

Μάθημα: Βάσεις Δεδομένων

Εξαμηνιαία Εργασία

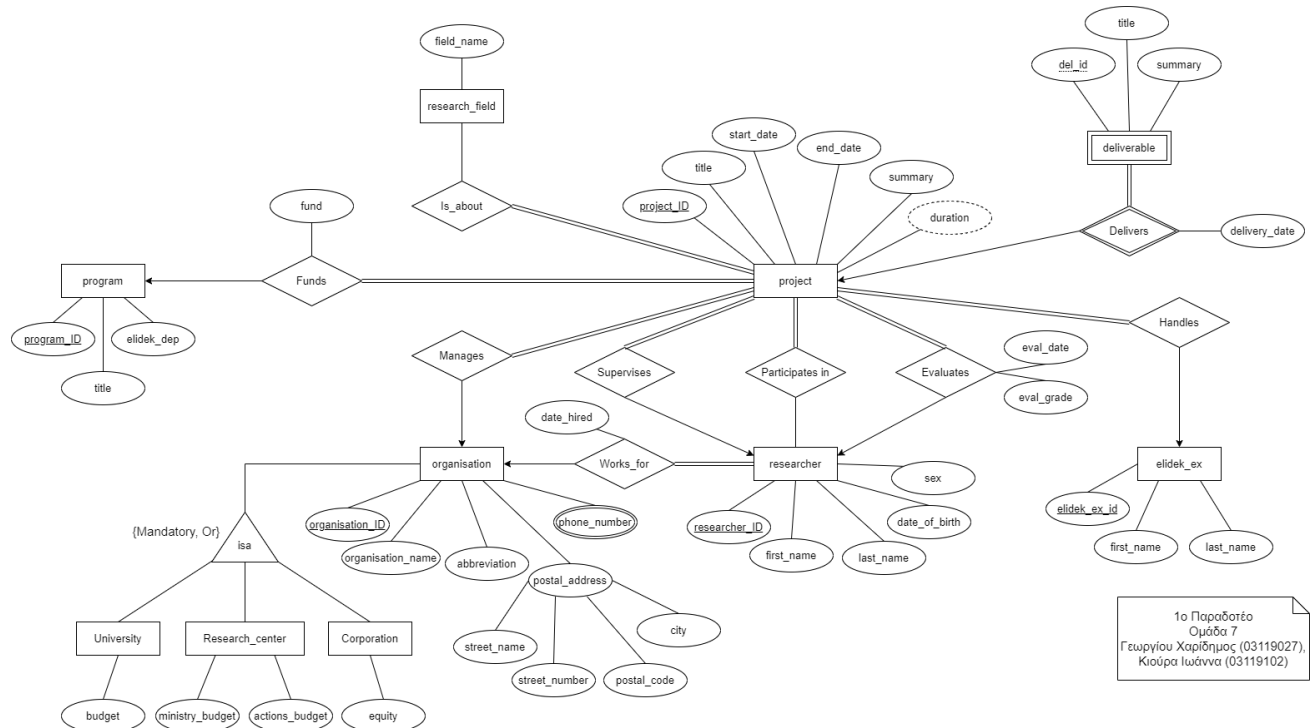
Ομάδα: 7

Μέλη: Γεωργίου Χαρίδημος (03119027), Κιούρα Ιωάννα (03119102)

Link gitrepo: <https://github.com/ChGeorgiou/dbclass2022.git>

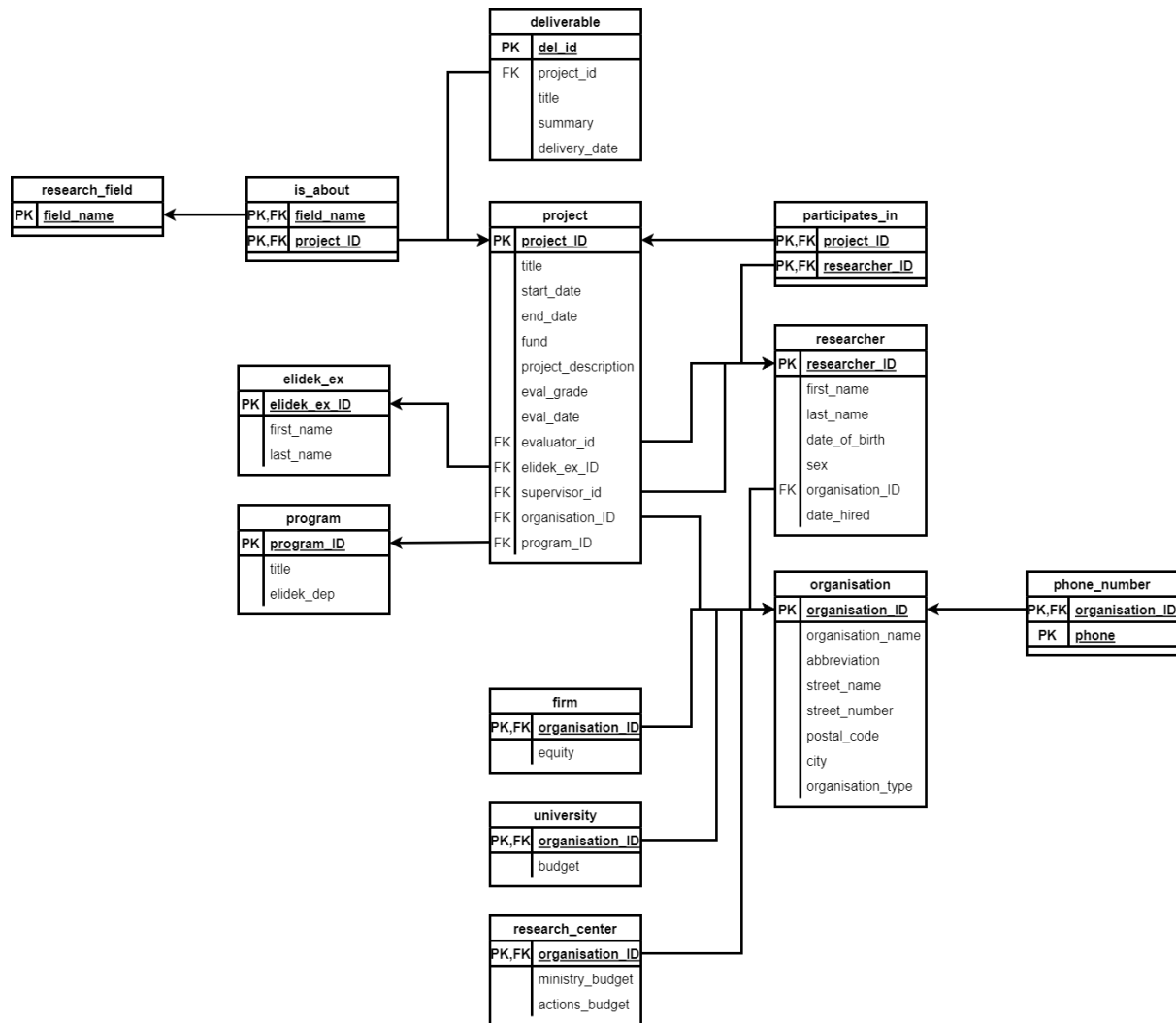
ER DIAGRAM

Για την υλοποίηση του 2ου παραδοτέου χρησιμοποιήθηκε μια παραλλαγή του ER που παραδώσαμε:



RELATIONAL DIAGRAM

Το σχεσιακό διάγραμμα που προέκυψε από αυτό είναι το εξής:



Από το ER Diagram στο Relational Diagram:

Οι σχέσεις funds, evaluates, supervises, works for, delivers, handles και manages είναι σχέσεις 1-N με total participation στην πλευρά του N, οπότε δεν υπάρχουν στο Relational, αλλά χρησιμοποιήθηκαν foreign keys για να αντιπροσωπευτούν. Για παράδειγμα, για την σχέση funds, στην πλευρά του project, που έχει total participation και μπορεί να έχει ακριβώς 1 πρόγραμμα που το χρηματοδοτεί, μπήκε στα attributes του το id του προγράμματος που το χρηματοδότησε καθώς και το attribute fund της σχέσης funds. Με παρόμοιο τρόπο μεταφέρθηκαν στο relational και οι υπόλοιπες σχέσεις του ίδιου τύπου. Οι μόνες σχέσεις που μεταφέρονται αυτούσιες στο relational είναι η is_about και η participates_in. Αυτές έγιναν δύο tables με attributes μόνο τα foreign keys των δύο άκρων τους, αφού δεν είχαν attributes οι ίδιες. Το derived attribute duration δεν υπάρχει στο σχεσιακό. Το multivalued attribute phone_number έγινε, σύμφωνα με την θεωρία, ξεχωριστό table με foreign key το primary key του οργανισμού και primary key την τούπλα organisation_id, phone. Για την αναπαράσταση του isa, χρησιμοποιήσαμε εξειδίκευση του οργανισμού σε firm, university, research_center με foreign key το id του οργανισμού.

Με βάση το παραπάνω σχεσιακό ξεκινήσαμε να κατασκευάζουμε την βάση μας με τις ακόλουθες εντολές:

- 1 • DROP DATABASE IF EXISTS elidek;
- 2 • CREATE DATABASE elidek;
- 3 • USE elidek;

CREATE TABLES

Έπειτα δημιουργήθηκαν τα απαραίτητα tables. Πρώτο δημιουργήθηκε το table του οργανισμού (**organisation**):

```
5 • CREATE TABLE IF NOT EXISTS organisation (  
6     organisation_id INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,  
7     organisation_name VARCHAR(45) UNIQUE NOT NULL,  
8     city VARCHAR(45) NOT NULL,  
9     street_name VARCHAR(45) NOT NULL,  
10    street_number INT(5) NOT NULL,  
11    postal_code INT(5) NOT NULL,  
12    abbreviation VARCHAR(45) NULL DEFAULT 'None',  
13    organisation_type VARCHAR(15) NOT NULL,  
14    budget1 BIGINT(12) NOT NULL,  
15    budget2 BIGINT(12) NULL,  
16    constraint has_type check(organisation_type in ('Research Center','Firm','University')),  
17    PRIMARY KEY (organisation_id)  
18 );
```

Καθώς και τον τύπων οργανισμών που έχουμε (υλοποίηση του IsA):

```
30 • CREATE TABLE IF NOT EXISTS research_center (  
31     organisation_id INT(10) UNSIGNED NOT NULL,  
32     organisation_type VARCHAR(15) NOT NULL DEFAULT ('Research Center'),  
33     ministry_budget BIGINT(12) NOT NULL,  
34     actions_budget BIGINT(12) NOT NULL,  
35     constraint has_type check(organisation_type in ('Research Center')),  
36     PRIMARY KEY (organisation_id),  
37     FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)  
38         ON UPDATE CASCADE  
39         ON DELETE CASCADE  
40 );
```

```

42 • ⊖ CREATE TABLE IF NOT EXISTS university (
43     organisation_id INT(10) UNSIGNED NOT NULL,
44     organisation_type VARCHAR(15) NOT NULL DEFAULT ('University'),
45     budget BIGINT(12) NOT NULL,
46     constraint has_type check(organisation_type in ('University')),
47     PRIMARY KEY (organisation_id),
48     FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)
49         ON UPDATE CASCADE
50         ON DELETE CASCADE
51 );

53 • ⊖ CREATE TABLE IF NOT EXISTS firm (
54     organisation_id INT(10) UNSIGNED NOT NULL,
55     organisation_type VARCHAR(15) NOT NULL DEFAULT ('Firm'),
56     equity BIGINT(12) NOT NULL,
57     constraint has_type check(organisation_type in ('Firm')),
58     PRIMARY KEY (organisation_id),
59     FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)
60         ON UPDATE CASCADE
61         ON DELETE CASCADE
62 );

```

Αλλά και της εξαρτημένης από τον οργανισμό οντότητας των τηλεφωνικών αριθμών:

```

20 • ⊖ CREATE TABLE IF NOT EXISTS phone_number (
21     phone_id INT(10) UNSIGNED NOT NULL UNIQUE AUTO_INCREMENT,
22     organisation_id INT(10) UNSIGNED NOT NULL,
23     phone BIGINT(10) UNSIGNED NOT NULL,
24     FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)
25         ON UPDATE CASCADE
26         ON DELETE CASCADE,
27     PRIMARY KEY (organisation_id, phone)
28 );

```

Παρατηρήσεις:

1. Primary key για τον οργανισμό, τα πανεπιστήμια, τα ερευνητικά κέντρα και τις εταιρείες είναι το id του οργανισμού, που αποδίδεται σε κάθε νέα εισαγωγή αυτόματα από το σύστημα και είναι μοναδικό. Για τα πανεπιστήμια, τα ερευνητικά κέντρα και τις εταιρείες αυτό το id είναι επίσης foreign key που αντιστοιχεί στο ίδιο id ενός οργανισμού. Σε περίπτωση που διαγραφεί ο οργανισμός στον οποίο αντιστοιχεί, διαγράφεται και το αντίστοιχο tuple στην υποκατηγορία του. Σε περίπτωση που το id αυτό ενημερωθεί, ενημερώνεται και στο αντίστοιχο tuple στην υποκατηγορία του.
2. Θεωρήθηκε ότι δύο οργανισμοί δεν θα επιτρέπεται να έχουν το ίδιο όνομα στο σύστημα και έτσι προστέθηκε στο organisation_name η χαρακτηριστική λέξη UNIQUE.
3. Στο table organisation προστέθηκαν τα attributes: budget1, budget2. Αυτό έγινε ώστε να μπορεί να κάνει ένας χρήστης insert έναν οργανισμό και αυτόματα να γίνεται insert

και στο αντίστοιχο table του ίδιου τύπου με την χρήση ενός trigger. Συγκεκριμένα, το budget1 δεν μπορεί να πάρει NULL τιμές και αντιστοιχεί στα attributes equity, ministry_budget και budget των firm, research_center και university αντίστοιχα. Το budget2 μπορεί να έχει τιμή NULL και θα λαμβάνεται υπόψη μόνο όταν εισάγεται ένα research_center και θα αντιστοιχίζεται στο attribute του actions_budget. Το trigger που προκαλεί αυτό το έξτρα insert εμφανίζεται αργότερα στην αναφορά.

4. Προστέθηκε σε καθένα από τα tables firm, university, research_center ένα παραπάνω attribute, το organisation_type. Για το καθένα από αυτά τα tables το attribute αυτό μπορεί να έχει μόνο μία τιμή και αυτή είναι ο τύπος του συγκεκριμένου table (firm, university, research center). Αυτό χρειάζεται επίσης για την υλοποίηση ενός trigger που πριν, γίνει μία εισαγωγή ή μια ενημέρωση στο αντίστοιχο table, ελέγχει ότι ο οργανισμός που πάει να εισαχθεί πχ ως firm έχει όντως organisation_type = firm. Και αυτό το trigger θα εμφανιστεί αργότερα στην αναφορά μας.
5. Δημιουργήθηκε και ένας περιορισμός για το table οργανισμός ώστε το attribute organisation_type να έχει τιμές μόνο τις Firm, Research Center, University όπως ζητείται και από την εκφώνηση.
6. Όταν διαγράφεται ένας οργανισμός διαγράφονται επίσης και οι αριθμοί τηλεφώνου που του αντιστοιχούν.
7. Τέλος, προστέθηκε στο table phone_number ένα παραπάνω attribute, το phone_id. Αυτό προστέθηκε για να διευκολύνει την υλοποίηση του UI.

Έπειτα δημιουργήθηκε το table του ερευνητή (**researcher**):

```
64 ● ○ CREATE TABLE IF NOT EXISTS researcher (  
65     researcher_id int(10) UNSIGNED NOT NULL AUTO_INCREMENT,  
66     first_name varchar(45) NOT NULL,  
67     last_name varchar(45) NOT NULL,  
68     sex varchar(6) NOT NULL,  
69     date_of_birth DATE NOT NULL,  
70     date_hired DATE NOT NULL,  
71     organisation_id INT(10) UNSIGNED NOT NULL,  
72     CONSTRAINT sex_value CHECK (sex IN ('male', 'female')),  
73     CONSTRAINT age_hired CHECK (DATEDIFF(date_hired, date_of_birth) >= 18*365.25),  
74     PRIMARY KEY (researcher_id),  
75     FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)  
76         ON UPDATE CASCADE  
77         ON DELETE CASCADE  
78 ) ;
```

Παρατηρήσεις:

1. Primary key του ερευνητή είναι ένα id που αποδίδεται αυτόματα από την βάση και είναι μοναδικό.
2. Το id του οργανισμού στον οποίο δουλεύει ο κάθε ερευνητής είναι foreign key που αναφέρεται στον οργανισμό. Αν διαγραφεί ο οργανισμός, διαγράφονται και όλοι οι ερευνητές που δουλεύουν σε αυτόν.
3. Εισήχθει ένας περιορισμός για το φύλλο του κάθε υπαλλήλου (αρσενικό, θηλυκό).

4. Εισήχθει ένας περιορισμός ώστε να κατά την ημερομηνία πρόσληψης ενός researcher, αυτός να ήταν τουλάχιστον 18 χρονών. Έχει εισαχθεί επίσης ένα trigger που ελέγχει αν ο εργαζόμενος researcher είναι τουλάχιστον 18 χρονών και θα φανεί παρακάτω.

Έπειτα δημιουργούμε τα προγράμματα (**programs**):

```
82 • ○ CREATE TABLE IF NOT EXISTS program (  
83     program_id INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,  
84     title VARCHAR(45) NOT NULL,  
85     elidek_dep VARCHAR(45) NOT NULL,  
86     PRIMARY KEY (program_id)  
87 );
```

Παρατηρήσεις:

1. Primary key εδώ είναι ένα ID που αποδίδει αυτόματα το σύστημα.

Στην συνέχεια έχουμε την δημιουργία του table των στελεχών του ΕΛΙΔΕΚ (**elidek_ex**):

```
89 • ○ CREATE TABLE IF NOT EXISTS elidek_ex (  
90     elidek_ex_id INT(5) UNSIGNED NOT NULL AUTO_INCREMENT,  
91     first_name VARCHAR(45) NOT NULL,  
92     last_name VARCHAR(45) NOT NULL ,  
93     PRIMARY KEY (elidek_ex_id)  
94 );
```

Παρατηρήσεις:

1. Primary key εδώ είναι ένα ID που αποδίδει αυτόματα το σύστημα.

Και τώρα το table του έργου (**project**):

```

96 • CREATE TABLE IF NOT EXISTS project(
97     project_id INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
98     start_date DATE NOT NULL,
99     end_date DATE NOT NULL,
100    duration INT(5) NULL,
101    fund BIGINT(10) UNSIGNED NOT NULL,
102    project_title VARCHAR(100) UNIQUE NOT NULL,
103    project_description VARCHAR(255),
104    elidek_ex_id INT(5) UNSIGNED NOT NULL,
105    program_id INT(10) UNSIGNED NOT NULL,
106    organisation_id int(10) UNSIGNED NOT NULL,
107    supervisor_id INT(10) UNSIGNED NOT NULL,
108    evaluator_id INT(10) UNSIGNED NOT NULL,
109    eval_grade INT(3) UNSIGNED NOT NULL,
110    eval_date DATE NOT NULL,
111    FOREIGN KEY (elidek_ex_id) REFERENCES elidek_ex (elidek_ex_id)
112        ON UPDATE CASCADE ON DELETE RESTRICT,
113    FOREIGN KEY (program_id) REFERENCES program (program_id)
114        ON UPDATE CASCADE ON DELETE RESTRICT,
115    FOREIGN KEY (organisation_id) REFERENCES organisation (organisation_id)
116        ON UPDATE CASCADE ON DELETE RESTRICT,
117    FOREIGN KEY (supervisor_id) REFERENCES researcher (researcher_id)
118        ON UPDATE CASCADE ON DELETE RESTRICT,
119    FOREIGN KEY (evaluator_id) REFERENCES researcher (researcher_id)
120        ON UPDATE CASCADE ON DELETE RESTRICT,
121    PRIMARY KEY(project_id),
122    CONSTRAINT fund_size CHECK (fund >= 100000 AND fund <= 1000000),
123    CONSTRAINT duration_length CHECK (duration <= 4 AND duration >= 1),
124    CONSTRAINT start_end CHECK (start_date < end_date),
125    CONSTRAINT evaluation CHECK (start_date >= eval_date)
126 );

```

Παρατηρήσεις:

1. Primary key εδώ είναι ένα ID που αποδίδει αυτόματα το σύστημα.
2. Κάνουμε την παραδοχή ότι η βάση μας δεν θα επιτρέψει να υπάρχουν έργα που έχουν τον ίδιο τίτλο και έτσι ο τίτλος του έργου project_title τέθηκε UNIQUE
3. Η χρονική διάρκεια του έργου σε χρόνια duration, παρόλου που είναι derived attribute, μπήκε στο table και παίρνει τιμή με την βοήθεια ενός trigger που χρησιμοποιεί τις ημερομηνίες έναρξης και λήξης του έργου για να υπολογίσει την διάρκειά του.
4. Τα attributes elidek_ex_id (το στέλεχος ΕΛΙΔΕΚ που διαχειρίζεται το έργο), organisation_id (ο οργανισμός που διαχειρίζεται το έργο), supervisor_id (ο επιστημονικός υπεύθυνος του έργου), evaluator_id (ο αξιολογητής του έργου), program_id (το πρόγραμμα στο οποίο ανήκει το έργο) είναι foreign keys και σε

περίπτωση που κάποιο από αυτά ανανεωθεί, ανανεώνεται και για το έργο, ενώ παράλληλα δεν επιτρέπεται να διαγραφούν όσο υπάρχει το έργο.

5. Το attribute fund, το ποσό δηλαδή της χρηματοδότησης, έχει περιοριστεί μεταξύ των τιμών 100.000 και 1.000.000 όπως ζητήθηκε από την εκφώνηση.
6. Η διάρκεια ενός έργου σε χρόνια (duration) έχει περιοριστεί να κυμαίνεται από 1 έως 4 χρόνια, όπως ζητήθηκε από την εκφώνηση.
7. Έχει επιβληθεί περιορισμός η ημερομηνία έναρξης ενός έργου να είναι μικρότερη από την ημερομηνία λήξης του.
8. Έχει επιβληθεί περιορισμός η ημερομηνία αξιολόγησης του έργου να είναι μικρότερη από την ημερομηνία έναρξής του. Δηλαδή το έργο πρώτα αξιολογείται και μετά καθορίζεται η ημερομηνία έναρξής του.

Έπειτα δημιουργείται το table που εκφράζει την συμμετοχή ενός ερευνητή σε ένα έργο (**participates_in**):

```
129 • CREATE TABLE IF NOT EXISTS participates_in(  
130     p_id INT(10) UNSIGNED NOT NULL UNIQUE AUTO_INCREMENT,  
131     project_id int(10) UNSIGNED NOT NULL,  
132     researcher_id int(10) UNSIGNED NOT NULL,  
133     FOREIGN KEY (project_id) REFERENCES project (project_id)  
134         ON UPDATE CASCADE  
135         ON DELETE CASCADE,  
136     FOREIGN KEY (researcher_id) REFERENCES researcher (researcher_id)  
137         ON UPDATE CASCADE  
138         ON DELETE CASCADE,  
139     PRIMARY KEY (project_id, researcher_id)  
140 );
```

Παρατηρήσεις:

1. Primary key εδώ είναι η τούπλα (project_id, researcher_id) που προκύπτουν από το insert.
2. Προστέθηκε το attribute p_id για να διευκολυνθεί η κατασκευή του User Interface. Εμπρόκειτο για ένα ID που αποδίδει αυτόματα το σύστημα με κάθε insert.
3. Το project_id, researcher_id είναι επίσης foreign keys και αν ενημερωθούν, ενημερώνεται και η σχέση, ενώ αν διαγραφούν, διαγράφεται και η σχέση.

Έπειτα δημιουργούμε το table για τα παραδοτέα (**deliverables**):


```

142 • ○ CREATE TABLE IF NOT EXISTS deliverable (
143         del_id INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
144         project_id INT(10) UNSIGNED NOT NULL,
145         title VARCHAR(100) NOT NULL,
146         summary VARCHAR(300) NOT NULL,
147         delivery_date DATE NOT NULL,
148         FOREIGN KEY (project_id) REFERENCES project (project_id)
149             ON UPDATE CASCADE
150             ON DELETE CASCADE,
151         PRIMARY KEY (del_id)
152     );

```

Παρατηρήσεις:

1. Primary key εδώ είναι ένα ID που αποδίδεται αυτόματα από το σύστημα.
2. Το attribute project_id είναι foreign key με αναφορά στο project. Αν διαγραφεί το project που αντιστοιχεί σε αυτό το project_id διαγράφονται και όλα τα deliverables που έχουν ως foreign key αυτό το project_id. Αν το project_id ενημερωθεί, ενημερώνεται και στα deliverables που το έχουν.

Έπειτα δημιουργούμε το table για το επιστημονικό πεδίο (**research_field**):

```

154 • ○ CREATE TABLE IF NOT EXISTS research_field (
155         field_name VARCHAR(200) NOT NULL,
156         PRIMARY KEY (field_name)
157     );

```

Παρατηρήσεις:

1. Primary key εδώ είναι το όνομα του πεδίου, καθώς θεωρήθηκε απίθανο να υπάρχουν δύο πεδία με ίδιο όνομα.

Τέλος, δημιουργούμε το table **is_about** που εκφράζει σε ποιο επιστημονικό πεδίο ανήκει ένα έργο:

```

159 • CREATE TABLE IF NOT EXISTS is_about (
160     a_id INT(10) UNSIGNED NOT NULL UNIQUE AUTO_INCREMENT,
161     project_id INT(10) UNSIGNED NOT NULL,
162     field_name VARCHAR(200) NOT NULL,
163     FOREIGN KEY (project_id) REFERENCES project (project_id)
164         ON UPDATE CASCADE
165         ON DELETE CASCADE,
166     FOREIGN KEY (field_name) REFERENCES research_field (field_name)
167         ON UPDATE CASCADE
168         ON DELETE RESTRICT,
169     PRIMARY KEY (project_id, field_name)
170 );

```

Παρατηρήσεις:

1. Primary key εδώ είναι η τούπλα (project_id, field_name).
2. Τα project_id, field_name είναι επίσης και foreign keys και αν κάποιο ενημερωθεί, ενημερώνεται και στην is_about. Αν η οντότητα στην οποία αντιστοιχεί το project_id διαγραφεί, διαγράφονται και όλες οι σχέσεις που περιέχουν αυτό το project_id, ενώ η διαγραφή της οντότητας επιστημονικού πεδίου που αντιστοιχεί στο field_name δεν θα επιτραπεί.
3. Προστέθηκε επίσης ένα παραπάνω attribute, το a_id για να διευκολύνει την κατασκευή του User Interface.

TRIGGERS

Στην συνέχεια, για να εξασφαλιστεί η ορθότητα της βάσης μας εισήχθησαν και τα παρακάτω triggers:

1^ο Trigger:

```

175 • CREATE TRIGGER insert_sup AFTER INSERT ON project
176     FOR EACH ROW
177     BEGIN
178         INSERT INTO participates_in (researcher_id, project_id) VALUES (NEW.supervisor_id, NEW.project_id);
179     END$
180

```

Το παραπάνω trigger επεμβαίνει μετά από κάθε insert σε ένα έργο και βάζει τον ερευνητή που αντιστοιχεί στο supervisor_id του έργου να δουλεύει στο έργο.

2ο Trigger:

```

181 • CREATE TRIGGER insert_at_once AFTER INSERT ON organisation
182     FOR EACH ROW
183     BEGIN
184         IF (NEW.organisation_type = "University") THEN
185             INSERT INTO university (organisation_id,organisation_type,budget)
186             VALUES (NEW.organisation_id,NEW.organisation_type,NEW.budget1);
187         ELSEIF (NEW.organisation_type = "Firm") THEN
188             INSERT INTO firm (organisation_id,organisation_type,equity)
189             VALUES (NEW.organisation_id,NEW.organisation_type,NEW.budget1);
190         ELSEIF (NEW.organisation_type = "Research Center") THEN
191             INSERT INTO research_center (organisation_id,organisation_type,ministry_budget,actions_budget)
192             VALUES (NEW.organisation_id,NEW.organisation_type,NEW.budget1,NEW.budget2);
193         END IF;
194     END$

```

Αυτό το trigger αναφέρθηκε προηγουμένως και είναι αυτό που υλοποιεί την αυτόματη εισαγωγή των σχετικών attribute του οργανισμού στις οντότητες university, research_center, firm ανάλογα με το organisation_type του οργανισμού.

3ο Trigger

```

196 • CREATE TRIGGER update_at_once AFTER UPDATE ON organisation
197     FOR EACH ROW
198     BEGIN
199         IF (NEW.organisation_type = "University") THEN
200             UPDATE university
201             SET organisation_id = NEW.organisation_id, organisation_type = NEW.organisation_type, budget = NEW.budget1
202             WHERE organisation_id = NEW.organisation_id;
203         ELSEIF (NEW.organisation_type = "Firm") THEN
204             UPDATE firm
205             SET organisation_id = NEW.organisation_id, organisation_type = NEW.organisation_type, equity = NEW.budget1
206             WHERE organisation_id = NEW.organisation_id;
207         ELSEIF (NEW.organisation_type = "Research Center") THEN
208             UPDATE research_center
209             SET organisation_id = NEW.organisation_id, organisation_type = NEW.organisation_type,
210             ministry_budget = NEW.budget1, actions_budget = NEW.budget2
211             WHERE organisation_id = NEW.organisation_id;
212         END IF;
213     END$

```

Αυτό το trigger έχει την ίδια λειτουργία με το προηγούμενο με την διαφορά ότι επεμβαίνει στην περίπτωση που γίνεται update των δεδομένων του οργανισμού ώστε να γίνει ενημέρωση και στις οντότητες university, firm research_center ανάλογα με το organisation_type.

3ο Trigger:

```

215 • CREATE TRIGGER duration_insert BEFORE INSERT ON project
216     FOR EACH ROW
217     BEGIN
218         SET NEW.duration = DATEDIFF(NEW.end_date, NEW.start_date)/365.25;
219     END$

```

Αυτό το trigger αναφέρθηκε και παραπάνω και είναι αυτό που με κάθε εισαγωγή στο project ρυθμίζει το derived attribute duration να ισούται με την διαφορά της ημερομηνίας λήξης και της ημερομηνίας έναρξης του έργου σε χρόνια.

4ο Trigger:

```
221 • CREATE TRIGGER duration_update BEFORE UPDATE ON project
222     FOR EACH ROW
223     BEGIN
224         SET NEW.duration = DATEDIFF(NEW.end_date, NEW.start_date)/365.25;
225     END$
```

Αυτό το trigger λειτουργεί με τον ίδιο τρόπο με το προηγούμενο επεμβαίνοντας όμως μόνο πριν γίνει ένα update στα έργα.

5ο-6ο Triggers:

```
227 • CREATE TRIGGER ev_works_on_proj_insert BEFORE INSERT ON participates_in
228     FOR EACH ROW
229     BEGIN
230         IF ((SELECT evaluator_id FROM project WHERE project_id = NEW.project_id) = NEW.researcher_id) THEN
231             SIGNAL SQLSTATE '45000'
232             SET MESSAGE_TEXT = 'Error: A researcher can not work on a project he evaluated';
233         END IF;
234     END$

236 • CREATE TRIGGER ev_works_on_proj_update1 BEFORE UPDATE ON participates_in
237     FOR EACH ROW
238     BEGIN
239         IF ((SELECT evaluator_id FROM project WHERE project_id = NEW.project_id) = NEW.researcher_id) THEN
240             SIGNAL SQLSTATE '45000'
241             SET MESSAGE_TEXT = 'Error: A researcher can not work on a project he evaluated';
242         END IF;
243     END$
```

Αυτά τα trigger εξασφαλίζουν ότι ο αξιολογητής ενός έργου δεν θα μπορέσει ποτέ να δουλέψει σε αυτό, είτε με update είτε με insert στην participates_in.

7ο-8ο Triggers:

```
245 • CREATE TRIGGER ev_same_org_insert BEFORE INSERT ON project
246     FOR EACH ROW
247     BEGIN
248         IF ((SELECT organisation_id FROM researcher r WHERE r.researcher_id = NEW.evaluator_id) = NEW.organisation_id) THEN
249             SIGNAL SQLSTATE '45000'
250             SET MESSAGE_TEXT = 'Error: A researcher can not evaluate a project of an organisation he works for';
251         END IF;
252     END$

254 • CREATE TRIGGER ev_same_org_update BEFORE UPDATE ON project
255     FOR EACH ROW
256     BEGIN
257         IF ((SELECT organisation_id FROM researcher r WHERE r.researcher_id = NEW.evaluator_id) = NEW.organisation_id) THEN
258             SIGNAL SQLSTATE '45000'
259             SET MESSAGE_TEXT = 'Error: A researcher can not evaluate a project of an organisation he works for';
260         END IF;
261     END$
```

Αυτά τα triggers εξασφαλίζουν ότι ένας ερευνητής δεν θα μπορέσει να αξιολογήσει ένα έργο του οργανισμού για τον οποίο εργάζεται.

9ο Trigger:

```
263 • CREATE TRIGGER sup_work_on_proj_update AFTER UPDATE ON project
264     FOR EACH ROW
265     BEGIN
266     IF NOT EXISTS(SELECT researcher_id FROM participates_in
267         WHERE project_id = NEW.project_id AND researcher_id = NEW.supervisor_id) THEN
268         INSERT INTO participates_in (researcher_id, project_id) VALUES (NEW.supervisor_id, NEW.project_id);
269     END IF;
270     END$
```

Αυτό το trigger ενεργοποιείται όταν ένας ερευνητής γίνεται επιστημονικός υπεύθυνος ενός έργου και δουλειά του είναι να εισάγει στο table participates_in την τούπλα με το id του καινούριου επιστημονικού υπευθύνου και του id του έργου. Δηλαδή, αν βάλουμε ως επιστημονικό υπεύθυνο έναν ερευνητή, που πληρεί όλες τις υπόλοιπες προϋποθέσεις που θέτουν τα υπόλοιπα triggers, αλλά δεν δουλεύει ήδη στο έργο αυτό, τον βάζουμε να δουλεύει.

10ο-11ο Triggers:

```
272 • CREATE TRIGGER res_part_in_insert BEFORE INSERT ON participates_in
273     FOR EACH ROW
274     BEGIN
275     IF (SELECT organisation_id FROM researcher r WHERE r.researcher_id = NEW.researcher_id)
276     <> (SELECT organisation_id FROM project p WHERE p.project_id = NEW.project_id) THEN
277     SIGNAL SQLSTATE '45000'
278     SET MESSAGE_TEXT = 'Error: A researcher can not work on a project of an organisation he does not work for';
279     END IF;
280     END$

282 • CREATE TRIGGER res_part_in_update BEFORE UPDATE ON participates_in
283     FOR EACH ROW
284     BEGIN
285     IF (SELECT organisation_id FROM researcher r WHERE r.researcher_id = NEW.researcher_id)
286     <> (SELECT organisation_id FROM project p WHERE p.project_id = NEW.project_id) THEN
287     SIGNAL SQLSTATE '45000'
288     SET MESSAGE_TEXT = 'Error: A researcher can not work on a project of an organisation he does not work for';
289     END IF;
290     END$
```

Αυτά τα triggers εξασφαλίζουν ότι ένας ερευνητής που δεν δουλεύει σε έναν οργανισμό δεν θα μπορέσει να δουλέψει στα έργα του. Έτσι στα έργα ενός οργανισμού δουλεύουν μόνο ερευνητές του ίδιου οργανισμού.

12ο-17ο Triggers:

```
292 • CREATE TRIGGER res_type_insert BEFORE INSERT ON research_center
293     FOR EACH ROW
294     BEGIN
295     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
296     <> NEW.organisation_type THEN
297     SIGNAL SQLSTATE '45000'
298     SET MESSAGE_TEXT = 'Error: This organisation is not a Research Center';
299     END IF;
300     END$
```

```

302 ● CREATE TRIGGER res_type_update BEFORE UPDATE ON research_center
303     FOR EACH ROW
304     BEGIN
305     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
306     <> NEW.organisation_type THEN
307     SIGNAL SQLSTATE '45000'
308         SET MESSAGE_TEXT = 'Error: This organisation is not a Research Center';
309     END IF;
310     END$

312 ● CREATE TRIGGER uni_type_insert BEFORE INSERT ON university
313     FOR EACH ROW
314     BEGIN
315     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
316     <> NEW.organisation_type THEN
317     SIGNAL SQLSTATE '45000'
318         SET MESSAGE_TEXT = 'Error: This organisation is not a University';
319     END IF;
320     END$

322 ● CREATE TRIGGER uni_type_update BEFORE UPDATE ON university
323     FOR EACH ROW
324     BEGIN
325     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
326     <> NEW.organisation_type THEN
327     SIGNAL SQLSTATE '45000'
328         SET MESSAGE_TEXT = 'Error: This organisation is not a University';
329     END IF;
330     END$

332 ● CREATE TRIGGER firm_type_insert BEFORE INSERT ON firm
333     FOR EACH ROW
334     BEGIN
335     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
336     <> NEW.organisation_type THEN
337     SIGNAL SQLSTATE '45000'
338         SET MESSAGE_TEXT = 'Error: This organisation is not a Firm';
339     END IF;
340     END$

342 ● CREATE TRIGGER firm_type_update BEFORE UPDATE ON firm
343     FOR EACH ROW
344     BEGIN
345     IF (SELECT organisation_type FROM organisation o WHERE o.organisation_id = NEW.organisation_id)
346     <> NEW.organisation_type THEN
347     SIGNAL SQLSTATE '45000'
348         SET MESSAGE_TEXT = 'Error: This organisation is not a Firm';
349     END IF;
350     END$

```

Αυτά τα triggers εξασφαλίζουν ότι δεν θα εισαχθεί ποτέ στα tables university, research_center, firm ένας οργανισμός που δεν είναι της ίδιας κατηγορίας. Στο δικό μας UI ωστόσο είναι περιττά, αφού εισαγωγή σε αυτά τα tables γίνεται μόνο μέσω της εισαγωγής στο table organisation και κατά την μεταφορά των δεδομένων από το organisation στα tables-υποκατηγορίες του, ελέγχεται η κατηγορία στην οποία ανήκει ο οργανισμός. Τους αφήσαμε ωστόσο για λόγους διατήρησης της ακεραιότητας της βάσης μας.

18o-19o Trigger:

```
352 • CREATE TRIGGER deliverable_date_in BEFORE INSERT ON deliverable
353     FOR EACH ROW
354     BEGIN
355     IF ((SELECT start_date FROM project p WHERE p.project_id = NEW.project_id) > NEW.delivery_date)
356     OR ((SELECT end_date FROM project p WHERE p.project_id=NEW.project_id) < NEW.delivery_date) THEN
357     SIGNAL SQLSTATE '45000'
358     SET MESSAGE_TEXT = "Error: A project's deliverable should have a delivery date
359     before the beginning and after the end of the project";
360     END IF;
361     END$

363 • CREATE TRIGGER deliverable_date_up BEFORE UPDATE ON deliverable
364     FOR EACH ROW
365     BEGIN
366     IF ((SELECT start_date FROM project p WHERE p.project_id = NEW.project_id) > NEW.delivery_date)
367     OR ((SELECT end_date FROM project p WHERE p.project_id=NEW.project_id) < NEW.delivery_date) THEN
368     SIGNAL SQLSTATE '45000'
369     SET MESSAGE_TEXT = "Error: A project's deliverable should have a delivery date
370     before the beginning and after the end of the project";
371     END IF;
372     END$
```

Κάνουμε την παραδοχή ότι τα παραδοτέα παραδίδονται κατά την διάρκεια διεξαγωγής του κάθε έργου. Αυτά τα triggers εξασφαλίζουν ότι ένα παραδοτέο θα έχει ημερομηνία παράδοσης μεταξύ της ημερομηνίας έναρξης και της ημερομηνίας λήξης του έργου στο οποίο ανήκει.

20o-21o Triggers:

```
372 • CREATE TRIGGER birth_in BEFORE INSERT ON researcher
373     FOR EACH ROW
374     BEGIN
375     IF (DATEDIFF(CURDATE(), NEW.date_of_birth) < 18*365.25) THEN
376     SIGNAL SQLSTATE '45000'
377     SET MESSAGE_TEXT = "Error: Can not have a researcher that is younger than 18 years old";
378     END IF;
379     END$

380
381 • CREATE TRIGGER birth_up BEFORE UPDATE ON researcher
382     FOR EACH ROW
383     BEGIN
384     IF (DATEDIFF(CURDATE(), NEW.date_of_birth) < 18*365.25) THEN
385     SIGNAL SQLSTATE '45000'
386     SET MESSAGE_TEXT = "Error: Can not have a researcher that is younger than 18 years old";
387     END IF;
388     END$
```

Κάνουμε την παραδοχή ότι όλοι οι ερευνητές, για να είναι ερευνητές σε οργανισμούς, πρέπει να είναι τουλάχιστον 18 χρονών. Αυτά τα triggers εξασφαλίζουν ακριβώς αυτό.

22o Trigger:


```

390 • CREATE TRIGGER new_org BEFORE UPDATE ON researcher
391     FOR EACH ROW
392     BEGIN
393     IF (OLD.organisation_id <> NEW.organisation_id
394     AND EXISTS(SELECT i.researcher_id FROM participates_in i INNER JOIN project p
395                 ON i.researcher_id = NEW.researcher_id
396                 AND i.project_id = p.project_id
397                 AND p.end_date > CURDATE())) THEN
398         SIGNAL SQLSTATE '45000'
399         SET MESSAGE_TEXT = "Error: Can not change a researcher's organisation
400                             if he still works on his former organisation's projects";
401     END IF;
402     END$

```

Αυτό το trigger εξασφαλίζει ότι ένας ερευνητής δεν μπορεί να αλλάξει τον οργανισμό στον οποίο δουλεύει, αν στον οργανισμό που βρίσκεται συμμετέχει σε ενεργά έργα. Δηλαδή για να δουλέψει ένας ερευνητής σε κάποιον άλλο οργανισμό, θα πρέπει πρώτα να σταματήσει να δουλεύει στα έργα του. Για τα ανενεργά έργα στα οποία είχε δουλέψει αυτός ο ερευνητής για αυτόν τον οργανισμό, διατηρείται η πληροφορία της συμμετοχής του.

Τα drop:

```

1      -- DROP VIEWS
2 •    DROP VIEW projects_per_organisation;
3 •    DROP VIEW projects_per_researcher;
4
5      -- DROP TABLES
6 •    DROP TABLE deliverable;
7 •    DROP TABLE project;
8 •    DROP TABLE researcher;
9 •    DROP TABLE phone_number;
10 •   DROP TABLE research_center;
11 •   DROP TABLE university;
12 •   DROP TABLE firm;
13 •   DROP TABLE organisation;
14 •   DROP TABLE program;
15 •   DROP TABLE elidek_ex;
16 •   DROP TABLE participates_in;
17 •   DROP TABLE is_about;
18 •   DROP TABLE research_field;

```

INDEXES

Όπως είναι γνωστό, τα indexes είναι αρκετά χρήσιμα για μία βάση, καθώς βελτιώνουν την απόδοση και επισπεύδουν την αναζήτηση και λήψη δεδομένων. Έτσι τα queries υλοποιούνται πιο αποδοτικά. Είναι ιδιαίτερα χρήσιμα όταν ορίζονται σε attributes που διατρέχονται συχνά για βάσεις με μεγάλο όγκο δεδομένων. Τα δικά μας δεδομένα δεν είναι ιδιαίτερα πολλά, αλλά θα ορίσουμε κάποια indexes για τους σκοπούς της άσκησης. Αρχικά ωστόσο να επισημάνουμε ότι index δημιουργείται αυτόματα για τα attributes που είναι primary keys καθώς και τα attributes που είναι unique. Είναι ωστόσο χρήσιμο να ορίσουμε επιπλέον indexes για τα attributes που δεν είναι ούτε primary keys, ούτε unique αλλά

χρησιμοποιούνται συχνά στα queries. Με αυτήν την λογική, καταλήγουμε στην δημιουργία των εξής indexes:

```
189 • CREATE INDEX s_d ON project (start_date);
190 • CREATE INDEX e_d ON project (end_date);
191 • CREATE INDEX r_id ON participates_in (researcher_id);
192 • CREATE INDEX pa_id ON participates_in (project_id);
193 • CREATE INDEX p_n ON is_about (project_id);
```

INSERT

Τα insert είναι ένα μεγάλο αρχείο που βρίσκεται ολόκληρο στο git repo, το λινκ για το οποίο βρίσκεται στην αρχή της αναφοράς μας. Ενδεικτικά παραθέτουμε εδώ κάποια από αυτά:

```
21 • INSERT INTO organisation (organisation_name,city,street_name,street_number,postal_code,
22 organisation_type,budget1) VALUES ("Baxter, Jones and Taylor", "Martinshire", "Randy Pike", "208",
23 "79854", "Firm", "5933489113");
24 • INSERT INTO organisation (organisation_name,city,street_name,street_number,postal_code,
25 organisation_type,budget1) VALUES ("Jones, Palmer and Osborn", "Wardborough", "White Crest", "558",
26 "82596", "Firm", "7378691546");

56 • INSERT INTO phone_number (organisation_id,phone) VALUES ("1", "2101285997");
57 • INSERT INTO phone_number (organisation_id,phone) VALUES ("2", "2107502229");

114 • INSERT INTO researcher (first_name,last_name,sex,date_of_birth,date_hired,organisation_id)
115 VALUES ("Teresa", "Kelly", "female", "1942-12-18", "2020-10-17", "34");
116 • INSERT INTO researcher (first_name,last_name,sex,date_of_birth,date_hired,organisation_id)
117 VALUES ("Bridget", "Bridges", "female", "1968-11-02", "2011-11-10", "12");

1116 • INSERT INTO program (title,elidek_dep) VALUES ("Solar panels", "Energy Dept");
1117 • INSERT INTO program (title,elidek_dep) VALUES ("Green Greece", "Environmental Dept");

1150 • INSERT INTO elidek_ex (first_name,last_name) VALUES ("Donna", "Johnson");
1151 • INSERT INTO elidek_ex (first_name,last_name) VALUES ("Rachel", "Terrell");

1300 • INSERT INTO project (start_date,end_date,fund,project_title,project_description,elidek_ex_id,
1301 program_id,organisation_id,supervisor_id,evaluator_id,eval_grade,eval_date) VALUES ("2006-06-19",
1302 "2009-02-04", "988989", "Orbital clustering magnetosphere", "Something scientific", "95", "27", "13",
1303 "274", "764", "80", "2005-10-16");
1304 • INSERT INTO project (start_date,end_date,fund,project_title,project_description,elidek_ex_id,
1305 program_id,organisation_id,supervisor_id,evaluator_id,eval_grade,eval_date) VALUES ("2012-01-16",
1306 "2013-06-21", "232737", "Orbital agglomerative algorithm", "Something scientific", "50", "21", "17",
1307 "374", "707", "83", "2011-01-30");

1561 • INSERT INTO participates_in (project_id,researcher_id) VALUES ("1", "241");
1562 • INSERT INTO participates_in (project_id,researcher_id) VALUES ("1", "330");

3612 • INSERT INTO deliverable (project_id,title,summary,delivery_date) VALUES ("197", "A nice title",
3613 "A nice summary", "2019-11-12");
3614 • INSERT INTO deliverable (project_id,title,summary,delivery_date) VALUES ("161", "A nice title",
3615 "A nice summary", "2023-01-01");

3730 • INSERT INTO research_field (field_name) VALUES ("Medicine");
3731 • INSERT INTO research_field (field_name) VALUES ("Mathematics");
3748 • INSERT INTO is_about (field_name,project_id) VALUES ("Physics","1");
3749 • INSERT INTO is_about (field_name,project_id) VALUES ("Mathematics","2");
```

QUERIES

Τα queries που ζητούνται από την εκφώνηση και έχουν υλοποιηθεί στην εφαρμογή μας:

```
1  -- QUERIES
2  -- 3.1
3  -- To search for the requested projects
4  •  SELECT p.project_id AS id, p.project_title AS title
5      FROM project p
6      INNER JOIN elidek_ex e
7      ON p.elidek_ex_id = e.elidek_ex_id
8      WHERE 1=1;
9  /* To meet users criteria the following could be added after "1=1":
10     " AND start_date > '$s_date'"
11     " AND end_date < '$e_date'"
12     " AND duration = '$duration'"
13     " AND e.first_name = '$exec[0]' AND e.last_name = '$exec[1]'"
14  */
15
16  -- To find the researchers participating in a project with a certain project_id
17  •  SELECT * FROM researcher r
18      INNER JOIN participates_in p
19      ON p.researcher_id = r.researcher_id
20      WHERE p.project_id = ?
21      ORDER BY r.last_name;
22  -- questionmark (?) gets binded to the id of the project that the user requested
23
24  -- 3.2
25  -- To create the 1st view
26  •  CREATE VIEW projects_per_researcher AS
27      SELECT r.researcher_id, r.first_name, r.last_name, p.project_id, p.project_title
28      FROM researcher r
29      INNER JOIN participates_in i
30      ON r.researcher_id = i.researcher_id
31      INNER JOIN project p
32      ON p.project_id = i.project_id
33      GROUP BY r.researcher_id, p.project_id
34      ORDER BY r.researcher_id, p.project_id;
35
36  -- To select attributes from the 1st view
37  •  SELECT ID, first_name, last_name, project_id, project_title
38      FROM projects_per_researcher;
39
40  -- To create 2nd the view
41  •  CREATE VIEW projects_per_organisation AS
42      SELECT o.organisation_id, o.organisation_name, p.project_id, p.project_title
43      FROM project p
44      INNER JOIN organisation o
```

```

45         ON p.organisation_id = o.organisation_id
46         GROUP BY o.organisation_id, p.project_id;
47
48     -- To select attributes from the 2nd view
49 • SELECT organisation_id, organisation_name, project_id, project_title
50     FROM projects_per_organisation;
51
52     -- 3.3
53 • SELECT p.project_title AS title, p.project_id AS p_id
54     FROM project p
55     INNER JOIN is_about i
56     ON p.project_id = i.project_id
57     WHERE field_name = '$name' AND p.end_date > CURDATE()
58     ORDER BY title;
59     -- '$name' is defined by user input
60
61     -- 3.4
62 • WITH proj_per_year AS (SELECT o.organisation_id AS ID, YEAR(p.start_date) AS xronos, COUNT(*) AS N
63     FROM organisation o, project p
64     WHERE o.organisation_id = p.organisation_id
65     GROUP BY o.organisation_name, YEAR(p.start_date))
66     SELECT o.organisation_id AS o_id, o.organisation_name AS o_name
67     FROM organisation o, proj_per_year y1, proj_per_year y2
68     WHERE (o.organisation_id = y1.ID
69     AND o.organisation_id = y2.ID
70         AND y1.xronos = y2.xronos-1
71         AND y1.N = y2.N
72         AND y1.N >= 10);
73
74     -- 3.5
75 • SELECT x.field_name as first_field, i.field_name as second_field, COUNT(*) as quantity
76     FROM is_about x
77     INNER JOIN is_about i
78     ON i.project_id = x.project_id
79     WHERE i.field_name > x.field_name
80     GROUP BY x.field_name, i.field_name
81     ORDER BY COUNT(*) DESC LIMIT 3;
82
83     -- 3.6
84 • WITH max_proj(N, ID) AS (select count(*) AS N, i.researcher_id AS ID
85     FROM participates_in i, project p
86     WHERE i.project_id = p.project_id
87     AND p.end_date >= CURDATE()
88     GROUP BY i.researcher_id)
89     SELECT r.first_name as f_name, r.last_name as l_name, (SELECT max(N) from max_proj) AS number_of_projects
90     FROM researcher r

```

```

91 WHERE (DATEDIFF(CURDATE(), r.date_of_birth) < 40*365.25)
92 AND (SELECT m.N FROM max_proj m WHERE m.ID = r.researcher_id) = (SELECT max(N) FROM max_proj);
93
94 -- 3.7
95 • SELECT org.organisation_name as o_name, SUM(p.fund) as funding, e.first_name as f_name,
96      e.last_name as l_name, COUNT(p.fund) as amount
97 FROM project p
98 INNER JOIN elidek_ex e
99 ON p.elidek_ex_id = e.elidek_ex_id
100 INNER JOIN organisation org
101 ON p.organisation_id = org.organisation_id
102 WHERE org.organisation_type = 'Firm'
103 GROUP BY e.elidek_ex_id, org.organisation_id
104 ORDER BY SUM(p.fund) DESC
105 LIMIT 5;
106
107 -- 3.8
108 • SELECT r.first_name as f_name, r.last_name as l_name, COUNT(*) as amount
109 FROM researcher r
110 INNER JOIN participates_in w
111 ON r.researcher_id = w.researcher_id
112 INNER JOIN project p
113 ON p.project_id = w.project_id
114 WHERE NOT EXISTS (SELECT * FROM deliverable d WHERE p.project_id = d.project_id)
115      AND DATEDIFF(CURRENT_DATE(), p.end_date) < 0
116 GROUP BY r.researcher_id
117 HAVING COUNT(*) > 4
118 ORDER BY COUNT(*) DESC;

```

DELETE

Τα delete είναι τα παρακάτω:

```

1  -- DELETE
2 • DELETE FROM project;
3 • DELETE FROM program;
4 • DELETE FROM organisation;
5 • DELETE FROM elidek_ex;
6 • DELETE FROM research_field;
7 • DELETE FROM researcher;
8 • DELETE FROM phone_number;
9 • DELETE FROM university;
10 • DELETE FROM research_center;
11 • DELETE FROM firm;
12 • DELETE FROM participates_in;
13 • DELETE FROM deliverable;
14 • DELETE FROM is_about;

```

Εγκατάσταση της εφαρμογής

Για την υλοποίηση χρησιμοποιήθηκε το πακέτο του XAMPP (MySQL, Apache) καθώς και το MySQL Workbench. Για την εγκατάσταση της βάσης ακολουθούνται τα εξής βήματα (τα αρχεία SQL βρίσκονται στον φάκελο SQLfiles του gitrepo):

1. Τρέχουμε το αρχείο elidek_schema.sql
2. Στη συνέχεια, εισάγουμε στη βάση τα δοκιμαστικά δεδομένα τρέχοντας το αρχείο insertdata.sql

Για το user interface χρησιμοποιήθηκαν οι PHP και HTML με ορισμένα τμήματα κώδικα CSS για τη μορφοποίηση. Για την εγκατάσταση του U.I. πρέπει να κατέβει ο φάκελος elidek που περιέχει όλα τα αρχεία .php και να φορτωθεί μέσα στον φάκελο htdocs, οποίος βρίσκεται εντός του φακέλου xampp που δημιουργείται κατά την εγκατάσταση του xampp σε προεπιλεγμένη από τον χρήστη τοποθεσία. Η σύνδεση της βάσης με το web app γίνεται στο αρχείο config.php το οποίο καλείται και σε όλα τα υπόλοιπα.

Αφού γίνουν τα παραπάνω βήματα σωστά, το U.I. θα εμφανίζεται στη διεύθυνση http://localhost/elidek/index_project.php (Η αρχική σελίδα για τα projects του ΕΛΙΔΕΚ, ενδεικτικά, καθώς το U.I. δεν έχει αρχική σελίδα, αλλά navigation bar όπως αναλύεται παρακάτω).

Σχετικά με το U.I.

Το U.I. υποστηρίζει λειτουργίες CRUD για όλα τα tables. Επίσης, περιέχει υλοποιήσεις όλων των ερωτημάτων του ζητήματος 3 της εργασίας που εμφανίζονται μετά από κλικ στην ετικέτα «Queries» πάνω αριστερά. Βασικό τρόπο «μεταπήδησης» ανάμεσα στις διάφορες σελίδες του web application αποτελεί η navigation bar που βρίσκεται στην κορυφή. Στην παρακάτω εικόνα παρατίθεται ενδεικτικά στιγμιότυπο από το κεντρικό menu των προγραμμάτων (Programs):

Program Details

+ Add New Record

Queries

Projects

Organisations

Researchers

Programs

Research Fields



















ELIDEK executives

Project Deliverables

Organisation Phones

Project-Field

Researcher-Project

#	Title	Department	
1	Bike city	Enviromental Dept	  
2	Graves experiment	Mathematics Dept	  
3	Solar panels	Energy Dept	  
4	Green Greece	Environmental Dept	  
5	Lava manipulation	Physics Dept	  
6	Faster blood examinations	Medical Dept	  

Όλες οι καρτέλες του navigation οδηγούν σε αντίστοιχες σελίδες με τα menu των διαφόρων tables της βάσης (εκτός από τα “Queries”, “Organisations”). Σε καθεμιά από αυτές τις σελίδες φαίνεται αρχικά ένας πίνακας με τα βασικά στοιχεία των tables της βάσης. Με το

πράσινο κουμπί «Add New record» πάνω δεξιά προστίθεται καινούρια εγγραφή στο εκάστοτε table της βάσης δεδομένων. Με τα τρία κουμπιά στη δεξιότερη στήλη του πίνακα ο χρήστης μπορεί να δει περισσότερες πληροφορίες για την κάθε εγγραφή, να την επεξεργαστεί ή να την διαγράψει.

Η καρτέλα Organisations πρωτού παραπέμπει στο menu των οργανισμών, αφήνει των χρήστη να επιλέξει αν επιθυμεί να μεταβεί στο menu των εταιρειών, των πανεπιστημίων ή των ερευνητικών κέντρων. Υπάρχει και επιλογή λίστας όλων των οργανισμών όπου όμως δεν δίνονται στον χρήστη δυνατότητες επεξεργασίας και διαγραφής.

The screenshot shows the 'Choose type' form. At the top is a navigation bar with tabs: Queries (highlighted in green), Projects, Organisations, Researchers, Programs, Research Fields, ELIDEK executives, Project Deliverables, and Organisation Phones. Below the navigation bar is a sub-header with 'Project-Field' and 'Researcher-Project'. The main content area has the title 'Choose type' and the instruction 'Please select the type of the organisation.' Below this are four radio button options: 'Firm', 'University', 'Research Center', and 'View all (only for viewing)'. A blue 'Submit' button is at the bottom left.

Η καρτέλα Queries του navigation, παραπέμπει σε radio button με τις επιλογές των 8 queries της εκφώνησης. Αφού ο χρήστης επιλέξει ποιο query επιθυμεί να εκτελέσει, τον παραπέμπει σε σχετική σελίδα.

The screenshot shows the 'Choose query' form. It has the same navigation bar and sub-header as the previous form. The main content area has the title 'Choose query' and the instruction 'Please select the query you want to execute.' Below this are five radio button options: 'Query 3.1', 'Query 3.2', 'Query 3.3', 'Query 3.4', and 'Query 3.5'. A vertical scrollbar is visible on the right side of the form.

(Οι υπόλοιπες επιλογές είναι εμφανείς με scrolling down)

Όλα τα αρχεία με τον κώδικα σε php/html και sql βρίσκονται ανεβασμένα στο gitrepo: <https://github.com/ChGeorgiou/dbclass2022.git>