Βάσεις Δεδομένων

ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ

Ακαδημαϊκό Έτος 2021-2022

Ομάδα Project 77

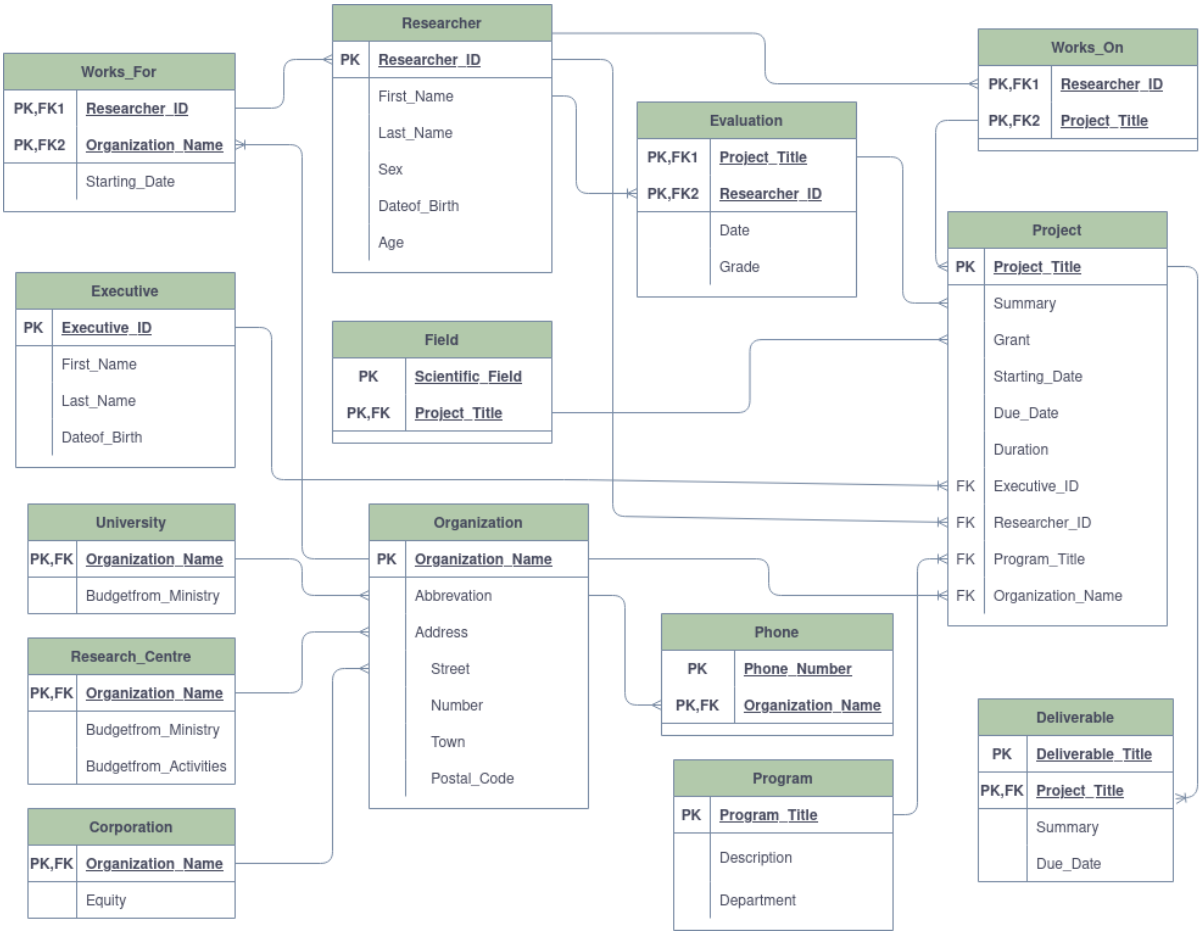
Μαρία Κοιλαλού | el19211

Δέσποινα Τόμκου | el19181

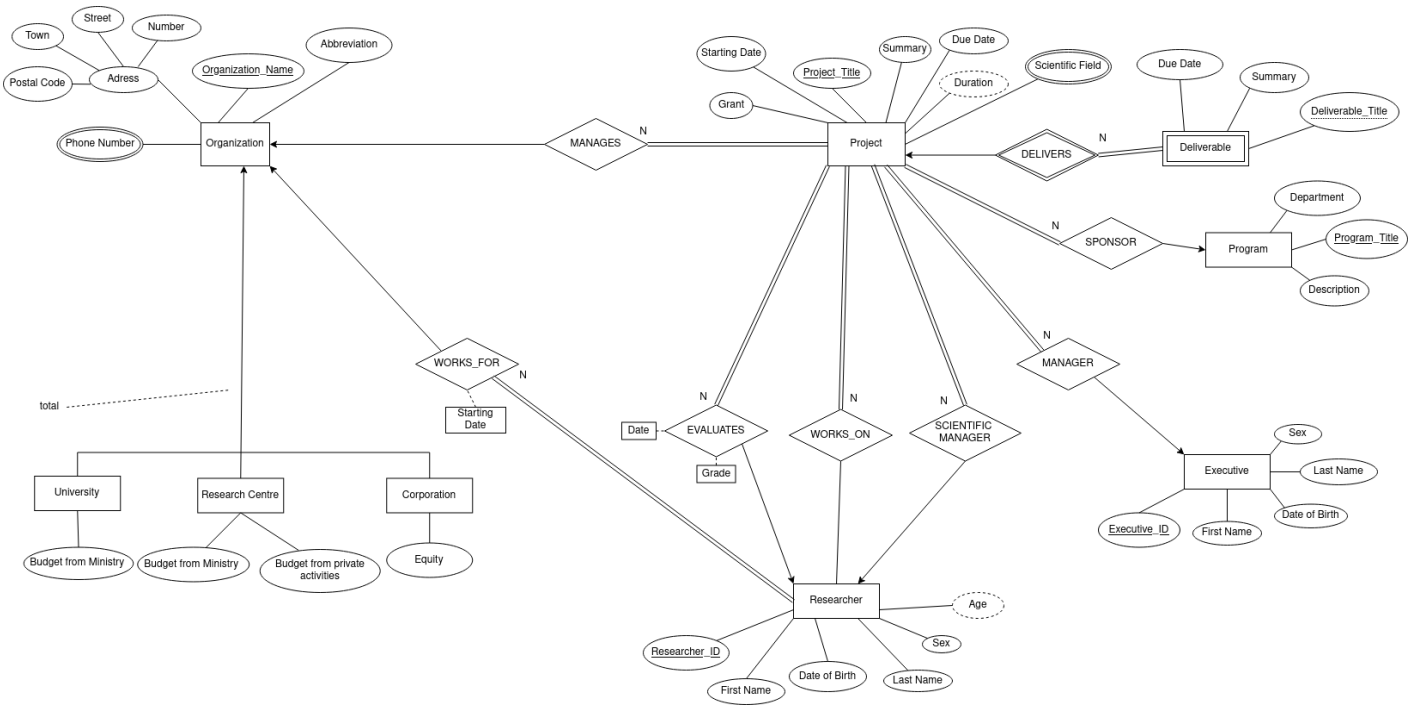
Φωτεινή Σταθοπούλου | el19032

Το ζητούμενο σύστημα αποθήκευσης αναπτύχθηκε για την αποθήκευση, διαχείριση και ανάλυση των πληροφοριών που συγκεντρώνονται στο ίδρυμα.

2.1. Παραθέτουμε παρακάτω το Σχεσιακό Διάγραμμα της βάσης μας



Επίσης παραθέτουμε και το ER Diagram διότι έχουν γίνει μερικές μικρές τροποποιήσεις από το 1ο Παραδοτέο.



Ξεκινώντας με γνώμονα το ER Diagram φτιάχνουμε το Σχεσιακό Μοντέλο μετατρέποντας τα Entity Sets σε Relations με Attributes τα αντίστοιχα Attributes που έχει το κάθε Entity Set. Επίσης μετατρέπουμε και όλες τις Many-to-Many Relationships σε Relations ενώ σε κάθε One-to-Many Relationship βάζουμε το Primary Key της πλευράς του One ως Foreign Key στην πλευρά των Many.

Δημιουργούμε τους παραπάνω πίνακες στην SQL θέτοντας ως NOT NULL τα Primary και Foreign Keys αφού είναι απαραίτητα για την αρχικοποίηση και την φυσική υπόσταση της βάσης μας.

Κάθε Primary και Foreign Key δημιουργεί αυτόματα και το δικό του Index στο Workbench. Εμείς δημιουργήσαμε άλλα 5 ευρετήρια για την ελαχιστοποίηση του χρόνου υλοποίησης των queries.

```
1 CREATE INDEX idx_Last_Name_Researcher ON Researcher(Last_Name);
2 CREATE INDEX idx_Age ON Researcher(Age);
3 CREATE INDEX idx_Due_Date ON Project(Due_Date);
4 CREATE INDEX idx_First_Name_Executive ON Executive(First_Name);
5 CREATE INDEX idx_Last_Name_Executive ON Executive(Last_Name);
```

2.2 DDL & DML

Query για την δημιουργία της βάσης

```
1     CREATE DATABASE `database`;
2 •   ALTER DATABASE `database` CHARACTER
3     SET utf8
4     COLLATE utf8_bin;
5 •   USE `database`;
```

Αφού έχουμε κατασκευάσει το αρχικό schema μας ξεκινάμε να κατασκευάζουμε τα Tables με την παρακάτω σειρά.

Ξεκινάμε κατασκευάζοντας πρώτα τα Tables που δεν έχουν κάποιο Foreign Key ως Attribute δηλαδή δεν εξαρτώνται από κάποιο άλλο Table.

Query για την δημιουργία του table **Researcher**

```
CREATE TABLE `Researcher` (
  `Researcher_ID` bigint(20) NOT NULL,
  `First_Name` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Last_Name` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Sex` enum('M','F') COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Dateof_Birth` date DEFAULT NULL,
  `Age` int(11) GENERATED ALWAYS AS (2022 - to_days(`Dateof_Birth`) / 365.25) VIRTUAL,
  PRIMARY KEY (`Researcher_ID`),
  UNIQUE KEY `Researcher_ID_UNIQUE` (`Researcher_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Researcher_ID είναι το Primary Key αφού είναι ένας αριθμός μοναδικός για κάθε έναν Researcher . Υπάρχουν κάποια επιμέρους χαρακτηριστικά του ως Attributes και η Ηλικία του που είναι ένα παραγόμενο Attribute αφού υπολογίζεται από την ημερομηνία γέννησης του.

Query για την δημιουργία του table **Organization**

```
CREATE TABLE `Organization` (
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Abbrevation` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Street` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Town` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Number` int(11) DEFAULT NULL,
  `Postal_Code` int(11) DEFAULT NULL,
  PRIMARY KEY (`Organization_Name`),
  UNIQUE KEY `Organization_Name_UNIQUE` (`Organization_Name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Organization_Name είναι το Primary Key αφού είναι μοναδικό για κάθε έναν Organization.
Αφού φτιάξουμε το Table για τον Organization, μπορούμε να φτιάξουμε και τα επιμέρους Tables για τις περιπτώσεις που ο οργανισμός είναι University, Research Centre και Corporation.

Query για την δημιουργία του table **University**

```
CREATE TABLE `University` (
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Budgetof_Ministry` bigint(20) DEFAULT NULL,
  PRIMARY KEY (`Organization_Name`),
  UNIQUE KEY `Organization_Name_UNIQUE` (`Organization_Name`),
  CONSTRAINT `fk_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization`
(`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Organization_Name είναι Primary και Foreign Key στο Table University είναι μοναδικό για κάθε ένα University ενώ παράλληλα κληρονομεί από το table Organization όλα τα επιμέρους χαρακτηριστικά που έχει ένα University εξαιτίας της λειτουργίας τους ως Organization.

Query για την δημιουργία του table **Research_Centre**

```
CREATE TABLE `Research_Centre` (
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Budgetfrom_Ministry` bigint(20) DEFAULT NULL,
  `Budgetfrom_Activities` bigint(20) DEFAULT NULL,
  PRIMARY KEY (`Organization_Name`),
  CONSTRAINT `fk3_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization`
(`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Organization_Name είναι Primary και Foreign Key στο Table Research Centre είναι μοναδικό για κάθε ένα Research Centre ενώ παράλληλα κληρονομεί από το table Organization όλα τα επιμέρους χαρακτηριστικά που έχει ένα Research Centre εξαιτίας της λειτουργίας τους ως Organization.

Query για την δημιουργία του table **Corporation**

```
CREATE TABLE `Corporation` (
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Equity` bigint(20) DEFAULT NULL,
  PRIMARY KEY (`Organization_Name`),
  CONSTRAINT `fk8_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization`
(`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Organization_Name είναι Primary και Foreign Key στο Table Corporation είναι μοναδικό για κάθε ένα Corporation ενώ παράλληλα κληρονομεί από το table Organization όλα τα επιμέρους χαρακτηριστικά που έχει ένα Corporation εξαιτίας της λειτουργίας τους ως Organization.

Query για την δημιουργία του table **Phone**

```
CREATE TABLE `Phone` (
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Phone_Number` bigint(20) NOT NULL,
  PRIMARY KEY (`Organization_Name`,`Phone_Number`),
  KEY `fk4_Organization_Name_idx` (`Organization_Name`),
  CONSTRAINT `fk4_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization`
(`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Phone_Number και το Organization_Name αποτελούν το Primary Key αφού κάθε Organization έχει ένα ή παραπάνω τηλέφωνα επικοινωνίας. Το Organization_Name είναι και Foreign Key από το Table Organization.

Αφού περάσαμε όλα τα Tables που είναι άμμεσα παραγόμενα από το Organization συνεχίζουμε με το Table Program και Executive ώστε να έχουμε όλα τα Tables που χρειαζόμαστε για την κατασκευή του Table **Project**

Query για την δημιουργία του table **Program**

```
CREATE TABLE `Program` (
  `Program_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Description` mediumtext COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Department` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  PRIMARY KEY (`Program_Title`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Program_Title είναι το Primary Key αφού είναι μοναδικό για κάθε ένα Program.

Query για την δημιουργία του table **Executive**

```
CREATE TABLE `Executive` (
  `Executive_ID` bigint(20) NOT NULL,
  `First_Name` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Last_Name` varchar(45) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Dateof_Birth` date DEFAULT NULL,
  PRIMARY KEY (`Executive_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Executive_ID είναι το Primary Key αφού είναι μοναδικό για κάθε ένα Program.

Query για την δημιουργία του table **Project**

```
CREATE TABLE `Project` (
  `Project_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Summary` mediumtext COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Grant` double DEFAULT NULL,
  `Starting_Date` date DEFAULT NULL,
  `Due_Date` date DEFAULT NULL,
  `Program_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Executive_ID` bigint(20) NOT NULL,
  `Researcher_ID` bigint(20) NOT NULL,
  `Duration` bigint(20) GENERATED ALWAYS AS (to_days(`Due_Date`) - to_days(`Starting_Date`)) VIRTUAL,
  PRIMARY KEY (`Project_Title`),
  UNIQUE KEY `Project_Title_UNIQUE` (`Project_Title`),
  KEY `fk4_Program_Title_idx` (`Program_Title`),
  KEY `fk5_Organization_Name_idx` (`Organization_Name`),
  KEY `fk5_Executive_ID_idx` (`Executive_ID`),
  KEY `fk10_Researcher_ID_idx` (`Researcher_ID`),
  KEY `fk11_Researcher_ID_idx` (`Researcher_ID`),
  CONSTRAINT `fk11_Researcher_ID` FOREIGN KEY (`Researcher_ID`) REFERENCES `Researcher` (`Researcher_ID`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk4_Program_Title` FOREIGN KEY (`Program_Title`) REFERENCES `Program` (`Program_Title`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk5_Executive_ID` FOREIGN KEY (`Executive_ID`) REFERENCES `Executive` (`Executive_ID`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk5_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization` (`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Table Project έχει ως Primary Key το Project_Title που είναι μοναδικό για κάθε ένα από τα Projects. Επίσης έχει 4 Foreign Keys, το Program_Title για το πρόγραμμα το οποίο χρηματοδοτεί το έργο, το Organization_Name για τον οργανισμό που το διαχειρίζεται, το Researcher_ID για τον Scientific Manager του έργου (Οι Researchers που δουλεύουν στο έργο ή είναι υπεύθυνοι για το Evaluation του έργου βρίσκονται στα tables Works_On και Evaluation αντίστοιχα) και το Executive_ID για το στέλεχος του ΕΛ.ΙΔ.Ε.Κ που το διαχειρίζεται. Κάθε ένα από τα Foreign Keys κληρονομεί τα χαρακτηριστικά του αντίστοιχου table από το οποίο προήλθε. Επίσης έχουμε ως παραγόμενο Attribute το Duration που παράγεται από την ημερομηνία έναρξης και λήξης του Project.

Τώρα έχουμε όλα τα βασικά Tables για να φτιάξουμε όλα τα υπόλοιπα που εξαρτώνται από αυτά λόγω του ότι κληρονομούν μέσω των Foreign Keys τα χαρακτηριστικά τους.

Query για την δημιουργία του table **Deliverable**

```
CREATE TABLE `Deliverable` (
  `Deliverable_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Project_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Summary` mediumtext COLLATE utf8mb4_general_ci DEFAULT NULL,
  `Due_Date` date DEFAULT NULL,
  PRIMARY KEY (`Deliverable_Title`),
  KEY `fk1_Project_Title_idx` (`Project_Title`),
  CONSTRAINT `fk_Project_Title` FOREIGN KEY (`Project_Title`) REFERENCES `Project`
(`Project_Title`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Deliverable_Title και το Project_Title αποτελούν το Primary Key αφού κάθε Deliverable που ανήκει σε συγκεκριμένο έργο έχει μοναδικό όνομα. Το Project_Title είναι και Foreign Key από το Table Project.

Query για την δημιουργία του table **Evaluation**

```
CREATE TABLE `Evaluation` (
  `Project_Title` varchar(45) NOT NULL,
  `Researcher_ID` bigint NOT NULL,
  `Date` date DEFAULT NULL,
  `Grade` int DEFAULT NULL,
  PRIMARY KEY (`Project_Title`,`Researcher_ID`),
  UNIQUE KEY `Project_Title_UNIQUE` (`Project_Title`),
  KEY `fk0_Researcher_ID` (`Researcher_ID`),
  CONSTRAINT `fk0_Researcher_ID` FOREIGN KEY (`Researcher_ID`) REFERENCES `Researcher`
(`Researcher_ID`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk2_Project_Title` FOREIGN KEY (`Project_Title`) REFERENCES `Project`
(`Project_Title`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ai_ci
```

Το Researcher_ID και το Project_Title αποτελούν το Primary Key αφού κάθε Evaluation που αφορά ένα συγκεκριμένο έργο γίνεται evaluate από συγκεκριμένο Researcher .Το Project_Title και το Researcher_ID είναι και Foreign Keys από το Table Project και Researcher.

Query για την δημιουργία του table **Field**

```
DROP TABLE IF EXISTS `Field`;

CREATE TABLE `Field` (
  `Scientific_Field` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Project_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  PRIMARY KEY (`Scientific_Field`,`Project_Title`),
  KEY `fk3_Project_Title_idx` (`Project_Title`),
  CONSTRAINT `fk3_Project_Title` FOREIGN KEY (`Project_Title`) REFERENCES `Project`
(`Project_Title`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Scientific_Field και το Project_Title αποτελούν το Primary Key αφού κάθε Project ανήκει σε ένα ή περισσότερα Scientific Fieds. Το Project_Title είναι και Foreign Key από το Table Project.

Query για την δημιουργία του table **Works_On**

```
CREATE TABLE `Works_On` (
  `Project_Title` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Researcher_ID` bigint(20) NOT NULL,
  PRIMARY KEY (`Project_Title`,`Researcher_ID`),
  KEY `fk10_Project_Title_idx` (`Project_Title`),
  KEY `fk10_Researcher_ID_idx` (`Researcher_ID`),
  CONSTRAINT `fk10_Project_Title` FOREIGN KEY (`Project_Title`) REFERENCES `Project`
(`Project_Title`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk10_Researcher_ID` FOREIGN KEY (`Researcher_ID`) REFERENCES `Researcher`
(`Researcher_ID`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Researcher_ID και το Project_Title αποτελούν το Primary Key αφού κάθε Researcher δουλεύει σε ένα ή περισσότερα έργα .Το Project_Title και το Researcher_ID είναι και Foreign Keys από το Table Project και Researcher.

Query για την δημιουργία του table **Works_For**

```
CREATE TABLE `Works_For` (
  `Researcher_ID` bigint(20) NOT NULL,
  `Organization_Name` varchar(45) COLLATE utf8mb4_general_ci NOT NULL,
  `Starting_Date` date DEFAULT NULL,
  PRIMARY KEY (`Researcher_ID`,`Organization_Name`),
  UNIQUE KEY `Researcher_ID_UNIQUE` (`Researcher_ID`),
  KEY `fk6_Organization_Name_idx` (`Organization_Name`),
  CONSTRAINT `fk6_Researcher_ID` FOREIGN KEY (`Researcher_ID`) REFERENCES `Researcher`
(`Researcher_ID`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk9_Organization_Name` FOREIGN KEY (`Organization_Name`) REFERENCES `Organization`
(`Organization_Name`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Το Researcher_ID και το Organization_Name αποτελούν το Primary Key αφού σε κάθε Organization δουλεύει ένας ή περισσότεροι Researchers .Το Organization_Name και το Researcher_ID είναι και Foreign Keys από το Table Organization και Researcher.

Έχωντας κατασκευάσει όλα τα Tables κάνουμε insert τα Dummy Data από την ιστιοσελίδα Mockaroo για να γεμίσουμε την βάση μας, τρέχοντας το αρχείο InsertDummyData.sql που υπάρχει στο git repo μας.

Πριν αρχίσουμε να εκτελούμε Queries πάνω στην βάση μας προσθέτουμε διάφορα triggers που θα προφυλάξουμε την βάση μας από ανεπιθύμητες ενέργειες.

Trigger 1

```
1 delimiter //
2 CREATE TRIGGER EvaluatorCheck BEFORE INSERT ON Evaluation
3 FOR EACH ROW
4 BEGIN
5 DECLARE msg varchar(45);
6 IF NEW.Researcher_ID IN (SELECT c.Researcher_ID
7                          FROM Works_For c
8                          INNER JOIN Project r ON c.Organization_Name=r.Organization_Name
9                          WHERE NEW.Project_Title=r.Project_Title
0                         )
1 THEN
2 SET msg=('This researcher is already working for the organization on this project');
3 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT=msg;
4 END IF;
5 END//
6 delimeter;
```

Με αυτό το trigger πρωτού κάποιος πάει να προσθέσει έναν Researcher στον πίνακα Evaluation θα πρέπει να σιγουρευτεί πως ο Researcher στον οποίο αναθέτει την αξιολόγηση ενός Project δεν δουλεύει για τον Οργανισμό που το έχει αναλάβει.

Trigger 2

```
1 delimiter//
2 CREATE TRIGGER IsEvaluator BEFORE INSERT ON Works_On
3 FOR EACH ROW
4 BEGIN
5 DECLARE msg varchar(45);
6 IF NEW.Researcher_ID IN (SELECT c.Researcher_ID FROM Evaluation c
7                                                  INNER JOIN Works_On r ON
  c.Project_Title=r.Project_Title)
8 THEN
9 SET msg=('This researcher is already an evaluator in this project');
0 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT=msg;
1 END IF;
2 END//
3 delimiter;
```

Με αυτό το trigger πρωτού κάποιος πάει να προσθέσει έναν Researcher στον πίνακα Works_On θα πρέπει να σιγουρευτεί πως ο Researcher αυτός δεν είναι υπεύθυνος για την αξιολόγηση αυτού του Project.

Τώρα είμαστε σε θέση να εκτελέσουμε οποιοδήποτε Query στην βάση μας και να περιμένουμε να μας δώσει ορθό αποτέλεσμα.

Query 3.1: **Projects**

```
#3.1
SELECT P.Project_Title AS Project_Title,
       P.Summary AS Summary,
       P.Starting_Date AS Starting_Date,
       P.Duration AS Duration,
       CONCAT(E.First_Name , ' ' ,E.Last_Name) AS ExName FROM Project P
       INNER JOIN Executive E ON P.Executive_ID=E.Executive_ID WHERE P.Due_Date>curdate()
```

Query 3.3: **Ερευνητικό Πεδίο με ιδιαίτερο ενδιαφέρον**

```
#3.3
SELECT  F.Scientific_Field AS SField,
P.Project_Title AS Title,
CONCAT(R.First_Name , ' ' ,R.Last_Name) AS RName
FROM Field F
INNER JOIN Project P ON P.Project_Title = F.Project_Title
INNER JOIN Works_On W ON W.Project_Title = P.Project_Title
INNER JOIN Researcher R ON R.Researcher_ID = W.Researcher_ID
WHERE P.Starting_Date <= '2021-06-05' AND P.Due_Date >= '2022-06-05' ")
```

Query 3.4: **Οργανισμός με ίδιο αριθμό έργων για 2 συνεχόμενα έτη**

```
#3.4
SELECT * FROM
(SELECT Organization_Name, Ye,  Project_Cnt , LEAD(Project_Cnt, 1)
OVER (PARTITION BY Organization_Name ORDER BY Ye) Next_Project_Cnt
FROM
        (SELECT COUNT(P.Project_Title) AS Project_Cnt,
                O.Organization_Name ,
                YEAR(P.Starting_Date) AS Ye
         FROM Organization O
         INNER JOIN Project P ON P.Organization_Name = O.Organization_Name
         GROUP BY O.Organization_Name, YEAR(P.Starting_Date)
         HAVING COUNT(P.Project_Title) >= 10
        ) A
) B
WHERE Project_Cnt = Next_Project_Cnt;
```

Query 3.5: **Top-3 Ζεύγη Επιστημονικών Πεδίων**

```
#3.5
SELECT F1.Scientific_Field, F2.Scientific_Field, COUNT(F1.Project_Title)
FROM Field F1
INNER JOIN Field F2
        ON F1.Project_Title = F2.Project_Title
       AND F1.Scientific_Field < F2.Scientific_Field
WHERE F1.Scientific_Field <> F2.Scientific_Field
GROUP BY F1.Scientific_Field, F2.Scientific_Field
ORDER BY COUNT(F1.Project_Title) DESC
LIMIT 3;
```

Query 3.6: **Νέοι Ερευνητές με τα περισσότερα ενεργά έργα**

```
#3.6
SELECT R.Researcher_ID, R.Last_Name, R.First_Name, COUNT(W.Project_Title) Project_Cnt
FROM Researcher R
INNER JOIN Works_On W
              ON W.Researcher_ID = R.Researcher_ID
WHERE R.Age < 40
GROUP BY R.Researcher_ID, R.Last_Name, R.First_Name
HAVING COUNT(Project_Title) = (SELECT MAX(Project_Cnt)
                               FROM
                               (SELECT R.Researcher_ID, R.Last_Name, R.First_Name,
                               COUNT(W.Project_Title) Project_Cnt
                               FROM Researcher R
                               INNER JOIN Works_On W ON W.Researcher_ID = R.Researcher_ID
                               INNER JOIN Project P ON P.Project_Title = W.Project_Title
                               WHERE R.Age < 40 AND P.Due_Date >= '2022-05-28'
                               GROUP BY R.Researcher_ID, R.Last_Name, R.First_Name
                               ) A);
```

Query 3.7:

Top-5 Στελέχη με τα μεγαλύτερα ποσά χρηματοδότησης σε μια εταιρία

```
#3.7
SELECT SUM(P.Grant), C.Organization_Name, E.First_Name, E.Last_Name
FROM Executive E
INNER JOIN Project P ON E.Executive_ID = P.Executive_ID
INNER JOIN Organization O ON O.Organization_Name = P.Organization_Name
INNER JOIN Corporation C ON C.Organization_Name = O.Organization_Name
GROUP BY C.Organization_Name, E.Executive_ID
ORDER BY SUM(P.Grant) DESC
LIMIT 5;
```

Query 3.8:
Ερευνητές που εργάζονται σε τουλάχιστον 5 έργα χωρίς παραδοτέο

```
#3.8
SELECT COUNT(P.Project_Title), R.First_Name, R.Last_Name
FROM Researcher R
INNER JOIN Works_On W ON R.Researcher_ID = W.Researcher_ID
INNER JOIN Project P ON P.Project_Title = W.Project_Title
LEFT JOIN Deliverable D ON D.Project_Title = P.Project_Title
WHERE D.Project_Title IS NULL
GROUP BY  R.First_Name, R.Last_Name
HAVING (COUNT(P.Project_Title))>=5;
```

Επίσης πέρα από τα Queries που μπορούμε να εκτελέσουμε πάνω στην βάση μας μπορούμε να κατασκευάσουμε και διάφορες όψεις του σχεσιακού μοντέλου που θα μπορεί να δει ο χρήστης

View 1

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `ProjectResearch` AS
    SELECT
        CONCAT(`c`.`First_Name`, ' ', `c`.`Last_Name`) AS `Researcher_Name`,
        `a`.`Project_Title` AS `Project_Title`,
        CONCAT(`r`.`First_Name`, ' ', `r`.`Last_Name`) AS `ScientificManager_Name`,
        CONCAT(`p`.`First_Name`, ' ', `p`.`Last_Name`) AS `Evaluator_Name`
    FROM
        (((((`Project` `a`
        JOIN `Researcher` `r` ON ((`r`.`Researcher_ID` = `a`.`Researcher_ID`)))
        JOIN `Works_On` `w` ON ((`w`.`Project_Title` = `a`.`Project_Title`)))
        JOIN `Researcher` `c` ON ((`c`.`Researcher_ID` = `w`.`Researcher_ID`)))
        JOIN `Evaluation` `e` ON ((`e`.`Project_Title` = `a`.`Project_Title`)))
        JOIN `Researcher` `p` ON ((`p`.`Researcher_ID` = `e`.`Researcher_ID`)))
    WHERE
        ((`p`.`Researcher_ID` <> `c`.`Researcher_ID`)
            AND (`p`.`Researcher_ID` <> `r`.`Researcher_ID`))
    ORDER BY CONCAT(`c`.`First_Name`, ' ', `c`.`Last_Name`)
```

Αυτή η όψη του σχεσιακού μοντέλου παρουσιάζει τους Researchers και τα έργα στα οποία δουλεύουν μαζί με τον Scientific Manager και τον Evaluator του κάθε έργου.

View 2

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `ProjectEvaluation` AS
    SELECT
        `c`.`Project_Title` AS `Project_Title`,
        CONCAT(`w`.`First_Name`, ' ', `w`.`Last_Name`) AS `Evaluator_Name`,
        `r`.`Grade` AS `Grade`,
        `r`.`Date` AS `Date`,
        `c`.`Grant` AS `Grant`,
        `p`.`Program_Title` AS `Program_Title`,
        `p`.`Department` AS `Department`
    FROM
        (((`Project` `c`
        JOIN `Evaluation` `r` ON ((`r`.`Project_Title` = `c`.`Project_Title`)))
        JOIN `Researcher` `w` ON ((`w`.`Researcher_ID` = `r`.`Researcher_ID`)))
        JOIN `Program` `p` ON ((`p`.`Program_Title` = `c`.`Program_Title`)))
    ORDER BY `r`.`Grade` DESC
```

Αυτή η όψη του σχεσιακού μοντέλου παρουσιάζει την ημέρα, τον βαθμό και τον Evaluator του κάθε Project μαζί με το πρόγραμμα το οποίο τον χρηματοδοτει, το ποσό χρηματοδότησης του και την Διεύθυνση του ΕΛ.ΙΔ.Ε.Κ. στην οποία το πρόγραμμα ανήκει. Έτσι κάποιος μπορεί να δει τον βαθμό ενός Project μαζί με το ποσό χρηματοδότησης του και να μπορεί να αποφασίσει αν α[αιτούνται αυξήσεις και μειώσεις.

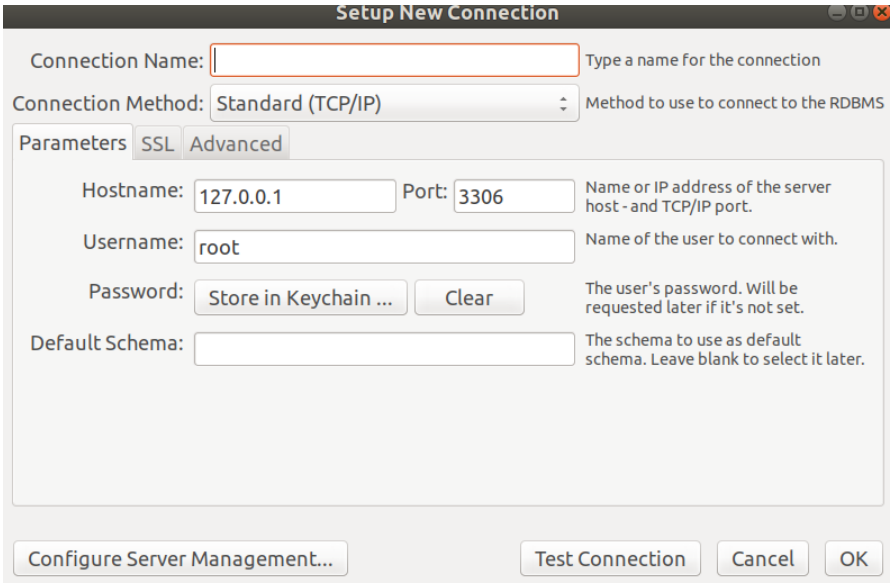
2.3 Τεχνολογία ανάπτυξης εφαρμογής

Για την διαχείριση και την ανάπτυξη της βάσης χρησιμοποιήθηκε το MySQL Workbench και για DBMS χρησιμοποιήθηκε η MySQL. Για το στήσιμο του web server χρησιμοποιείται node.js και για την σύνδεση μεταξύ βάσης και server χρησιμοποιείται το package mysql2 και ο driver MySQL Connector/J.
Για το UI χρησιμοποιήθηκε HTML, CSS, JavaScript, Node.js και NPM package manager.
Version:

- MySQL Ver 14.14 Distrib 5.7.38, for Linux (x86_64)
- MySQL Connector/J ver. 8.0.25
- node.js v14.18.0

2.4 Βήματα εγκατάστασης λογισμικού σε Linux (Ubuntu 18.04)

1. Απαιτείται η εγκατάσταση της MySQL Ver 14.14 Distrib 5.7.38, for Linux (x86_64) και του MySQL Workbench.
2. Αφού γίνει η εγκατάσταση φτιάχνουμε ένα connection με το επιθυμητό όνομα και password

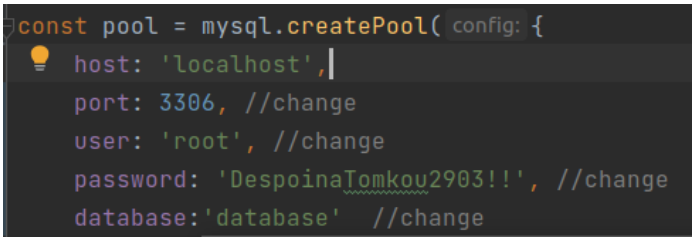


3. Σε αυτό το connection δημιουργούμε ένα νέο schema με το επιθυμητό όνομα και default collation. Με βάση τα παραπάνω βήματα φτιάχνουμε τα tables της βάσης δεδομένων μας, δημιουργούμε τα απαραίτητα views και εισάγουμε τα δεδομένα.
4. Προχωράμε στο στήσιμο του περιβάλλοντος πάνω στο οποίο τρέχει η εφαρμογή μας.Να επισημανθεί ότι για την αναπτυξη της εφαρμογής χρησιμοποιήθηκε IntelliJ IDEA. Εγκαθιστούμε node.js v14.18.0 και npm v6.14.15. Στην συνέχεια κάνουμε clone το project (databaseUI) σε κάποιον editor. Για παράδειγμα αν χρησιμοποιούμε IntelliJ εκτελούμε τις παρακάτω εντολές στο terminal του IntelliJ
    - ➢ cd IdeaProjects
    - ➢ git clone https://github.com/koutom111/DatabasesUI.git

5. Ανοίγουμε τον φάκελο databaseUI και ανοίγουμε το αρχείο package.json. Με την εντολή npm install εγκαθιστούμε τα παρακάτω dependencies που αναγράφονται μέσα στο αρχείο.

```
"dependencies": {
  "body-parser": "^1.20.0",
  "chalk": "^5.0.1",
  "connect-flash": "^0.1.1",
  "custom-env": "^2.0.1",
  "ejs": "^3.1.7",
  "express": "^4.18.1",
  "express-session": "^1.17.3",
  "faker": "^6.6.6",
  "mysql2": "^2.3.3"
},
"devDependencies": {
  "nodemon": "^2.0.16"
}
```

6. Στην συνέχεια ανοίγουμε τον φάκελο utils όπου βρίσκεται το database.js αρχείο. Εκεί προσαρμόζουμε τα port, user, password, database στην βάση που φτιάξαμε προηγουμένως.



```
const pool = mysql.createPool( config: {
    host: 'localhost',
    port: 3306, //change
    user: 'root', //change
    password: 'DespoinaTomkou2903!!', //change
    database:'database'  //change
```

7. Επιπλέον εγκαθιστούμε τον MySQL connector και τους απαραίτητους drivers. Για παράδειγμα στο IntelliJ αυτό γίνεται πηγαίνοντας στο View>Tool Windows>Database, πατώντας το + κουμπί και επιλέγοντας το Data source. Εκεί, αφού βάλουμε τα στοιχεία μας υπάρχει η επιλογή για εγκατάσταση όλων των απαραίτητων drivers.
8. Τώρα το πρόγραμμα είναι έτοιμο να τρέξει στον localhost με την εντολή node server.js

Ο πηγαίος κώδικας της εφαρμογής μας βρίσκεται στο git repository:
https://github.com/koutom111/DatabasesUI