_by_org(
```

```
    org_id char(20) not null,
    project_id int unsigned not null,
    CONSTRAINT PK_PROJECT_MANAGED_BY_ORG PRIMARY KEY (org_id, project_id),
    CONSTRAINT FK_PRO_MAN_BY_ORG_ORG_ID FOREIGN KEY (org_id) REFERENCES
organisation(org_id),
    CONSTRAINT FK_PRO_MAN_BY_ORG_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id)
);
```

Αντίστοιχα στον πίνακα έργο διαχειρίζεται από οργανισμό, δηλώνουμε τα `org_id, project_id` ως FOREIGN KEYS από τους πίνακες που πηγάζουν και το RPIMARY KEY (`org_id, project_id`).

Query για δημιουργια του πίνακα researcher:

```
CREATE TABLE researcher (
    researcher_id int unsigned not null AUTO_INCREMENT,
    researcher_first_name varchar(255) not null,
    researcher_last_name varchar(255)  not null,
    researcher_birth_date date not null,
    researcher_sex char(1) not null,
    CONSTRAINT PK_RESEARCHER PRIMARY KEY (researcher_id),
    CONSTRAINT CHK_RESEARCHER_SEX CHECK (researcher_sex in ('M', 'F', 'O'))
);
```

Δημιουργία του πίνακα researcher, με το ID του ως PRIMARY KEY, μιας και τα ονόματα δεν μπορούν να είναι UNIQUE. Δηλώνουμε not null σε κάθε πεδίο.

Δημιουργία του πίνακα res_works_on_project:

```
CREATE TABLE res_works_on_project (
    researcher_id int unsigned not null,
    project_id int unsigned not null,
    is_supervisor boolean not null,
    CONSTRAINT PK_RES_WORKS_ON_PROJECT PRIMARY KEY (researcher_id, project_id),
    CONSTRAINT FK_RES_WORKS_ON_PRO_PROJECT_ID FOREIGN KEY (project_id)
REFERENCES Project(project_id),
    CONSTRAINT FK_RES_WORKS_ON_PRO_RESEARCHER_ID FOREIGN KEY (researcher_id)
REFERENCES researcher(researcher_id)
);
```

Για τον πίνακα researcher work on project εκτός από το id του έργου που εργάζεται ο ερευνητής και το μοναδικό του id που αποτελούν FOREIGN KEYS και PRIMARY KEY δηλώνουμε το attribute is_supervisor που έχει οριστεί ως boolean και ελέγχει αν ο ερευνητής είναι επιστημονικός υπεύθυνος του αντίστοιχου έργου που εργάζεται.

Δημιουργία του πίνακα researcher_works_org:

```
CREATE TABLE researcher_works_org (
    org_id char(20) not null,
    researcher_id int unsigned not null,
    start_date_of_work date not null,
    date_end_of_work date,
    CONSTRAINT PK_RESEARCHER_WORKS_ORG PRIMARY KEY (researcher_id, org_id),
    CONSTRAINT FK_RES_WORKS_ORG_ORG_ID FOREIGN KEY (org_id) REFERENCES
organisation(org_id),
    CONSTRAINT FK_RES_WORKS_ORG_RESEARCHER_ID FOREIGN KEY (researcher_id)
REFERENCES researcher(researcher_id),
    CONSTRAINT CHK_date_end_WORK CHECK (date_end_of_work is null or
date_end_of_work >= start_date_of_work)
);
```

Για την σχέση ερευνητής εργάζεται σε οργανισμό εκτός από τα κατάλληλα FOREIGN KEYS και PRIMARY KEYS (`researcher_id, org_id`), ορίζουμε CHECK CONSTRAINT που ελέγχει ότι η εισαγόμενη ημερομηνία τέλους της εργασίας στον οργανισμό είναι λογικά τοποθετημένη >=`start_date_of_work` .Επιπροσθέτως, στο πεδίο `date_end_of_work` δεν ορίζουμε ως not null καθώς μπορεί ο ερευνητής να είναι αορίστου χρόνου εργαζόμενος.

Δημιουργία του πίνακα review:

```
CREATE TABLE review(
    researcher_id int unsigned not null,
    project_id int unsigned not null,
    grade int unsigned not null,
    review_date date not null,
    CONSTRAINT PK_REVIEW PRIMARY KEY (researcher_id, project_id),
    CONSTRAINT FK_REV_RESEARCHER_ID FOREIGN KEY (researcher_id) REFERENCES
researcher(researcher_id),
    CONSTRAINT FK_REV_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id),
    CONSTRAINT CHK_GRADE CHECK (grade >= 0 AND grade <= 10)
);
```

Δηλώνουμε τα κατάλληλα FOREIGN KEYS για `researcher_id, project_id` που αποτελούν και το PRIMARY KEY του TABLE. Στη συνέχεια δηλώνουμε τον βαθμό int που έχει CHECK CONSTRAINT να είναι του φάσματος 0-10. Ωστόσο, ένα αρκετά σημαντικό CONSTRAINT που χρειάζεται να δημιουργήσουμε είναι ο έλεγχος ένας reviewer να μην εργάζεται σε κάποιο έργο, άρα και οργανισμό τον οποίο θα αξιολογήσει. (*Πιο κάτω δίνεται κατάλληλο trigger για τον έλεγχο της εγκυρότητας του review μέσα από το res_works_on_project*).

Δημιουργία του πίνακα scientific_field:

```
CREATE TABLE scientific_field (
    sc_field_id int unsigned not null AUTO_INCREMENT,
      name_of_sc_f varchar(255) not null,
    CONSTRAINT PK_SCIENTIFIC_FIELD PRIMARY KEY (sc_field_id),
    CONSTRAINT UN_SCIENTIFIC_FIELD UNIQUE (name_of_sc_f)
);
```

Δημιουργία του πίνακα με τα είδη επιστημονικών πεδίων. Κάθε πεδίο έχει το ξεχωριστό του ID που αποτελεί και PRIMARY KEY του πίνακα και το όνομα του Πεδίου που έχει διευκρινιστεί ως UNIQUE.

Δημιουργία του πίνακα scientific_field_project:

```
CREATE TABLE scientific_field_project (
    sc_field_id int unsigned not null,
    project_id int unsigned not null,
    CONSTRAINT PK_SCIENTIFIC_FIELD_PROJECT PRIMARY KEY (project_id, sc_field_id),
    CONSTRAINT FK_SCI_FIELD_PRO_SC_FIELD_ID FOREIGN KEY (sc_field_id) REFERENCES
scientific_field(sc_field_id),
    CONSTRAINT FK_SCI_FIELD_PRO_PROJECT_ID FOREIGN KEY (project_id) REFERENCES
Project(project_id)
);
```

Για τον πίνακα έργο ανήκει σε scientific field έχουμε σχέση *..* οπότε με τα FOREIGN KEYS (`project_id, sc_field_id`) που είναι ορισμένα και ως PRIMARY KEYS. Εδώ και όπου αλλού είναι ορισμένα PRIMARY KEYS θεωρούνται αυτόματα indexes, όπως ισχύει και για τα FOREIGN KEYS τα οποία σε ορισμένες περιπτώσεις δηλώνονται.

Query για τη δημιουργία του πίνακα telephones

```
CREATE TABLE telephones(
      telephone varchar(20) not null,
    org_id char(20) not null,
      CONSTRAINT FK_TELEPHONES FOREIGN KEY (org_id) REFERENCES
organisation(org_id)
);
CREATE INDEX IDX_TELEPHONES_ORG_ID ON telephones (org_id);
```

Στον πίνακα telephones δηλώνουμε το attribute telephone και χρησιμοποιούμε το org_id για να δείξουμε σε ποιον οργανισμό ανήκει.

Ωστόσο, για την καλύτερη λειτουργία της αναζήτησης τους, χρησιμοποιούμε `CREATE INDEX IDX_TELEPHONES_ORG_ID ON telephones (org_id)` το οποίο ευρετήριο επιστρέφει τα επιθυμητά αποτελέσματα που είναι τα τηλέφωνα, απλά αναζητώντας το org_id.

TRIGGER SAMPLE

```
CREATE TRIGGER review_ins
    BEFORE INSERT
    ON review FOR EACH ROW
BEGIN
  declare rowcount int;
```

```
  select count(1)
  into rowcount
  from res_works_on_project
  where researcher_id = new .researcher_id and
        project_id = new.project_id;
  if (rowcount = 1)
  then
   signal sqlstate '45000' set message_text = 'Error. Reviewer working on this
project, cannot review.';
  end if;
END;
```

Το trigger αυτό ενεργοποιείται πριν την εισαγωγη δεδομενων στον πινακα review και ελέγχει αν το researcher_id που εισάγεται διαφέρει από το id που είναι αποθηκευμενο στο table project. Αν δεν συμβαίνει κάτι τέτοιο εμφανίζεται στον χρήστη κατάλληλο μήνυμα Error καθώς ενας researcher δεν μπορεί να δουλεύει σε έναν οργανισμό και να κάνει review ενος project του ίδιου οργανισμού.

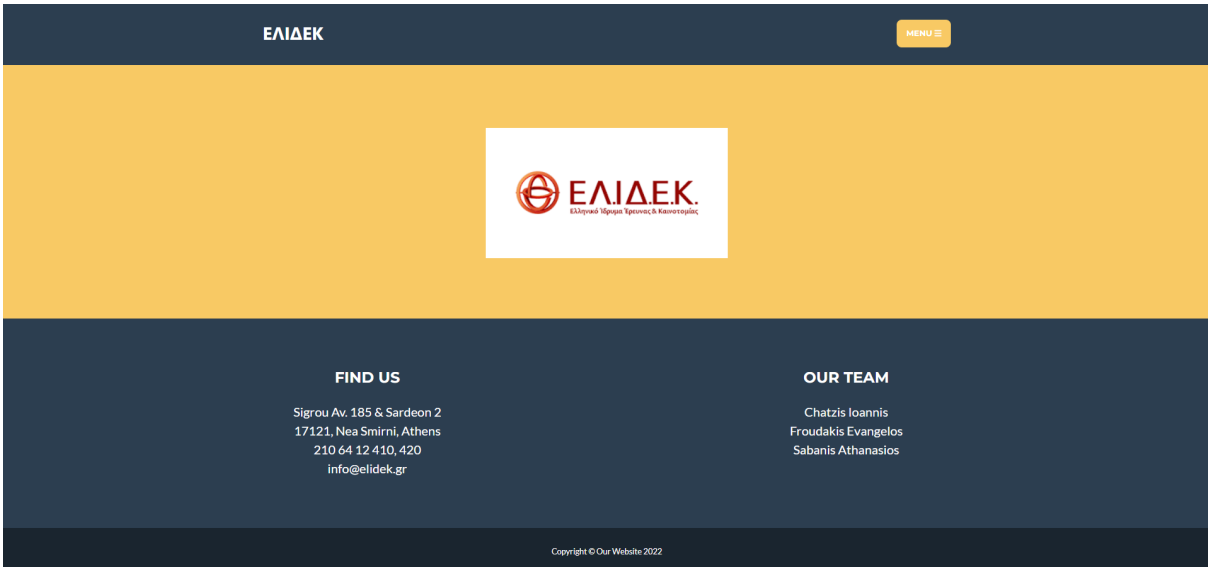
Query για την εισαγωγή data

```
INSERT INTO `employer` (`employer_id`, `employer_name`) VALUES
(2, 'Alfi Wavell'),
(27, 'Anya Henrionot'),
(30, 'Ariella Kolodziej'),
(23, 'Arlyne Igoe'),
(13, 'Barty Grayland'),
(19, 'Caroline Hillaby'),
(22, 'Caterina Ashborn'),
(6, 'Charmion Breznovic'),
(8, 'Danielle Scamaden');
```

Εδώ παρουσιάζεται ένα παράδειγμα Data Insert (Data Manipulation-DML) κατά το οποίο στο table employer γίνεται εισαγωγή στοιχείων στα employer_id και employer_name

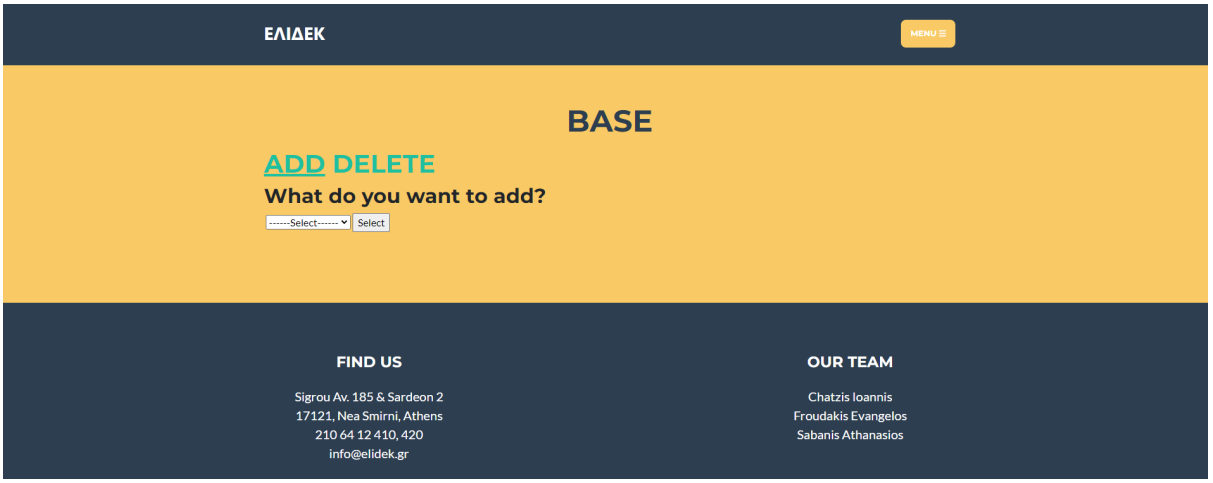
User Interface

Το UI προγραμματισμένο σε PHP, HTML, και στοιχεια απο CSS και JS. Ο χρηστης οταν ανοιγει την εφαρμογη βρισκεται στην αρχικη σελιδα.

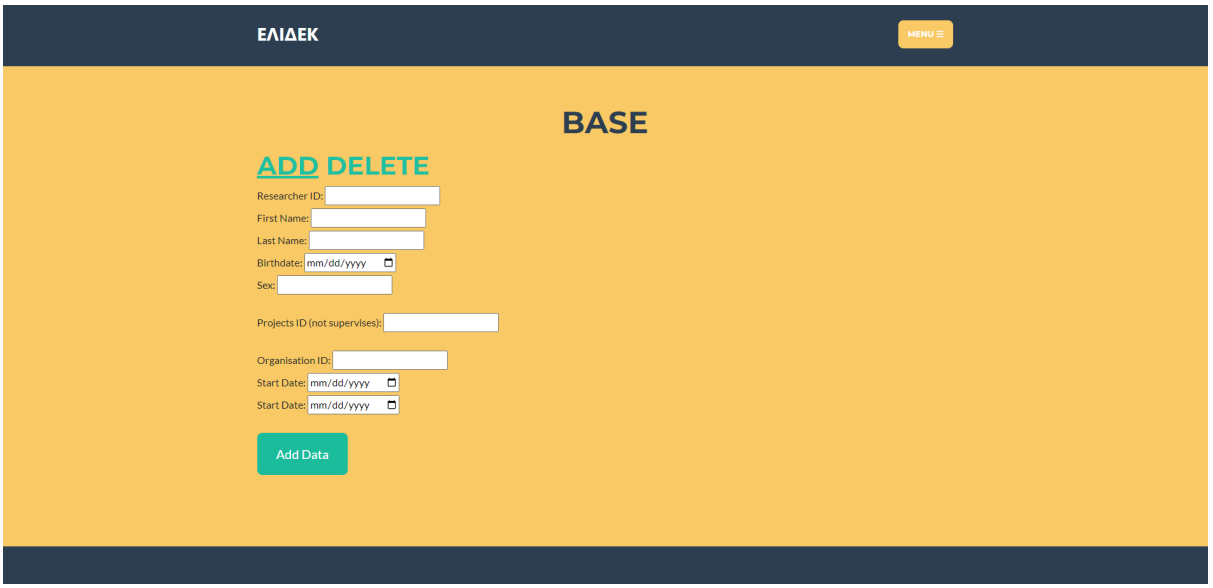


Πατώντας το πανω δεξια κουμπι που αναγραφει MENU βλεπει ενα drop down list με ολες τις επιλογες του που αποτελουντε απο επεξεργασια της βασης (BASE) και σελιδες για τα ερωτήματα της ασκησης (TEST 1-8).

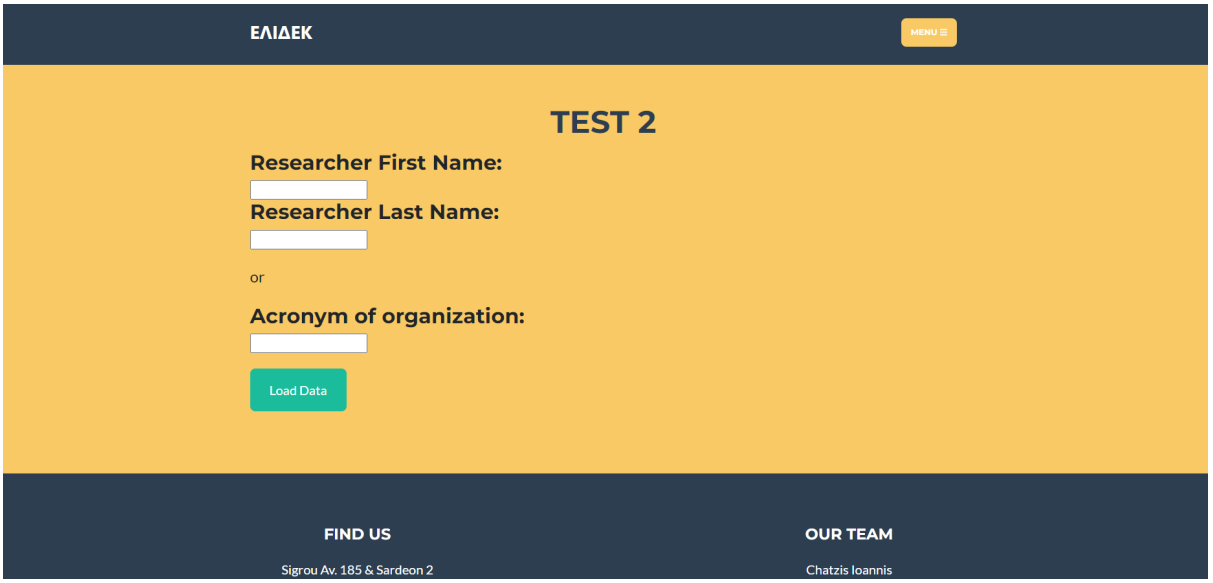
Μπαινοντας στην σελιδα για την επεξεργασια της βασης ο χρηστης μπορει να επιλεξει αν θελει να προσθεσει η να διαγραψει στοιχεια.



Επειτα απο αυτο πρεπει να δωσει τις καταλληλες πληροφοριες για να εκτελεστει η δραση που επελεξε. Για παράδειγμα παρακάτω φαίνονται τα πεδια που εμφανιζονται αν επιλεξει οτι θελει να προσθεσει εναν ερευνητη:



Παρακατω φαινεται ενδεικτικα και η σελιδα που βλεπει ο χρηστης αν επιλεξει να μπει στο TEST 2 (υποερωτημα 3.2). Αξιζει να σημειωθει οτι απο οποιαδηποτε σελιδα μπορει μεσω του menu να γυρισει στην αρχικη, αλλα και να παει και σε οποιαδηποτε αλλη.



Οδηγίες εγκατάστασεις για windows και MacOS

Απαιτούμενες Εφαρμογες:

-MySQL (DBMS)

-MAMP (Web Dev Stack)

Οδηγίες:

1.Αφού εγκαταστήσουμε το MAMP συνδεόμαστε στον server μεσω του OApache και ελεγχουμε τα ports για το MySQL (Apache Port: 8888, MySQL Port:8889) συνδεόμαστε με username και password “root” και “root” αντίστοιχα στο MySQL Workbench.

2.Εκτελούμε το script για τη δημιουργία της βάσης, επιλέγουμε τη βάση και εκτελούμε το script το οποίο περιέχει DDL και DML.(το scrip υπάρχει στο directory backup στο gitrepo)

3. Αντιγράφουμε τα αρχεια που βρίσκονται στο UI directory στο gitrepo στον φακελο htdocs του MAMP.

4. Τελος ανοίγουμε στον φυλλομετρητή της επιλογης μας την διευθηνση *http://localhost/index.php* και θα δουμε την αρχικη σελιδα της εφαρμογης.

Το git repo του project: https://github.com/jcchatzis/DB_Project_Elidek