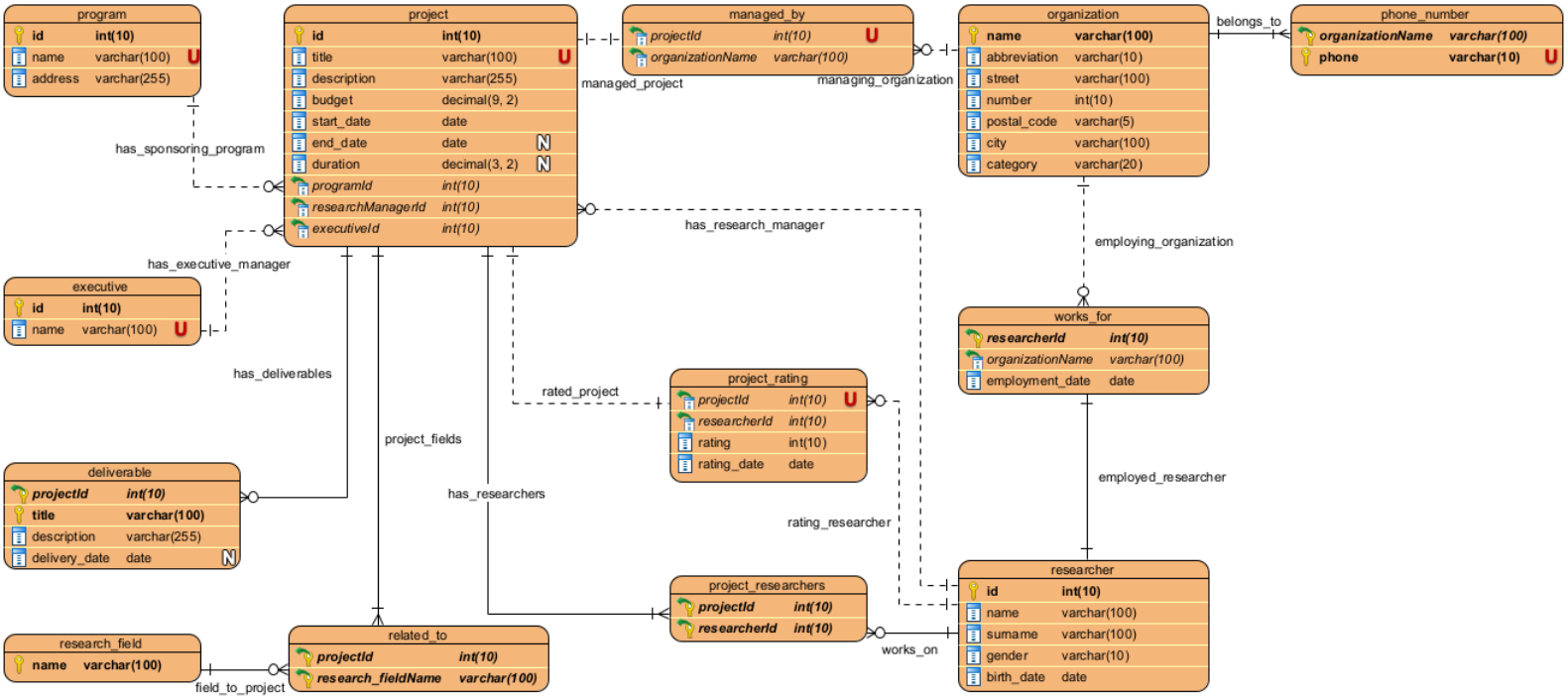
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ 2022

Κωνσταντίνος Διβριώτης
03114140
Ομάδα Project 19

Σχεσιακό Διάγραμμα της Βάσης Δεδομένων



1. Στελέχη που δουλεύουν για τον ΕΛ.ΙΔ.Ε.Κ. (executive)

```
CREATE TABLE `executive` (
  `id` int NOT NULL AUTO_INCREMENT COMMENT 'Μοναδικό αναγνωριστικό
στελέχους',
  `name` varchar(100) NOT NULL COMMENT 'Όνομα στελέχους (μοναδικό)',
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE = InnoDB AUTO_INCREMENT = 16 DEFAULT CHARSET = utf8mb4 COLLATE =
utf8mb4_0900_ai_ci COMMENT = 'Στελέχη ΕΛ.ΙΔ.Ε.Κ.';
```

Για κάθε στέλεχος υπάρχει ένα μοναδικό αναγνωριστικό (**id**) το οποίο είναι ένας αύξον αριθμός, ο οποίος παράγεται αυτόματα από τη βάση, και ένα πεδίο για το όνομα του στελέχους (**name**) (*μέχρι 100 χαρακτήρες*), το οποίο θεωρούμε ότι είναι μοναδικό.

2. **Προγράμματα που υλοποιούνται από τον ΕΛ.ΙΔ.Ε.Κ. και χρηματοδοτούν έργα (program)**

```
CREATE TABLE `program` (
  `id` int NOT NULL AUTO_INCREMENT COMMENT 'Μοναδικό αναγνωριστικό του
προγράμματος',
  `name` varchar(100) NOT NULL COMMENT 'Όνομα προγράμματος',
  `address` varchar(255) NOT NULL COMMENT 'Διεύθυνση',
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE = InnoDB AUTO_INCREMENT = 41 DEFAULT CHARSET = utf8mb4 COLLATE =
utf8mb4_0900_ai_ci COMMENT = 'Προγράμματα που χρηματοδοτούν έργα';
```

Για κάθε πρόγραμμα υπάρχει ένα μοναδικό αναγνωριστικό (**id**) το οποίο είναι ένας αύξον αριθμός, ο οποίος παράγεται αυτόματα από τη βάση, ένα πεδίο για το όνομα του προγράμματος (**name**) (*μέχρι 100 χαρακτήρες*), το οποίο θεωρούμε ότι είναι μοναδικό και ένα πεδίο για τη διεύθυνση του ΕΛ.ΙΔ.Ε.Κ. στην οποία ανήκει (**address**) (*μέχρι 255 χαρακτήρες*).

3. **Οργανισμοί που διαχειρίζονται έργα (organization)**

```
CREATE TABLE `organization` (
  `name` varchar(100) COMMENT 'Μοναδικό όνομα του οργανισμού',
  `abbreviation` varchar(10) NOT NULL COMMENT 'Συντομογραφία',
  `street` varchar(100) NOT NULL COMMENT 'Διεύθυνση οργανισμού: Οδός',
  `number` int NOT NULL COMMENT 'Διεύθυνση οργανισμού: Αριθμός',
  `postal_code` varchar(5) NOT NULL COMMENT 'Διεύθυνση οργανισμού: Τ.Κ.',
  `city` varchar(100) NOT NULL COMMENT 'Διεύθυνση οργανισμού: Πόλη',
  `category` varchar(20) NOT NULL COMMENT 'Κατηγορία: Ερευνητικό
Κέντρο(Research Center), Πανεπιστήμιο (University) ή Εταιρεία (Company)',
  PRIMARY KEY (`name`),
  CONSTRAINT `check_category` CHECK (
    (
      `category` in (
        _utf8mb4 'University',
        _utf8mb4 'Company',
        _utf8mb4 'Research Center'
      )
    )
  )
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci
COMMENT = 'Οργανισμός';
```

Για κάθε οργανισμό υπάρχει ένα πεδίο για το όνομα του οργανισμού (**name**) (*μέχρι 100 χαρακτήρες*), το οποίο θεωρούμε ότι είναι μοναδικό και συνεπώς χρησιμοποιείται ως κύριο κλειδί (*primary key*), ένα πεδίο για τη συντομογραφία (**abbreviation**) (*μέχρι 10 χαρακτήρες*). Επίσης υπάρχουν πληροφορίες για τη διεύθυνση του οργανισμού, η οποία περιλαμβάνει την οδό (**street** - *μέχρι 100 χαρακτήρες*), τον αριθμό (**number**), τον ταχυδρομικό κώδικα (**postal_code**) και την πόλη (**city**). Ακόμη υπάρχει ένα πεδίο που δείχνει σε ποια κατηγορία ανήκει ο οργανισμός (**category**), το οποίο παίρνει τιμές **University**, **Company** ή **Research Center**, ανάλογα αν ο οργανισμός είναι Πανεπιστήμιο, Εταιρεία ή Ερευνητικό Κέντρο αντίστοιχα. Ο έλεγχος εγκυρότητας του πεδίου γίνεται με τη βοήθεια του *CHECK CONSTRAINT* **check_category**.

Τέλος, κάθε οργανισμός έχει ένα ή περισσότερα τηλέφωνα επικοινωνίας, τα οποία αποθηκεύονται με τη βοήθεια ενός ακόμη table με όνομα **phone_number**, στο οποίο υπάρχουν εγγραφές για κάθε οργανισμό (**organizationName** - *FOREIGN KEY* στο πεδίο **name** του πίνακα ***organization***) μαζί με έναν αριθμό τηλεφώνου (**phone**) ο οποίος ανήκει στον οργανισμό. Θεωρούμε ότι κάθε αριθμός είναι μοναδικός, συνεπώς ορίζουμε ως κύριο κλειδί τον οργανισμό μαζί με τον αριθμό.
Επίσης ελέγχουμε την εγκυρότητα του αριθμού (**phone**) με τη βοήθεια του *CHECK CONSTRAINT* **check_phone** για να σιγουρευτούμε ότι έχει μήκος 10 χαρακτήρες και αποτελείται αποκλειστικά από ψηφία.

```
CREATE TABLE `phone_number` (
  `organizationName` varchar(100) NOT NULL COMMENT 'Όνομα του οργανισμού στον οποίο ανήκει ο αριθμός',
  `phone` varchar(10) NOT NULL COMMENT 'Αριθμός τηλεφώνου του οργανισμού (μοναδικός) - Ακριβώς 10 χαρακτήρες',
  PRIMARY KEY (`organizationName`, `phone`),
  UNIQUE KEY `phone_UNIQUE` (`phone`),
  CONSTRAINT `belongs_to` FOREIGN KEY (`organizationName`) REFERENCES `organization` (`name`),
  CONSTRAINT `check_phone` CHECK (regexp_like(`phone`,_utf8mb4'^[0-9]{10}$'))
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Τηλεφωνικοί αριθμοί (τουλάχιστον 1 για κάθε οργανισμό)';
```

4. **Ερευνητές που εργάζονται σε έργα (researcher)**

```
CREATE TABLE `researcher` (
  `id` int NOT NULL AUTO_INCREMENT COMMENT 'Μοναδικό αναγνωριστικό του ερευνητή',
  `name` varchar(100) NOT NULL COMMENT 'Όνομα',
  `surname` varchar(100) NOT NULL COMMENT 'Επίθετο',
  `gender` varchar(10) NOT NULL COMMENT 'Φύλο (male ή female)',
  `birth_date` date NOT NULL COMMENT 'Ημερομηνία Γέννησης',
  PRIMARY KEY (`id`),
  CONSTRAINT `check_gender` CHECK ((`gender` in (_utf8mb4 'male',_utf8mb4 'female')))
) ENGINE = InnoDB AUTO_INCREMENT = 111 DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Ερευνητές που εργάζονται σε έργα';
```

Για κάθε ερευνητή υπάρχει ένα μοναδικό αναγνωριστικό (**id**) το οποίο είναι ένας αύξον αριθμός, ο οποίος παράγεται αυτόματα από τη βάση. Επίσης καταχωρίζονται οι απαιτούμενες πληροφορίες οι οποίες είναι το όνομα (**name**) (*μέχρι 100 χαρακτήρες*), το επίθετο (**surname**) (*μέχρι 100 χαρακτήρες*), το φύλο (**gender**) και η ημερομηνία γέννησης (**birth_date**).
Ο έλεγχος εγκυρότητας του πεδίου **gender** γίνεται με τη βοήθεια του *CHECK CONSTRAINT* **check_gender**, ώστε να βεβαιωθούμε ότι παίρνει μόνο τις τιμές *male* ή *female*.

Κάθε ερευνητής εργάζεται σε έναν (και μόνο) οργανισμό από κάποια ημερομηνία. Η υπαλληλική σχέση μεταξύ κάθε ερευνητή/οργανισμού αποθηκεύεται στο table **works_for**, στο οποίο υπάρχουν εγγραφές για κάθε ερευνητή (**researcherId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***researcher***) μαζί με το όνομα του οργανισμού στον οποίο εργάζεται (**organizationName** - *FOREIGN KEY* στο πεδίο **name** του πίνακα ***organization***) και την ημερομηνία έναρξης της εργασίας

(**employment_date**). Εφόσον κάθε ερευνητής μπορεί να εργάζεται σε έναν μόνο οργανισμό, ορίζουμε ως κύριο κλειδί το αναγνωριστικό του ερευνητή.

```
CREATE TABLE `works_for` (
  `researcherId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του εργαζόμενου ερευνητή',
  `organizationName` varchar(100) NOT NULL COMMENT 'Όνομα του οργανισμού στον οποίο εργάζεται',
  `employment_date` date NOT NULL COMMENT 'Ημερομηνία έναρξης της εργασίας',
  PRIMARY KEY (`researcherId`),
  KEY `employing_organization` (`organizationName`),
  CONSTRAINT `employed_researcher` FOREIGN KEY (`researcherId`) REFERENCES `researcher` (`id`),
  CONSTRAINT `employing_organization` FOREIGN KEY (`organizationName`) REFERENCES `organization` (`name`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Υπαλληλική σχέση ερευνητή/οργανισμού';
```

5. Επιστημονικά πεδία που περιγράφουν θεματικές περιοχές των έργων (research_field)

```
CREATE TABLE `research_field` (
  `name` varchar(100) NOT NULL COMMENT 'Όνομα (μοναδικό) του επιστημονικού πεδίου',
  PRIMARY KEY (`name`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Επιστημονικά πεδία';
```

Για κάθε επιστημονικό πεδίο (τομέα έρευνας) υπάρχει το όνομά του (**name**) (*μέχρι 100 χαρακτήρες*), το οποίο θεωρούμε ότι είναι μοναδικό.

6. Έργα/επιχορηγήσεις (project)

```
CREATE TABLE `project` (
  `id` int NOT NULL AUTO_INCREMENT COMMENT 'Μοναδικό αναγνωριστικό του έργου',
  `title` varchar(100) NOT NULL COMMENT 'Τίτλος του έργου',
  `description` varchar(255) NOT NULL COMMENT 'Περίληψη',
  `budget` decimal(9, 2) NOT NULL COMMENT 'Ποσό χρηματοδότησης',
  `start_date` date NOT NULL COMMENT 'Έναρξη',
  `end_date` date DEFAULT NULL COMMENT 'Λήξη (αν είναι κενό, το έργο είναι ακόμα ενεργό)',
  `duration` decimal(3, 2) GENERATED ALWAYS AS (
    (
      (to_days(`end_date`) - to_days(`start_date`)) / 365
    )
  ) STORED COMMENT 'Διάρκεια του έργου (υπολογιζόμενο πεδίο)',
  `programId` int NOT NULL COMMENT 'Αναγνωριστικό του προγράμματος που χρηματοδοτεί το έργο',
  `researchManagerId` int NOT NULL COMMENT 'Επιστημονικός υπεύθυνος του έργου',
```

```
  `executiveId` int NOT NULL COMMENT 'Στέλεχος του ΕΛ.ΙΔ.Ε.Κ που
διαχειρίζεται το έργο',
  PRIMARY KEY (`id`),
  UNIQUE KEY `title_UNIQUE` (`title`),
  KEY `has_research_manager` (`researchManagerId`),
  KEY `has_executive_manager` (`executiveId`),
  KEY `has_sponsoring_program` (`programId`),
  CONSTRAINT `has_research_manager` FOREIGN KEY (`researchManagerId`)
REFERENCES `researcher` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `has_executive_manager ` FOREIGN KEY (`executiveId`)
REFERENCES ` has_executive_manager ` (`id`) ON UPDATE CASCADE,
  CONSTRAINT ` has_sponsoring_program ` FOREIGN KEY (`programId`)
REFERENCES `program` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `limit_end_date` CHECK (
    (
      (
        (to_days(`end_date`) - to_days(`start_date`)) / 365
      ) between 1
      and 4
    )
  )
) ENGINE = InnoDB AUTO_INCREMENT = 121 DEFAULT CHARSET = utf8mb4 COLLATE
= utf8mb4_0900_ai_ci COMMENT = 'Εργα/Επιχορηγήσεις';
```

Για κάθε έργο/επιχορήγηση υπάρχει ένα μοναδικό αναγνωριστικό (**id**) το οποίο είναι ένας αύξον αριθμός, ο οποίος παράγεται αυτόματα από τη βάση. Επίσης υπάρχουν τα σχετικά δεδομένα του, δηλαδή ο τίτλος (**title**) (*μέχρι 100 χαρακτήρες*), η περίληψη (**description**) (*μέχρι 255 χαρακτήρες*), το ποσό χρηματοδότησης/επιχορήγησης (**budget**), η ημερομηνία έναρξης (**start_date**) και λήξης (**end_date**) και η διάρκεια του έργου σε χρόνια (**duration**).
- Η εγκυρότητα της ημερομηνίας λήξης ελέγχεται με τη βοήθεια του *CHECK CONSTRAINT* **limit_end_date**, ώστε να βεβαιωθούμε ότι η ελάχιστη διάρκεια από την ημερομηνία έναρξης είναι **1 έτος** και η μέγιστη διάρκεια τα **4 έτη**. Η ημερομηνία λήξης μπορεί να είναι κενή (**NULL**), σε περίπτωση που το έργο είναι ακόμα ενεργό.
- Η διάρκεια του έργου σε έτη είναι υπολογιζόμενο πεδίο (**GENERATED**), το οποίο υπολογίζεται ως η διαφορά των πεδίων **start_date** και **end_date** σε έτη. Εάν το έργο είναι ακόμη ενεργό, τότε η διάρκεια θα είναι κενή (**NULL**).

Το κάθε έργο έχει έναν ερευνητή που είναι ο επιστημονικός υπεύθυνος του έργου (**researchManagerId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***researcher*** - **CONSTRAINT `has_research_manager`**), ένα πρόγραμμα το οποίο έχει χορηγήσει τη χρηματοδότηση (**programId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***program*** - **CONSTRAINT `has_sponsoring_program`**) και ένα στέλεχος το οποίο το διαχειρίζεται (**executiveId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***executive*** - **CONSTRAINT `has_executive_manager`**).

Επίσης, κάθε έργο έχει έναν οργανισμό που το διαχειρίζεται. Η σχέση μεταξύ κάθε έργου/οργανισμού αποθηκεύεται στο table **managed_by**, στο οποίο υπάρχουν εγγραφές για κάθε έργο (**projectId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***project***) μαζί με το όνομα του οργανισμού το οποίο το διαχειρίζεται (**organizationName** - *FOREIGN KEY* στο πεδίο **name** του πίνακα ***organization***). Εφόσον κάθε έργο διαχειρίζεται από έναν μόνο οργανισμό, ορίζουμε πεδίο **projectId** ως μοναδικό (**UNIQUE**).

```
CREATE TABLE `managed_by` (
  `projectId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του έργου το οποίο συμμετέχει στη σχέση',
  `organizationName` varchar(100) NOT NULL COMMENT 'Όνομα του οργανισμού που διαχειρίζεται το έργο',
  UNIQUE KEY `projectId_UNIQUE` (`projectId`),
  KEY `managing_organization` (`organizationName`),
  CONSTRAINT `managed_project` FOREIGN KEY (`projectId`) REFERENCES `project` (`id`),
  CONSTRAINT `managing_organization` FOREIGN KEY (`organizationName`) REFERENCES `organization` (`name`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Σχέση διαχείρισης μεταξύ έργου και οργανισμού';
```

Κάθε έργο έχει αφορά ένα ή περισσότερα επιστημονικά πεδία. Η σχέση μεταξύ κάθε έργου/σχετιζόμενων επιστημονικών πεδίων αποθηκεύεται στο table **related_to**, στο οποίο υπάρχουν εγγραφές για κάθε έργο (**projectId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***project***) μαζί με το όνομα του σχετιζόμενου επιστημονικού πεδίου (**research_fieldName** - *FOREIGN KEY* στο πεδίο **name** του πίνακα ***research_field***). Ορίζουμε ως κύριο κλειδί το **projectId** μαζί με το **research_fieldName** για να αποφύγουμε διπλές εγγραφές για κάποιο έργο με το ίδιο επιστημονικό πεδίο.

```
CREATE TABLE `related_to` (
  `projectId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του έργου',
  `research_fieldName` varchar(100) NOT NULL COMMENT 'Όνομα του επιστημονικού πεδίου με το οποίο συνδέεται',
  PRIMARY KEY (`projectId`, `research_fieldName`),
  KEY `field_to_project` (`research_fieldName`),
  CONSTRAINT `field_to_project` FOREIGN KEY (`research_fieldName`) REFERENCES `research_field` (`name`),
  CONSTRAINT `project_fields` FOREIGN KEY (`projectId`) REFERENCES `project` (`id`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci COMMENT = 'Σύνδεση έργου με επιστημονικά πεδία';
```

Ένα έργο ενδέχεται να έχει παραδοτέα, τα οποία παραδίδονται σε συγκεκριμένη ημερομηνία. Η σχέση μεταξύ κάθε έργου και των παραδοτέων του (εφόσον υπάρχουν) αποθηκεύεται στο table **deliverable**, στο οποίο υπάρχουν εγγραφές για έργα (**projectId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***project***) μαζί με το τις πληροφορίες για κάθε παραδοτέο του. Οι πληροφορίες για κάθε παραδοτέο είναι ο τίτλος του (**title**) (*μέχρι 100 χαρακτήρες*), η περίληψη (**description**) (*μέχρι 255 χαρακτήρες*) και η ημερομηνία παράδοσης (**delivery_date**). Ορίζουμε ως κύριο κλειδί το **projectId** μαζί με το **title**, θεωρώντας ότι κάθε έργο δε μπορεί να έχει πάνω από ένα παραδοτέο με τον ίδιο τίτλο.

```
CREATE TABLE `deliverable` (

  `projectId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του έργου το οποίο αφορά το παραδοτέο',

  `title` varchar(100) NOT NULL COMMENT 'Τίτλος παραδοτέου',

  `description` varchar(255) NOT NULL COMMENT 'Περιγραφή παραδοτέου',
```

```
  `delivery_date` date DEFAULT NULL COMMENT 'Ημερομηνία παράδοσης',
  PRIMARY KEY (`title`, `projectId`),
  KEY `has_deliverables` (`projectId`),
  CONSTRAINT `has_deliverables` FOREIGN KEY (`projectId`) REFERENCES
`project` (`id`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci
COMMENT = 'Παραδοτέα έργων';
```

Για κάθε έργο υπάρχουν ερευνητές που εργάζονται πάνω σε αυτό. Η σχέση μεταξύ κάθε έργου και των ερευνητών που εργάζονται σε αυτό αποθηκεύεται στο table **project_researchers**, στο οποίο υπάρχουν εγγραφές για κάθε έργο (**projectId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***project***) και τους ερευνητές που εργάζονται σε αυτό (**researcherId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***researcher***). Ορίζουμε ως κύριο κλειδί το **projectId** μαζί με το **researcherId**, για να αποφύγουμε διπλές εγγραφές για κάποιο έργο με τον ίδιο ερευνητή.

```
CREATE TABLE `project_researchers` (
  `projectId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του έργου',
  `researcherId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του
ερευνητή που εργάζεται στο έργο',
  PRIMARY KEY (`projectId`, `researcherId`),
  KEY `works_on` (`researcherId`)
  /*!80000 INVISIBLE */
,
  CONSTRAINT `has_researchers` FOREIGN KEY (`projectId`) REFERENCES
`project` (`id`),
  CONSTRAINT `works_on` FOREIGN KEY (`researcherId`) REFERENCES
`researcher` (`id`)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci
COMMENT = 'Ερευνητές σε κάθε έργο';
```

Τέλος το κάθε έργο, προκειμένου να χρηματοδοτηθεί, έχει αξιολογηθεί από έναν ερευνητή ο οποίος δεν ανήκει στο δυναμικό του οργανισμού που συμμετέχει στην πρόταση. Η σχέση μεταξύ κάθε έργου και της αξιολόγησής του αποθηκεύεται στο table **project_rating**, στο οποίο υπάρχουν εγγραφές για κάθε έργο (**projectId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***project***), τον ερευνητή που έκανε την αξιολόγηση (**researcherId** - *FOREIGN KEY* στο πεδίο **id** του πίνακα ***researcher***) και τις πληροφορίες για την αξιολόγηση. Οι πληροφορίες για κάθε αξιολόγηση είναι ο βαθμός της (**rating**) και η ημερομηνία αξιολόγησης (**rating_date**). Εφόσον κάθε έργο μπορεί να έχει μόνο μία αξιολόγηση, ορίζουμε πεδίο **projectId** ως μοναδικό (**UNIQUE**).
- Η εγκυρότητα του βαθμού αξιολόγησης ελέγχεται με τη βοήθεια του *CHECK CONSTRAINT* **check_rating**, ώστε να βεβαιωθούμε ότι παίρνει τιμές μεταξύ 1 και 10.

```
CREATE TABLE `project_rating` (
  `projectId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του έργου που
αξιολογήθηκε (κάθε έργο μπορεί να αξιολογηθεί μόνο μία φορά)',
  `researcherId` int NOT NULL COMMENT 'Μοναδικό αναγνωριστικό του
ερευνητή που αξιολόγησε το έργο',
  `rating` int NOT NULL COMMENT 'Αξιολόγηση (1-10)',
```

```
  `rating_date` date NOT NULL COMMENT 'Ημερομηνία αξιολόγησης',
  UNIQUE KEY `projectId_UNIQUE` (`projectId`),
  KEY `rated_project` (`projectId`),
  KEY `rating_researcher` (`researcherId`),
  CONSTRAINT `project_rating_ibfk_1` FOREIGN KEY (`projectId`) REFERENCES
`project` (`id`),
  CONSTRAINT `rating_researcher` FOREIGN KEY (`researcherId`) REFERENCES
`researcher` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `check_rating` CHECK (
    (
      `rating` between 1
      and 10
    )
  )
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci
COMMENT = 'Αξιολογήσεις έργων από ερευνητές';
```

Προκειμένου να γίνει ο έλεγχος ότι ο ερευνητής που έκανε την αξιολόγηση δεν ανήκει στον οργανισμό που συμμετέχει στην πρόταση, υπάρχει ένα TRIGGER που ενεργοποιείται πριν γίνει INSERT στο table ως εξής:

```
CREATE DEFINER=`root`@`localhost` TRIGGER
`check_researcher_not_in_organization` BEFORE INSERT ON `project_rating`
FOR EACH ROW BEGIN
           IF rating_researcher_in_organization(NEW.projectId,
NEW.researcherId) = 1 THEN
                SET NEW.researcherId = NULL;
           END IF;
     END
```

Το TRIGGER αυτό χρησιμοποιεί την παρακάτω συνάρτηση (`rating_researcher_in_organization`) για να ελέγξει στον πίνακα **works_for** εάν ο ερευνητής που έκανε την αξιολόγηση ανήκει στον οργανισμό που διαχειρίζεται το έργο ως εξής:

```
CREATE DEFINER=`root`@`localhost` FUNCTION
`rating_researcher_in_organization`(newProjectId INT, newResearcherId
INT) RETURNS int
    READS SQL DATA
BEGIN
RETURN EXISTS (
     SELECT m.organizationName FROM managed_by AS m
     JOIN works_for AS w
     ON newResearcherId = w.researcherId AND m.organizationName =
w.organizationName
     WHERE m.projectId = newProjectId
);
END
```

Ευρετήρια που έχουν οριστεί για τους πίνακες της ΒΔ

Για όλα τα πεδία (στήλες) που έχουμε ορίσει ως κλειδιά (PRIMARY/FOREIGN KEY) ή ως πεδία μοναδικών τιμών (UNIQUE) όπως αναφέρθηκαν παραπάνω, έχουν δημιουργηθεί αυτόματα τα κατάλληλα ευρετήρια (INDEXES). Ο σκοπός των ευρετηρίων είναι να κάνουν την αναζήτηση σε έναν πίνακα πιο γρήγορη, συνεπώς ορίζουμε ευρετήρια σε στήλες που δεν ανήκουν στις παραπάνω κατηγορίες αλλά χρησιμοποιούνται συχνά σε ερωτήματα (queries).

Αρχικά, για τον πίνακα των οργανισμών (organization), ορίζουμε ευρετήριο για την κατηγορία (*category*) κάθε οργανισμού (Πανεπιστήμιο, Εταιρεία ή Ερευνητικό Κέντρο), ώστε να μπορούμε γρήγορα να αναζητήσουμε οργανισμούς που ανήκουν σε μία συγκεκριμένη κατηγορία.

```
ALTER TABLE `organization`
  ADD INDEX idx_category (`category`);
```

Για τον πίνακα των έργων/επιχορηγήσεων (project), ορίζουμε ευρετήρια για τις στήλες που αφορούν την ημερομηνία έναρξης (*start_date*), τη διάρκεια του έργου (*duration*) και τον τίτλο του (*title*), τα οποία χρησιμοποιούνται συχνά σε ερωτήματα.

```
ALTER TABLE `project`
  ADD INDEX idx_start_date (`start_date`),
  ADD INDEX idx_duration (`duration`),
  ADD INDEX idx_title (`title`);
```

Για τον πίνακα των ερευνητών (researcher), ορίζουμε ευρετήριο για το όνομα (*name*) και το επίθετο (*surname*) του ερευνητή, το οποίο χρησιμοποιείται σε διάφορα ερωτήματα για την ταξινόμησή τους σε αλφαβητική σειρά.

```
ALTER TABLE `researcher`
  ADD INDEX idx_name (`name`),
  ADD INDEX idx_surname (`surname`);
```

Όψεις του σχεσιακού μοντέλου

1. **Έργα/επιχορηγήσεις ανά ερευνητή**

```
CREATE VIEW db_project.projects_per_researcher AS
    SELECT
        r.id AS id,
        r.name AS name,
        r.surname AS surname,
        r.gender AS gender,
        r.birth_date AS birth_date,
        p.title AS project_title,
        p.budget AS project_budget,
        p.start_date AS project_start_date,
        p.end_date AS project_end_date,
```

```
        p.duration AS project_duration
    FROM (
        db_project.researcher AS r
        JOIN db_project.project_researchers AS pr ON r.id =
pr.researcherId
        JOIN db_project.project AS p ON p.id = pr.projectId
     )
    ORDER BY r.name, p.start_date;
```

2. **Αναλυτικές πληροφορίες ανά έργο**

```
CREATE VIEW db_project.project_information AS
    SELECT
        p.id AS id,
        p.title AS title,
        p.description AS description,
        p.budget AS budget,
        p.start_date AS startDate,
        p.end_date AS endDate,
        p.duration AS duration,
        e.name AS executiveName,
        pr.name AS programName,
        pr.address AS programAddress,
        rm.name AS researchManagerName,
        rm.surname AS researchManagerSurname,
        rm.gender AS researchManagerGender,
        rm.birth_date AS researchManagerBirthDate,
        rating.rating AS rating,
        rating.rating_date AS ratingDate,
        rr.name AS ratingResearcherName,
        rr.surname AS ratingResearcherSurname,
        o.name AS organizationName,
        o.abbreviation AS organizationAbbreviation,
        o.street AS organizationStreet,
        o.number AS organizationStreetNumber,
        o.postal_code AS organizationPostalCode,
        o.city AS organizationCity,
        o.category AS organizationCategory,
        rt.research_fieldName AS researchFieldName
    FROM
        db_project.project AS p
        JOIN db_project.executive AS e ON e.id = p.executiveId
        JOIN db_project.program AS pr ON pr.id = p.programId
        JOIN db_project.researcher AS rm ON rm.id = p.researchManagerId
        JOIN db_project.project_rating AS rating ON rating.projectId =
p.id
        JOIN db_project.researcher AS rr ON rr.id = rating.researcherId
        JOIN db_project.managed_by AS mb ON mb.projectId = p.id
        JOIN db_project.organization AS o ON o.name = mb.organizationName
        JOIN db_project.related_to AS rt ON rt.projectId = p.id
    ORDER BY p.id , rt.research_fieldName;
```

DDL και DML scripts

Τα scripts για τη δημιουργία του σχήματος της βάσης (**D**ata **D**efinition **L**anguage - DDL script) και την εισαγωγή των δεδομένων στη βάση (**D**ata **M**anipulation **L**anguage - DML script) βρίσκονται στο Github repository, και συγκεκριμένα στο φάκελο **database**, με ονόματα Schema.DDL.sql και Data.DML.sql αντίστοιχα.

Σημείωση: Τα dummy δεδομένα για την εργασία δημιουργήθηκαν με τη βοήθεια του **`Fill Database.`**

Αναλυτικά βήματα εγκατάστασης της εφαρμογής

Το tech stack που χρησιμοποιήθηκε για την εφαρμογή είναι η MySQL για τη βάση δεδομένων, backend σε Node.js (με τη χρήση express) και HTML/CSS/JavaScript για το frontend. Για την εγκατάσταση και χρήση της εφαρμογής απαιτούνται τα παρακάτω βήματα:

1. Εγκατάσταση **Node.js** (χρησιμοποιήθηκε η έκδοση *14.18.0*)
2. Εγκατάσταση **MySQL** (χρησιμοποιήθηκε η έκδοση *8.0.27*)
3. Αντιγραφή της εφαρμογής από το **Github repository**.
4. Από το φάκελο στον οποίο έχουμε κατεβάσει την εφαρμογή, μεταβαίνουμε στον υποφάκελο **database** (cd database).
5. Κάνουμε import μέσω του MySQL τα scripts που υπάρχουν στο φάκελο, πρώτα το *Schema.DDL.sql* για να δημιουργήσουμε το σχήμα της βάσης, και στη συνέχεια το *Data.DML.sql* για να εισάγουμε στη βάση τα απαιτούμενα δεδομένα, ώστε να επιτυγχάνουν τα ζητούμενα queries.
6. Στη συνέχεια μεταβαίνουμε πίσω στο φάκελο όπου έχουμε κατεβάσει το project, κι από εκεί στον φάκελο src -> config (cd ../src/config) και ανοίγουμε το αρχείο *db.js*, ώστε να κάνουμε τις απαραίτητες τροποποιήσεις στα στοιχεία σύνδεσης με τη βάση δεδομένων (διεύθυνση IP που τρέχει ο server {127.0.0.1}, port που τρέχει ο server {3306}, όνομα χρήστη {root}, κωδικός πρόσβασης {12345678kD!} και το όνομα της βάσης {db_project}).
7. Μεταβαίνουμε πίσω στο φάκελο *src*, και μέσω του Command Prompt (ή shell κτλ) εκτελούμε την εντολή ***npm install*** ώστε να εγκατασταθούν όλα τα απαραίτητα packages τα οποία χρησιμοποιεί η εφαρμογή.
8. Μέσω του Command Prompt (ή shell κτλ) εκτελούμε την εντολή **node .** ώστε να ξεκινήσει η εφαρμογή (server). Εάν ο server έχει ξεκινήσει επιτυχώς, θα εμφανιστεί στην κονσόλα το μήνυμα *[server]: Server is running at http://localhost:80*
   Μέσω οποιουδήποτε φυλλομετρητή (browser) μπορούμε να μεταβούμε στη διεύθυνση http://localhost και να δούμε το frontend της εφαρμογής, ώστε να περιηγηθούμε στις αντίστοιχες σελίδες που υλοποιούν τις ζητούμενες λειτουργίες.

GitHub repository της εφαρμογής

https://github.com/kdivriotis/db_project