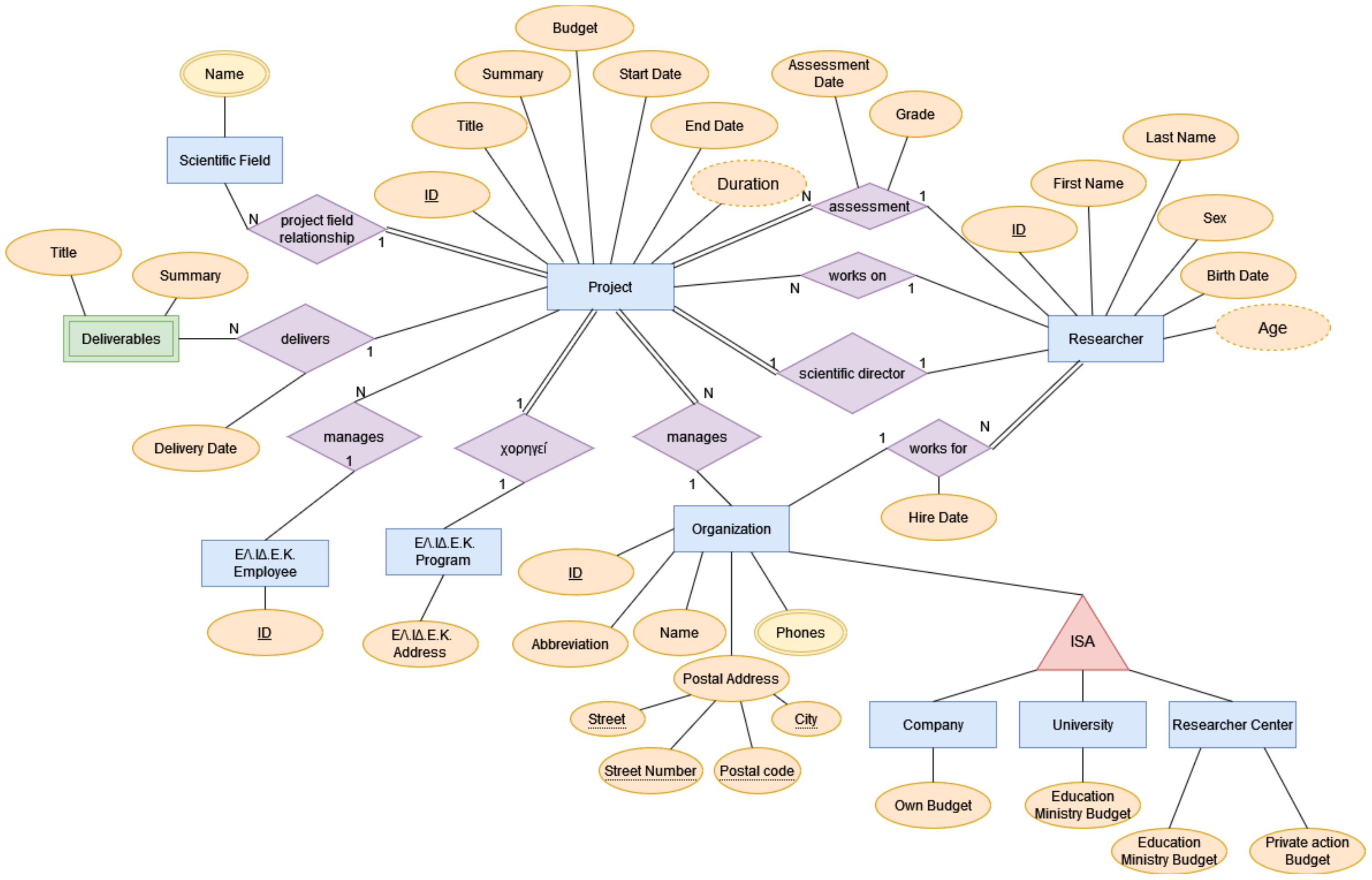
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Ακ. έτος 2021-2022, 6ο εξάμηνο, ΣΗΜΜΥ

Βάσεις Δεδομένων
Εξαμηνιαία Εργασία

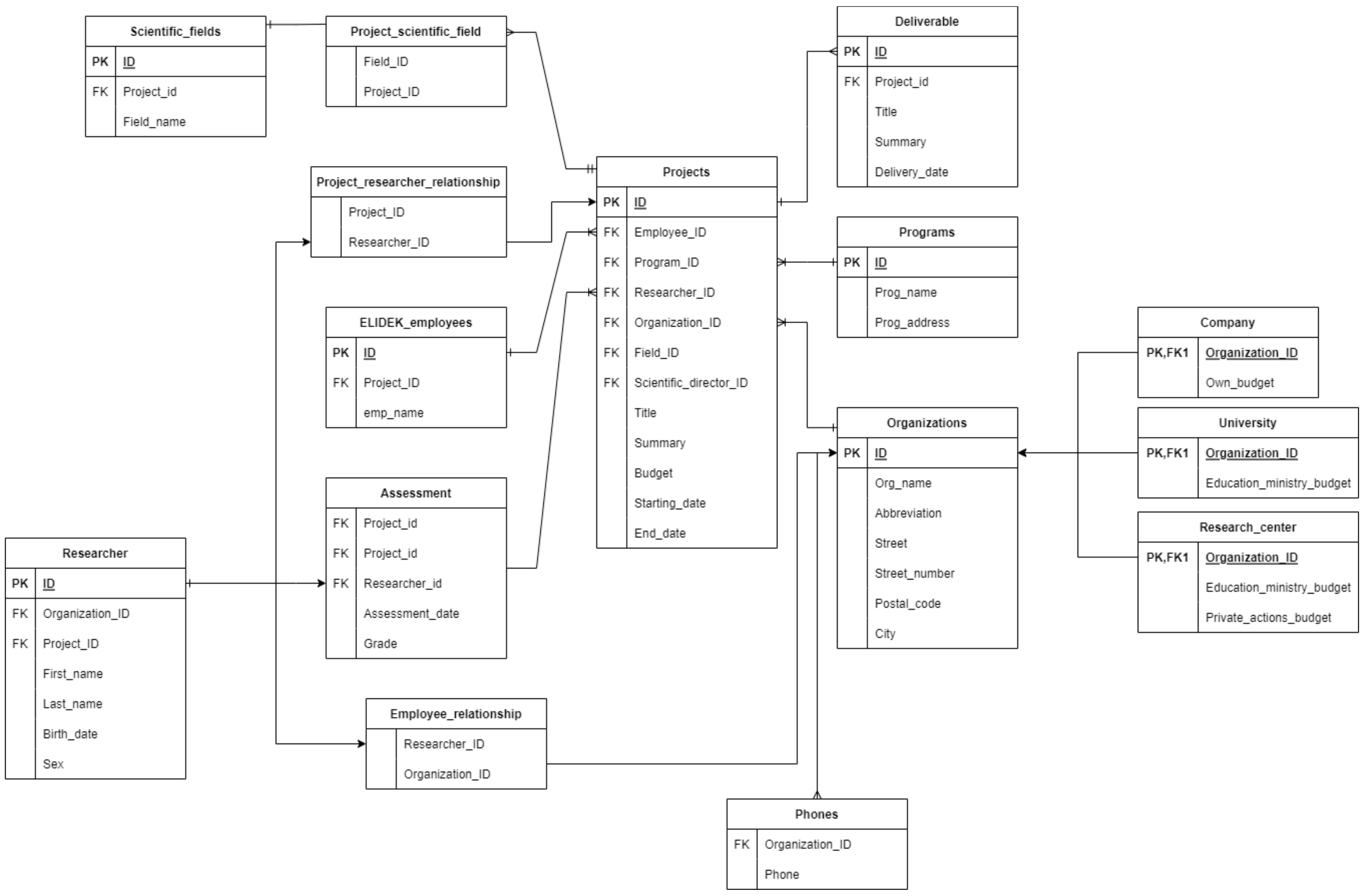
Μπακαλούδης Παναγιώτης 03119600
Τουρνάρης Ελευθέριος 03118845

Το Ελληνικό Ίδρυμα Έρευνας και Καινοτομίας - ΕΛ.ΙΔ.Ε.Κ. είναι σημαντικός χρηματοδότης της ακαδημαϊκής έρευνας στην Ελλάδα. Σχεδιάσαμε και υλοποιήσαμε ένα σύστημα αποθήκευσης, διαχείρισης και ανάλυσης των πληροφοριών που συγκεντρώνονται από το ίδρυμα.

Α. Βελτιωμένο ER διάγραμμα.

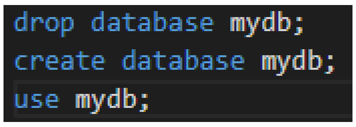


B. 2.1 Σχεσιακό διάγραμμα



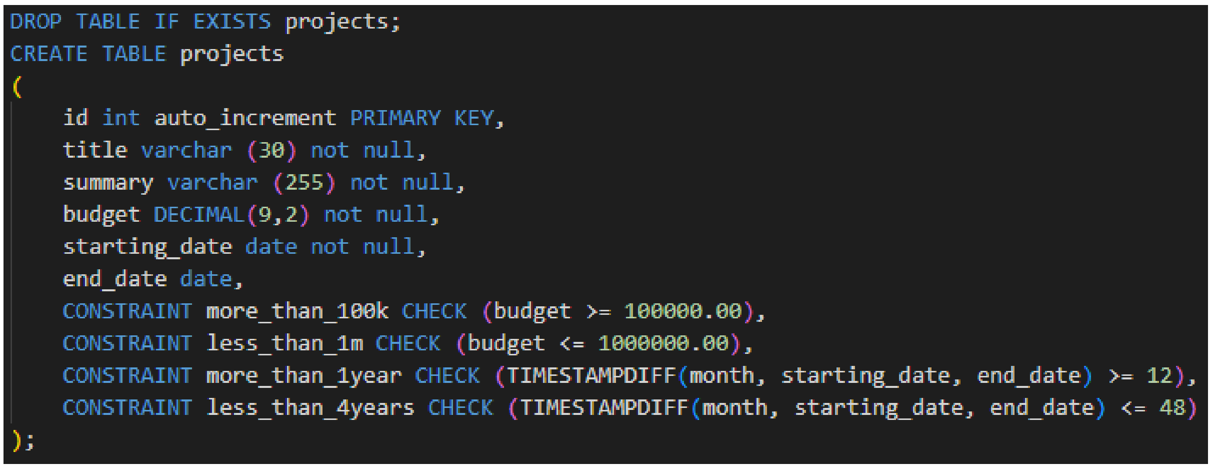
DDL Script

- Query για την δημιουργία της βάσης:



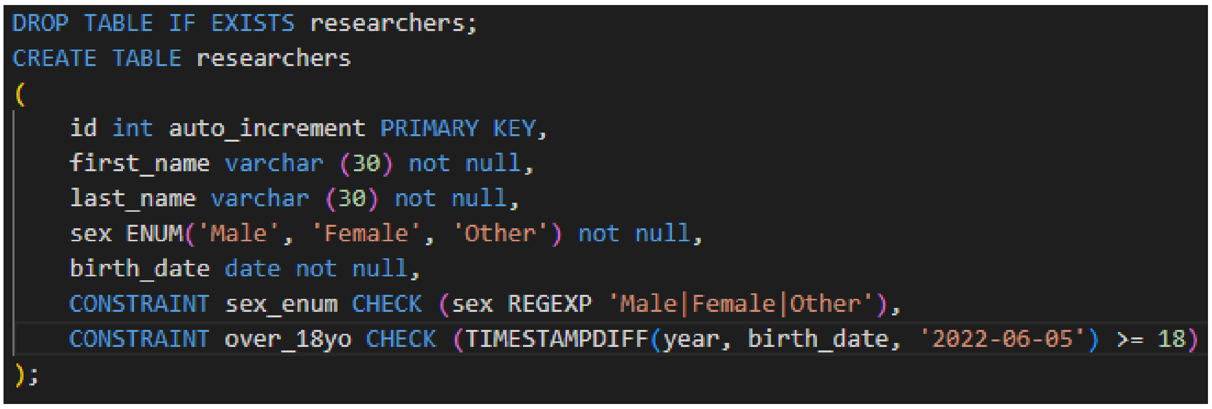
```sql
drop database mydb;
create database mydb;
use mydb;
```

- Query για την δημιουργία του πίνακα Projects:



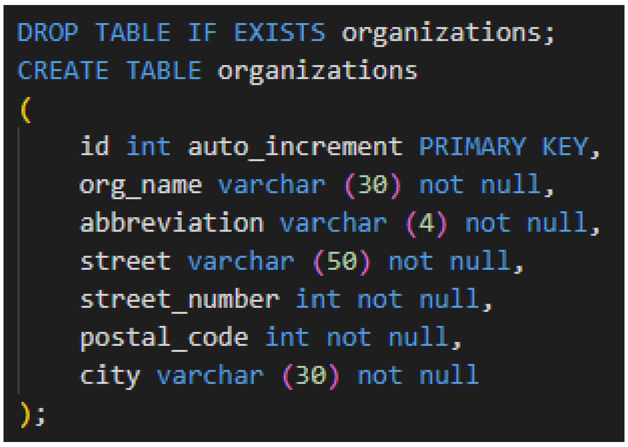
```sql
DROP TABLE IF EXISTS projects;
CREATE TABLE projects
(
    id int auto_increment PRIMARY KEY,
    title varchar (30) not null,
    summary varchar (255) not null,
    budget DECIMAL(9,2) not null,
    starting_date date not null,
    end_date date,
    CONSTRAINT more_than_100k CHECK (budget >= 100000.00),
    CONSTRAINT less_than_1m CHECK (budget <= 1000000.00),
    CONSTRAINT more_than_1year CHECK (TIMESTAMPDIFF(month, starting_date, end_date) >= 12),
    CONSTRAINT less_than_4years CHECK (TIMESTAMPDIFF(month, starting_date, end_date) <= 48)
);
```

- Query για την δημιουργία του πίνακα Researchers:



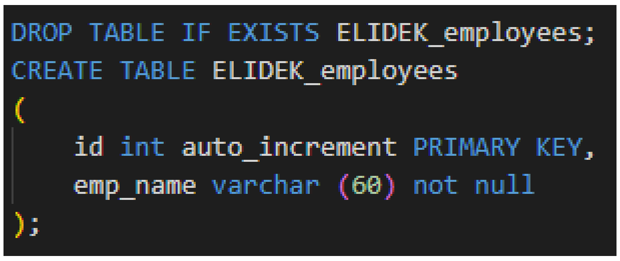
```sql
DROP TABLE IF EXISTS researchers;
CREATE TABLE researchers
(
    id int auto_increment PRIMARY KEY,
    first_name varchar (30) not null,
    last_name varchar (30) not null,
    sex ENUM('Male', 'Female', 'Other') not null,
    birth_date date not null,
    CONSTRAINT sex_enum CHECK (sex REGEXP 'Male|Female|Other'),
    CONSTRAINT over_18yo CHECK (TIMESTAMPDIFF(year, birth_date, '2022-06-05') >= 18)
);
```

- Query για την δημιουργία του πίνακα Organizations:



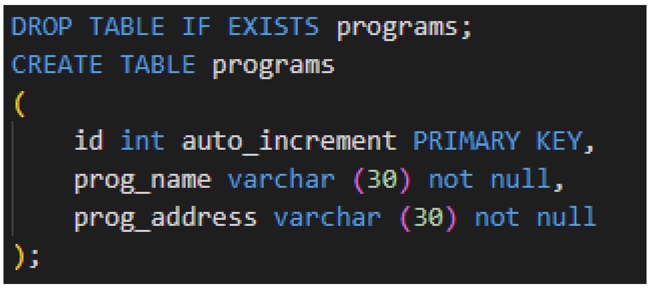
```sql
DROP TABLE IF EXISTS organizations;
CREATE TABLE organizations
(
    id int auto_increment PRIMARY KEY,
    org_name varchar (30) not null,
    abbreviation varchar (4) not null,
    street varchar (50) not null,
    street_number int not null,
    postal_code int not null,
    city varchar (30) not null
);
```

- Query για την δημιουργία του πίνακα ELIDEK employees:



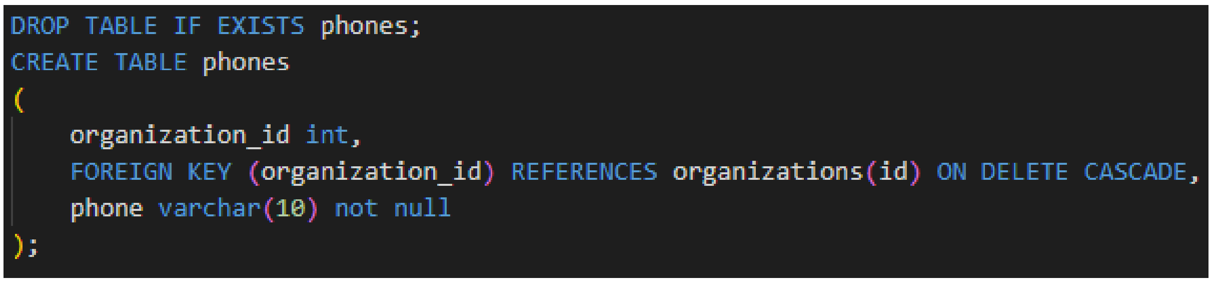
```
DROP TABLE IF EXISTS ELIDEK_employees;
CREATE TABLE ELIDEK_employees
(
    id int auto_increment PRIMARY KEY,
    emp_name varchar (60) not null
);
```

- Query για την δημιουργία του πίνακα Programs:



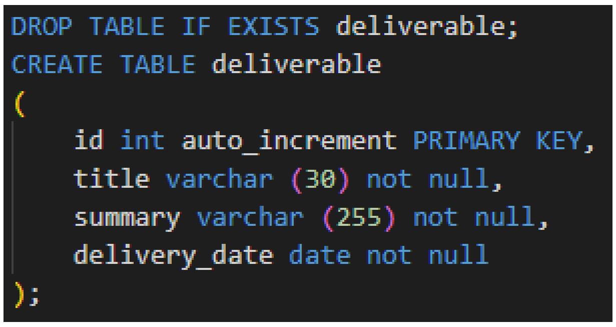
```
DROP TABLE IF EXISTS programs;
CREATE TABLE programs
(
    id int auto_increment PRIMARY KEY,
    prog_name varchar (30) not null,
    prog_address varchar (30) not null
);
```

- Query για την δημιουργία του πίνακα Scientific Fields:



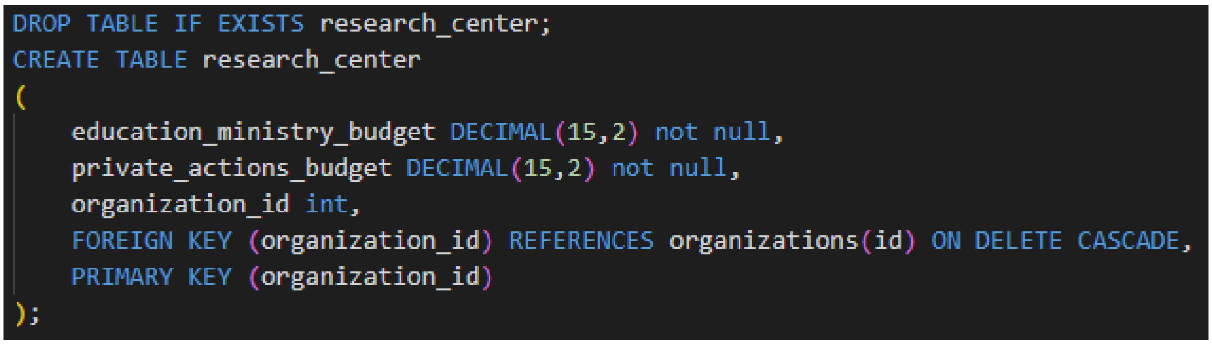
```
DROP TABLE IF EXISTS phones;
CREATE TABLE phones
(
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    phone varchar(10) not null
);
```

- Query για την δημιουργία του πίνακα Deliverable:



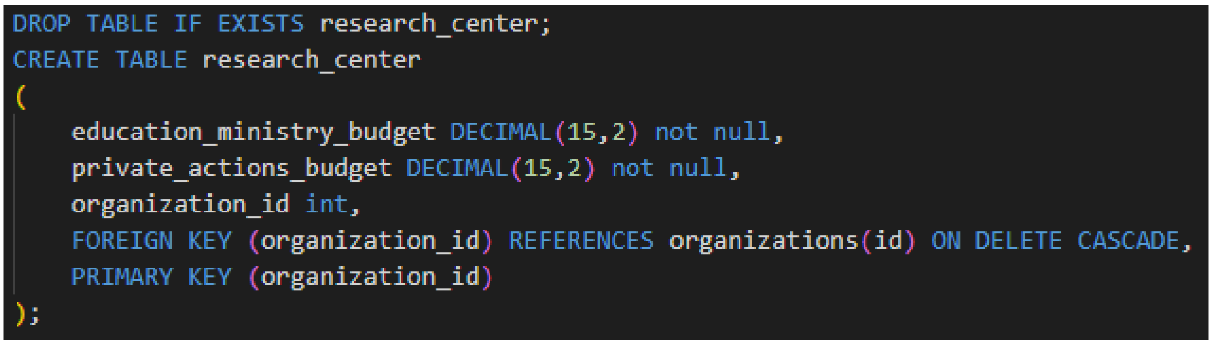
```
DROP TABLE IF EXISTS deliverable;
CREATE TABLE deliverable
(
    id int auto_increment PRIMARY KEY,
    title varchar (30) not null,
    summary varchar (255) not null,
    delivery_date date not null
);
```

- Query για την δημιουργία του πίνακα Phones:



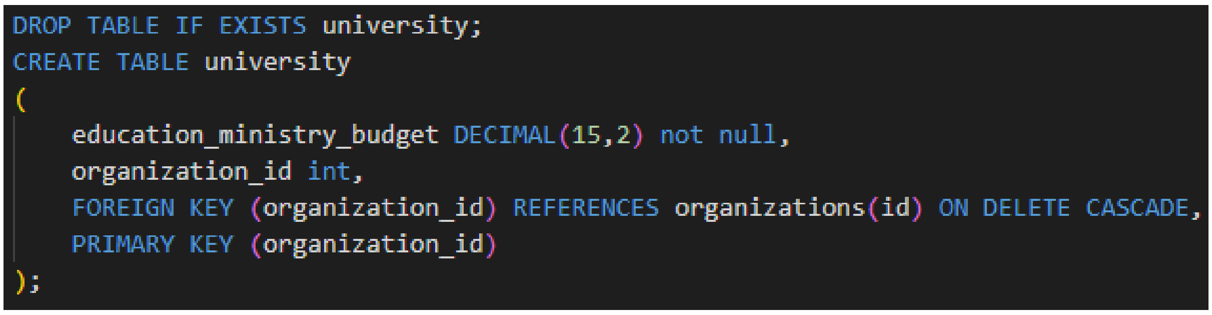
```
DROP TABLE IF EXISTS research_center;
CREATE TABLE research_center
(
    education_ministry_budget DECIMAL(15,2) not null,
    private_actions_budget DECIMAL(15,2) not null,
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    PRIMARY KEY (organization_id)
);
```

- Query για την δημιουργία του πίνακα Research Center:



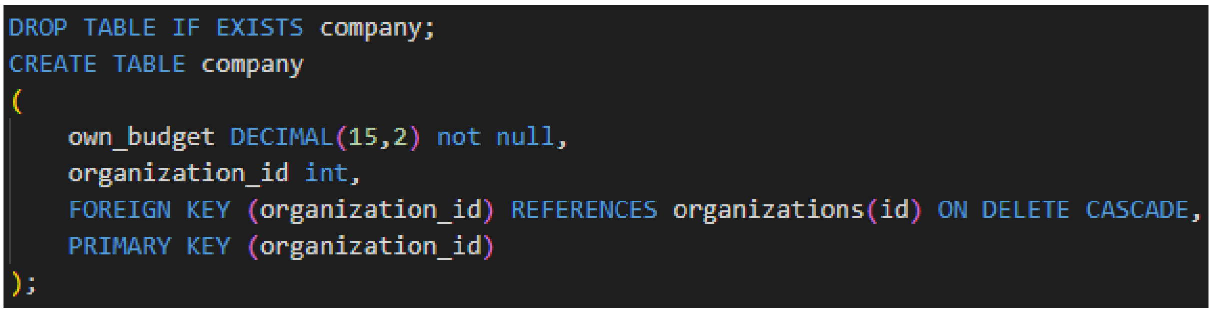
```
DROP TABLE IF EXISTS research_center;
CREATE TABLE research_center
(
    education_ministry_budget DECIMAL(15,2) not null,
    private_actions_budget DECIMAL(15,2) not null,
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    PRIMARY KEY (organization_id)
);
```

- Query για την δημιουργία του πίνακα University:



```
DROP TABLE IF EXISTS university;
CREATE TABLE university
(
    education_ministry_budget DECIMAL(15,2) not null,
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    PRIMARY KEY (organization_id)
);
```

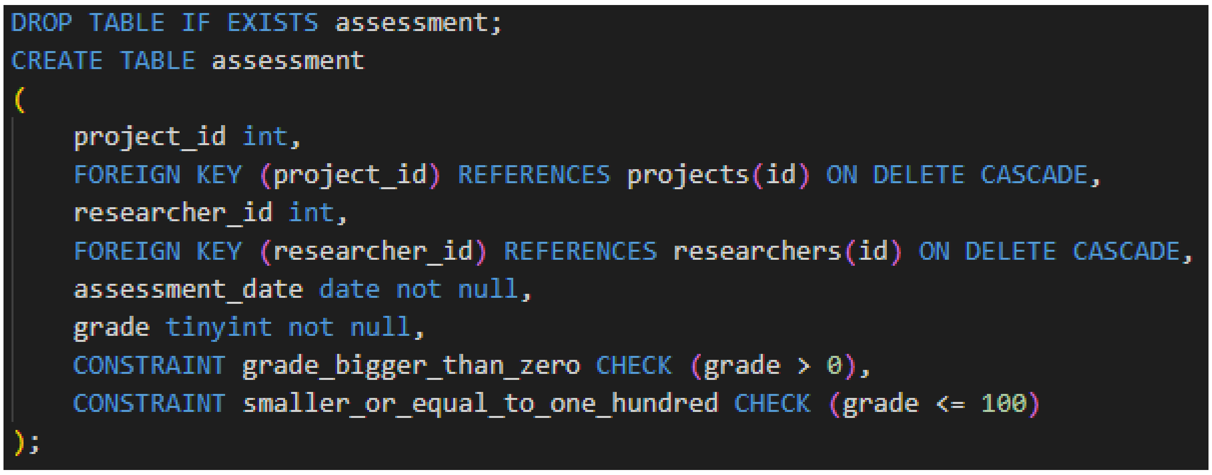
- Query για την δημιουργία του πίνακα Company:



```
DROP TABLE IF EXISTS company;
CREATE TABLE company
(
    own_budget DECIMAL(15,2) not null,
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    PRIMARY KEY (organization_id)
);
```

Δημιουργία σχέσεων μεταξύ των παραπάνω πινάκων:

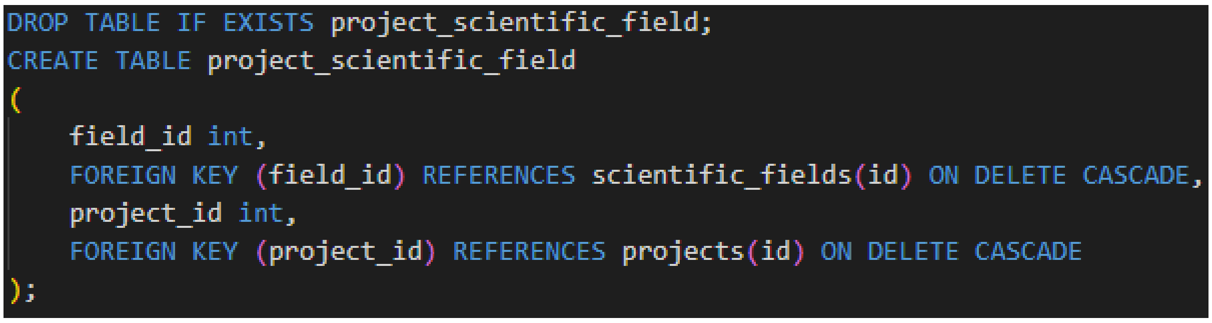
- Query για την δημιουργία του πίνακα assessment:



```
DROP TABLE IF EXISTS assessment;
CREATE TABLE assessment
(
    project_id int,
    FOREIGN KEY (project_id) REFERENCES projects(id) ON DELETE CASCADE,
    researcher_id int,
    FOREIGN KEY (researcher_id) REFERENCES researchers(id) ON DELETE CASCADE,
    assessment_date date not null,
    grade tinyint not null,
    CONSTRAINT grade_bigger_than_zero CHECK (grade > 0),
    CONSTRAINT smaller_or_equal_to_one_hundred CHECK (grade <= 100)
);
```

  Συνδέει τα έργα με τους ερευνητές που τα αξιολογούν (one to one relationship).

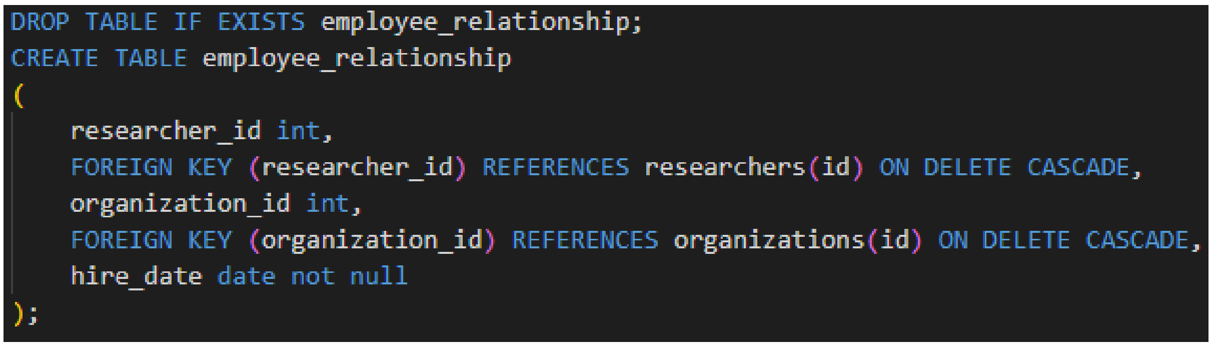
- Query για την δημιουργία του πίνακα project_scientific_field:



```
DROP TABLE IF EXISTS project_scientific_field;
CREATE TABLE project_scientific_field
(
    field_id int,
    FOREIGN KEY (field_id) REFERENCES scientific_fields(id) ON DELETE CASCADE,
    project_id int,
    FOREIGN KEY (project_id) REFERENCES projects(id) ON DELETE CASCADE
);
```

  Συνδέει τα έργα με τα επιστημονικά πεδία που καλύπτουν (many to many relationship).

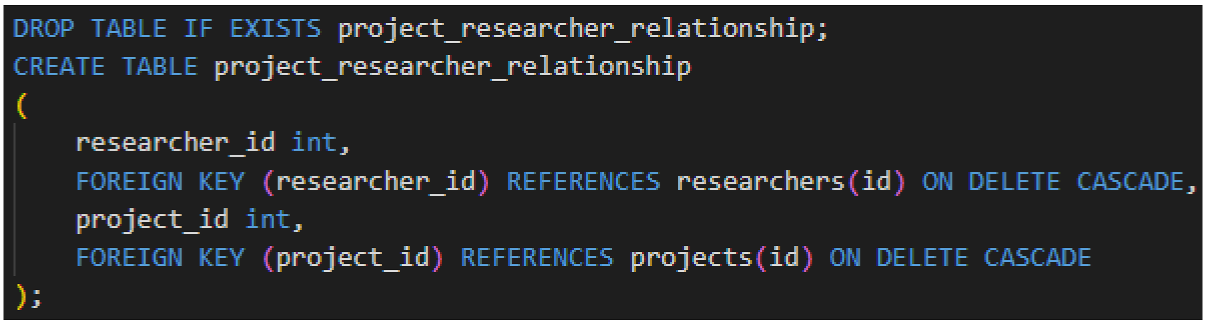
- Query για την δημιουργία του πίνακα employee_relationship:



```
DROP TABLE IF EXISTS employee_relationship;
CREATE TABLE employee_relationship
(
    researcher_id int,
    FOREIGN KEY (researcher_id) REFERENCES researchers(id) ON DELETE CASCADE,
    organization_id int,
    FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
    hire_date date not null
);
```

  Συνδέει τους οργανισμούς με τους ερευνητές που προσλαμβάνουν (many to many relationship).

- Query για την δημιουργία του πίνακα project_researcher_relationship:

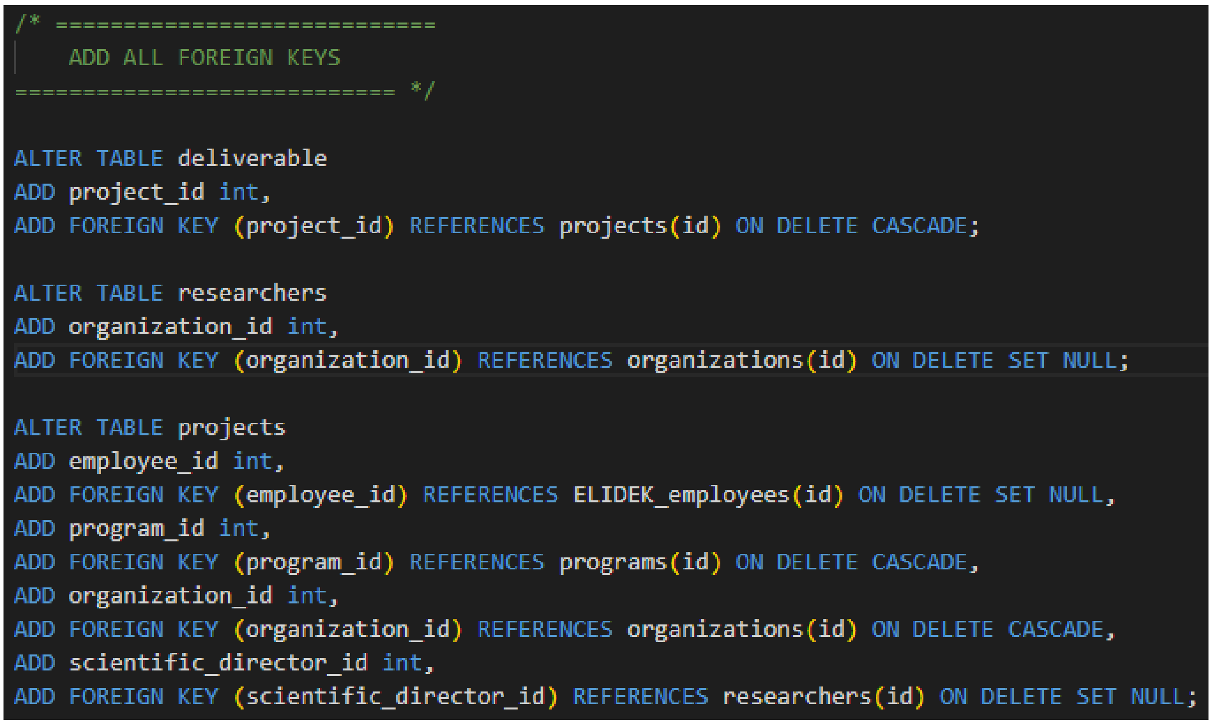


```sql
DROP TABLE IF EXISTS project_researcher_relationship;
CREATE TABLE project_researcher_relationship
(
    researcher_id int,
    FOREIGN KEY (researcher_id) REFERENCES researchers(id) ON DELETE CASCADE,
    project_id int,
    FOREIGN KEY (project_id) REFERENCES projects(id) ON DELETE CASCADE
);
```

Συνδέει τους ερευνητές με τα έργα στα οποία εργάζονται (many to many relationship).

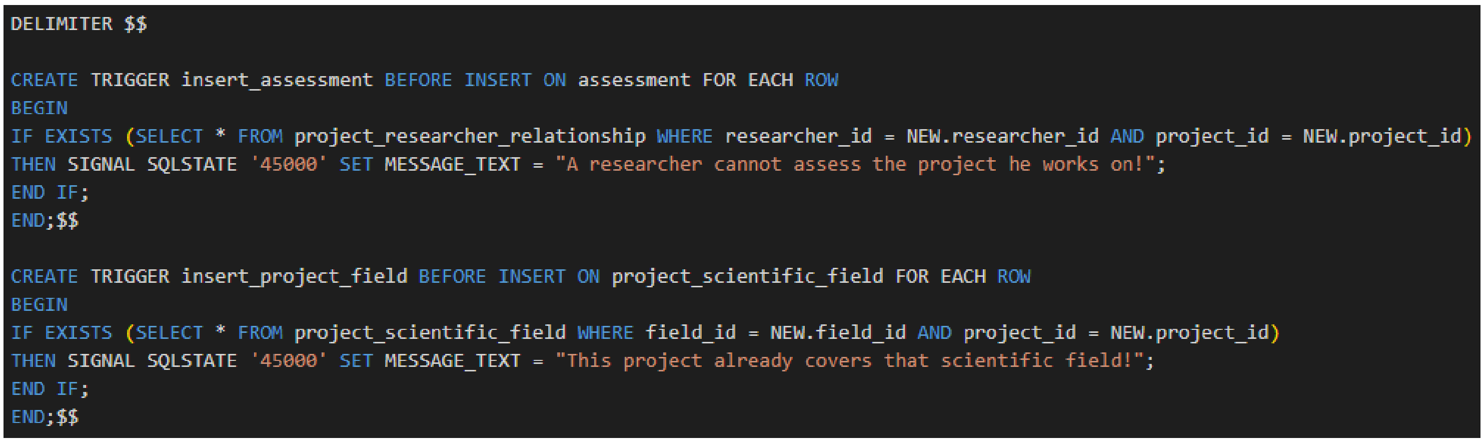
Προσθήκη foreign keys όπου θεωρούμε απαραίτητο:

- Query για την προσθήκη foreign keys:

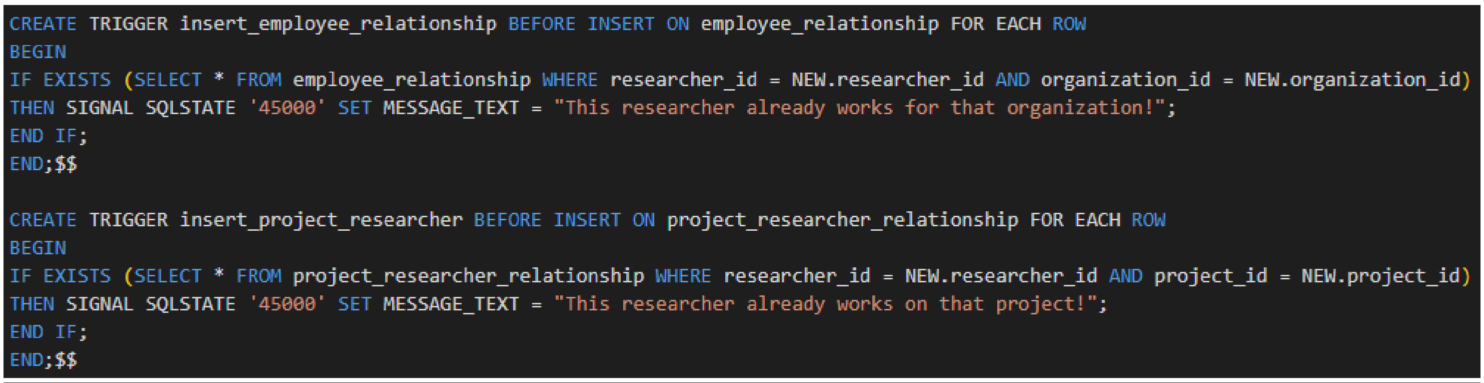


```sql
/* ============================
    ADD ALL FOREIGN KEYS
============================ */

ALTER TABLE deliverable
ADD project_id int,
ADD FOREIGN KEY (project_id) REFERENCES projects(id) ON DELETE CASCADE;

ALTER TABLE researchers
ADD organization_id int,
ADD FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE SET NULL;

ALTER TABLE projects
ADD employee_id int,
ADD FOREIGN KEY (employee_id) REFERENCES ELIDEK_employees(id) ON DELETE SET NULL,
ADD program_id int,
ADD FOREIGN KEY (program_id) REFERENCES programs(id) ON DELETE CASCADE,
ADD organization_id int,
ADD FOREIGN KEY (organization_id) REFERENCES organizations(id) ON DELETE CASCADE,
ADD scientific_director_id int,
ADD FOREIGN KEY (scientific_director_id) REFERENCES researchers(id) ON DELETE SET NULL;
```

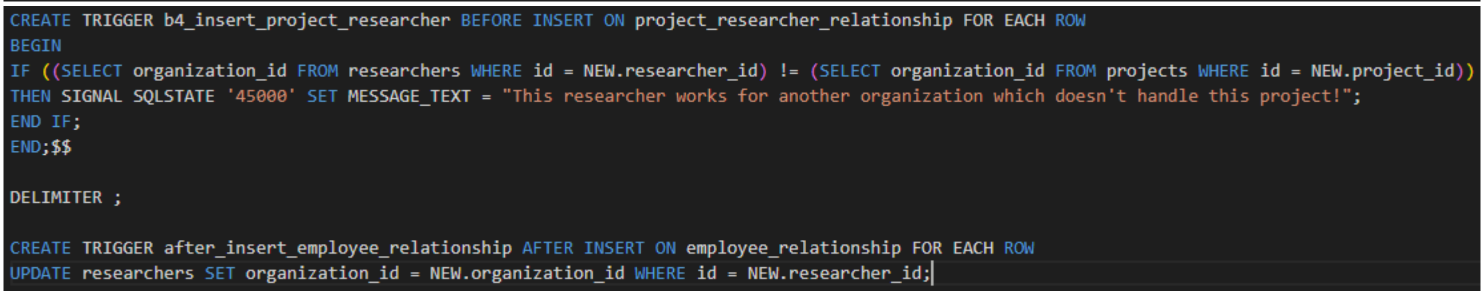
Δημιουργία Triggers για ελέγχους στις εγγραφές στην βάση. Σε περίπτωση μη επιθυμητής εγγραφής εμφανίζεται Error Message στην Mysql.



```
DELIMITER $$

CREATE TRIGGER insert_assessment BEFORE INSERT ON assessment FOR EACH ROW
BEGIN
IF EXISTS (SELECT * FROM project_researcher_relationship WHERE researcher_id = NEW.researcher_id AND project_id = NEW.project_id)
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "A researcher cannot assess the project he works on!";
END IF;
END;$$

CREATE TRIGGER insert_project_field BEFORE INSERT ON project_scientific_field FOR EACH ROW
BEGIN
IF EXISTS (SELECT * FROM project_scientific_field WHERE field_id = NEW.field_id AND project_id = NEW.project_id)
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "This project already covers that scientific field!";
END IF;
END;$$
```

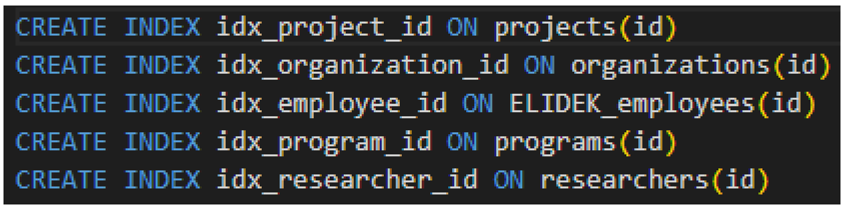


```
CREATE TRIGGER insert_employee_relationship BEFORE INSERT ON employee_relationship FOR EACH ROW
BEGIN
IF EXISTS (SELECT * FROM employee_relationship WHERE researcher_id = NEW.researcher_id AND organization_id = NEW.organization_id)
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "This researcher already works for that organization!";
END IF;
END;$$

CREATE TRIGGER insert_project_researcher BEFORE INSERT ON project_researcher_relationship FOR EACH ROW
BEGIN
IF EXISTS (SELECT * FROM project_researcher_relationship WHERE researcher_id = NEW.researcher_id AND project_id = NEW.project_id)
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "This researcher already works on that project!";
END IF;
END;$$
```



```
CREATE TRIGGER b4_insert_project_researcher BEFORE INSERT ON project_researcher_relationship FOR EACH ROW
BEGIN
IF ((SELECT organization_id FROM researchers WHERE id = NEW.researcher_id) != (SELECT organization_id FROM projects WHERE id = NEW.project_id))
THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = "This researcher works for another organization which doesn't handle this project!";
END IF;
END;$$

DELIMITER ;

CREATE TRIGGER after_insert_employee_relationship AFTER INSERT ON employee_relationship FOR EACH ROW
UPDATE researchers SET organization_id = NEW.organization_id WHERE id = NEW.researcher_id;
```

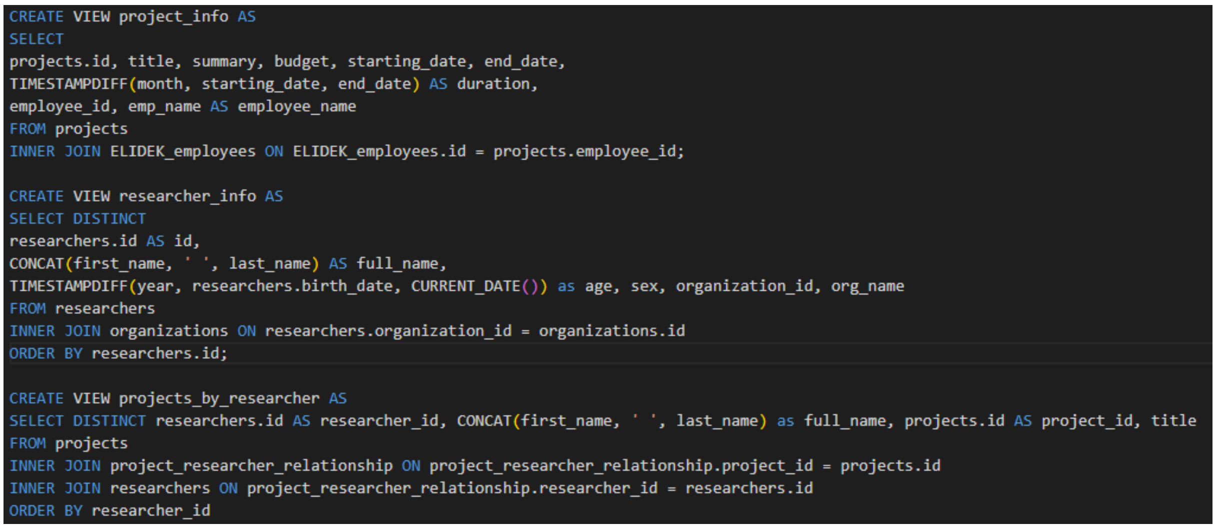
Δημιουργία Indexes:



```
CREATE INDEX idx_project_id ON projects(id)
CREATE INDEX idx_organization_id ON organizations(id)
CREATE INDEX idx_employee_id ON ELIDEK_employees(id)
CREATE INDEX idx_program_id ON programs(id)
CREATE INDEX idx_researcher_id ON researchers(id)
```

Δεν τα έχουμε υλοποιήσει στην βάση δεδομένων. Είναι χρήσιμα για βάσεις μεγαλύτερου μεγέθους, ώστε να επισπεύδουν τα συχνά queries.

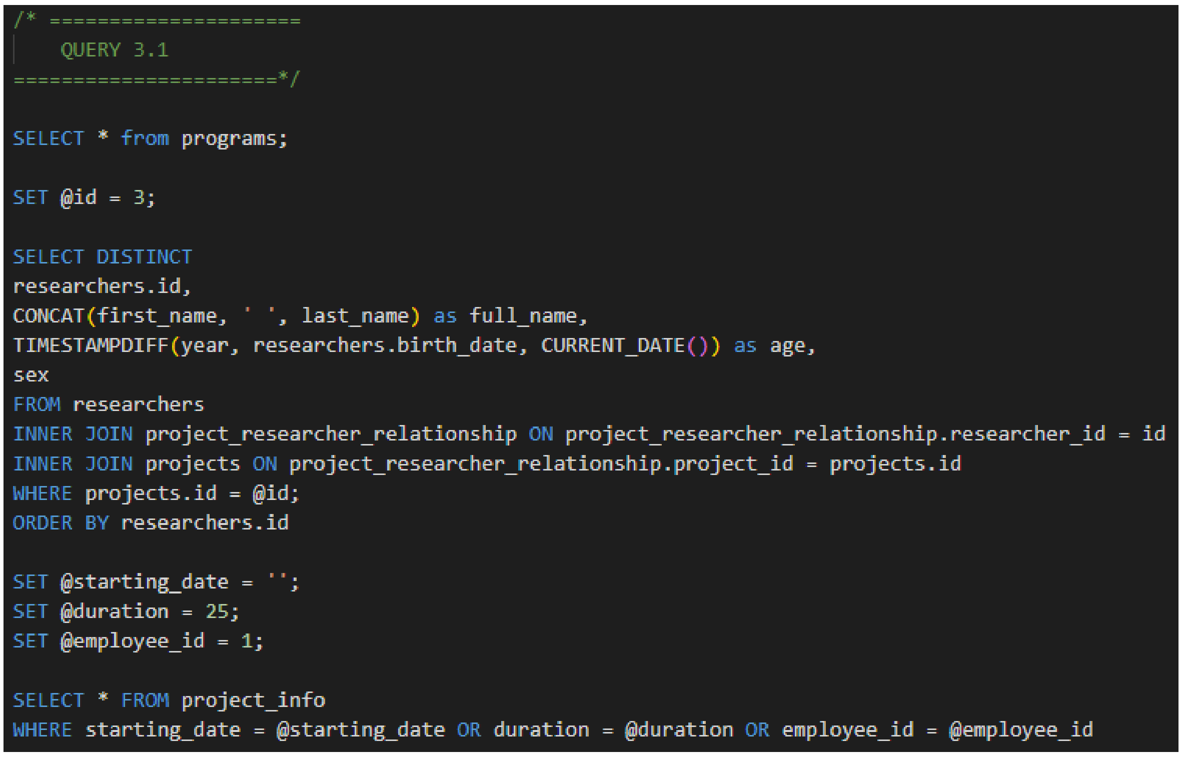
Δημιουργία Views:



```
CREATE VIEW project_info AS
SELECT
projects.id, title, summary, budget, starting_date, end_date,
TIMESTAMPDIFF(month, starting_date, end_date) AS duration,
employee_id, emp_name AS employee_name
FROM projects
INNER JOIN ELIDEK_employees ON ELIDEK_employees.id = projects.employee_id;

CREATE VIEW researcher_info AS
SELECT DISTINCT
researchers.id AS id,
CONCAT(first_name, ' ', last_name) AS full_name,
TIMESTAMPDIFF(year, researchers.birth_date, CURRENT_DATE()) as age, sex, organization_id, org_name
FROM researchers
INNER JOIN organizations ON researchers.organization_id = organizations.id
ORDER BY researchers.id;

CREATE VIEW projects_by_researcher AS
SELECT DISTINCT researchers.id AS researcher_id, CONCAT(first_name, ' ', last_name) as full_name, projects.id AS project_id, title
FROM projects
INNER JOIN project_researcher_relationship ON project_researcher_relationship.project_id = projects.id
INNER JOIN researchers ON project_researcher_relationship.researcher_id = researchers.id
ORDER BY researcher_id
```

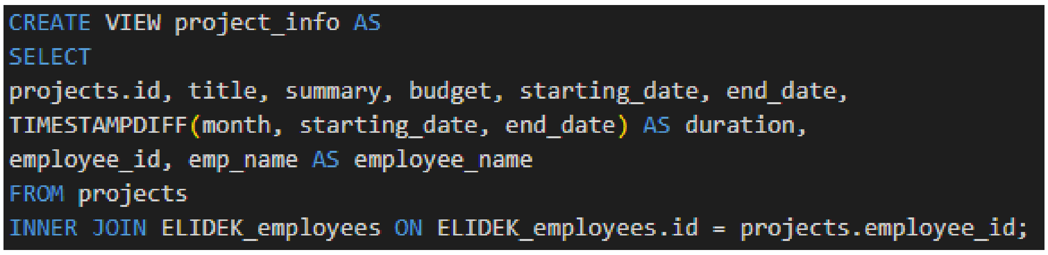
<u>DML Script</u>

3.1)



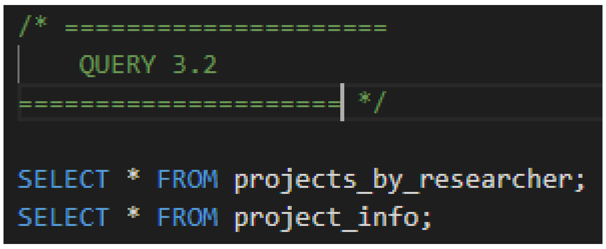
```
/* =====================
    QUERY 3.1
======================*/

SELECT * from programs;

SET @id = 3;

SELECT DISTINCT
researchers.id,
CONCAT(first_name, ' ', last_name) as full_name,
TIMESTAMPDIFF(year, researchers.birth_date, CURRENT_DATE()) as age,
sex
FROM researchers
INNER JOIN project_researcher_relationship ON project_researcher_relationship.researcher_id = id
INNER JOIN projects ON project_researcher_relationship.project_id = projects.id
WHERE projects.id = @id;
ORDER BY researchers.id

SET @starting_date = '';
SET @duration = 25;
SET @employee_id = 1;

SELECT * FROM project_info
WHERE starting_date = @starting_date OR duration = @duration OR employee_id = @employee_id
```

όπου project_info:



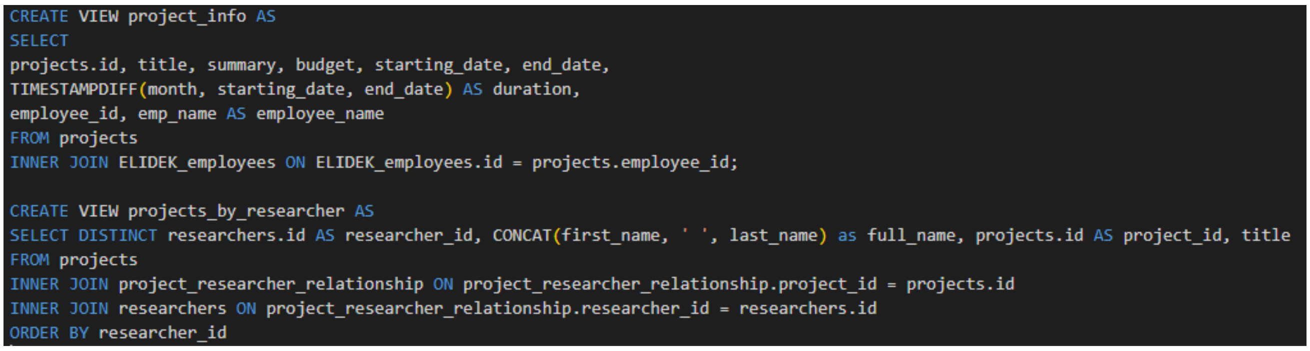
```
CREATE VIEW project_info AS
SELECT
projects.id, title, summary, budget, starting_date, end_date,
TIMESTAMPDIFF(month, starting_date, end_date) AS duration,
employee_id, emp_name AS employee_name
FROM projects
INNER JOIN ELIDEK_employees ON ELIDEK_employees.id = projects.employee_id;
```

3.2)



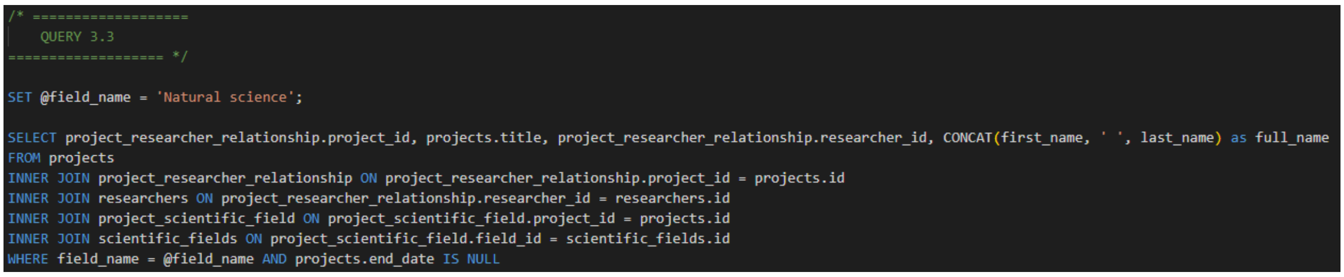
```sql
/* =====================
    QUERY 3.2
===================== */

SELECT * FROM projects_by_researcher;
SELECT * FROM project_info;
```

όπου projects_by_researcher και project_info:



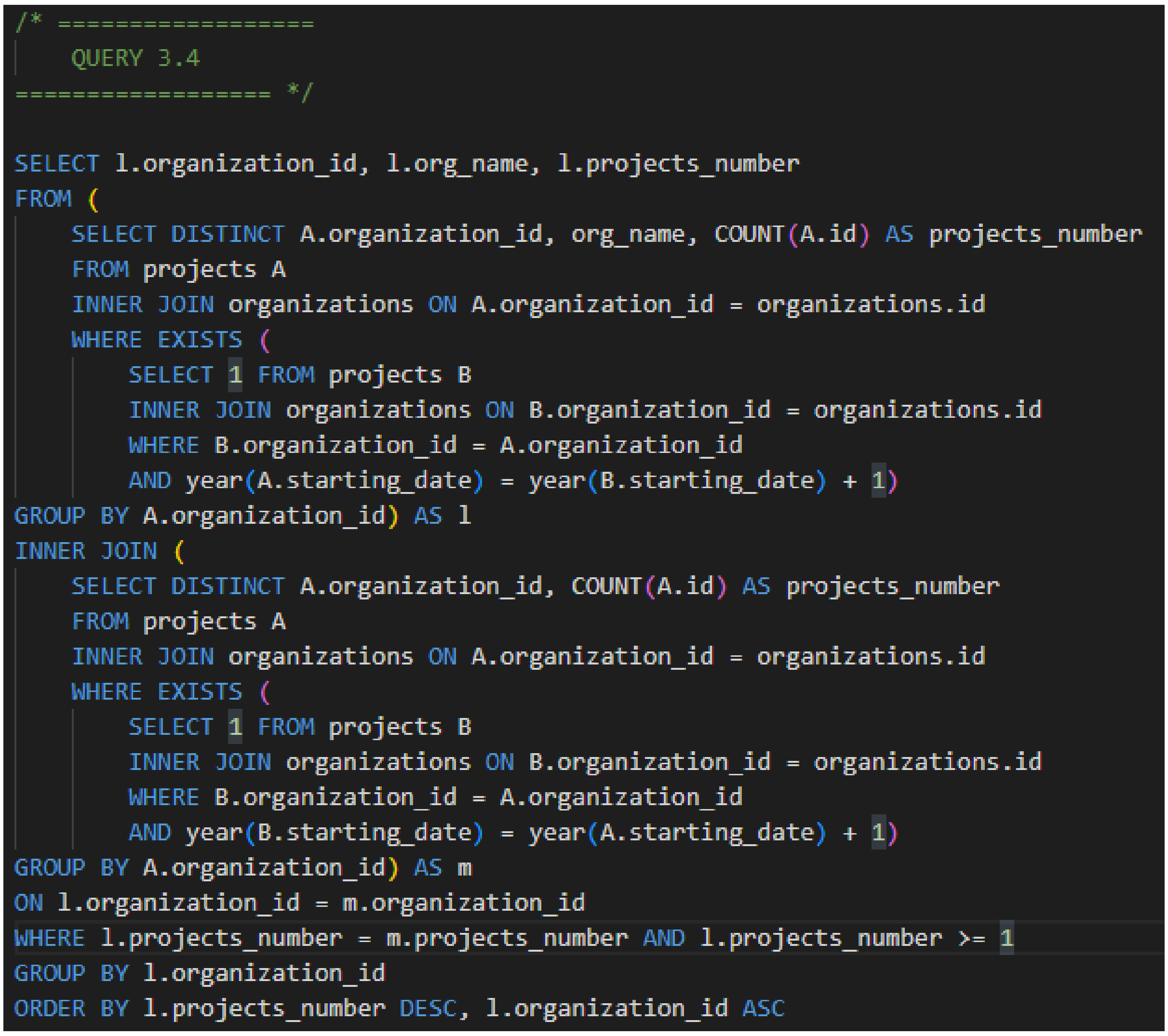
```sql
CREATE VIEW project_info AS
SELECT
projects.id, title, summary, budget, starting_date, end_date,
TIMESTAMPDIFF(month, starting_date, end_date) AS duration,
employee_id, emp_name AS employee_name
FROM projects
INNER JOIN ELIDEK_employees ON ELIDEK_employees.id = projects.employee_id;

CREATE VIEW projects_by_researcher AS
SELECT DISTINCT researchers.id AS researcher_id, CONCAT(first_name, ' ', last_name) as full_name, projects.id AS project_id, title
FROM projects
INNER JOIN project_researcher_relationship ON project_researcher_relationship.project_id = projects.id
INNER JOIN researchers ON project_researcher_relationship.researcher_id = researchers.id
ORDER BY researcher_id
```

3.3)



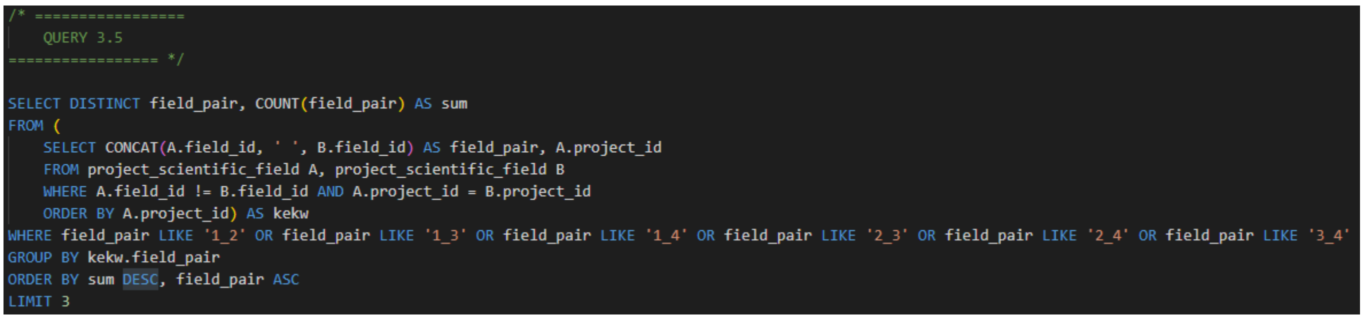
```sql
/* ==================
    QUERY 3.3
================== */

SET @field_name = 'Natural science';

SELECT project_researcher_relationship.project_id, projects.title, project_researcher_relationship.researcher_id, CONCAT(first_name, ' ', last_name) as full_name
FROM projects
INNER JOIN project_researcher_relationship ON project_researcher_relationship.project_id = projects.id
INNER JOIN researchers ON project_researcher_relationship.researcher_id = researchers.id
INNER JOIN project_scientific_field ON project_scientific_field.project_id = projects.id
INNER JOIN scientific_fields ON project_scientific_field.field_id = scientific_fields.id
WHERE field_name = @field_name AND projects.end_date IS NULL
```

3.4)



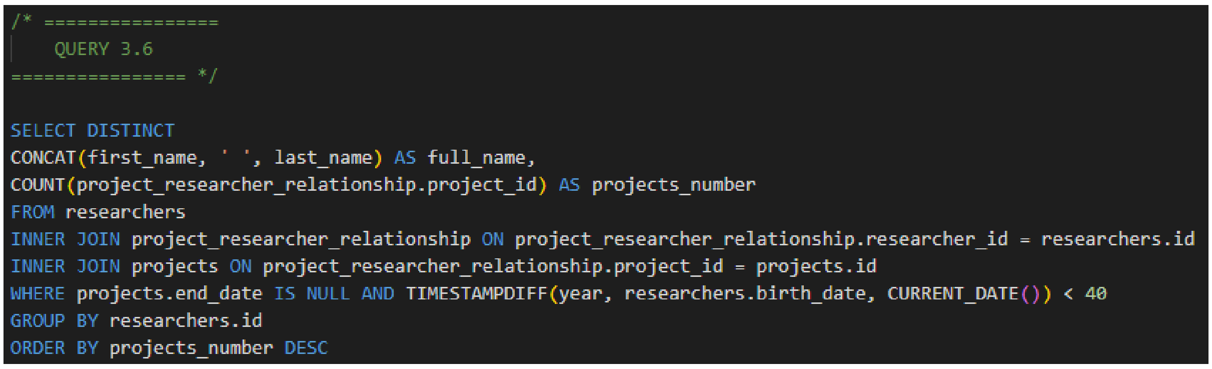
```sql
/* ==================
    QUERY 3.4
================== */

SELECT l.organization_id, l.org_name, l.projects_number
FROM (
    SELECT DISTINCT A.organization_id, org_name, COUNT(A.id) AS projects_number
    FROM projects A
    INNER JOIN organizations ON A.organization_id = organizations.id
    WHERE EXISTS (
        SELECT 1 FROM projects B
        INNER JOIN organizations ON B.organization_id = organizations.id
        WHERE B.organization_id = A.organization_id
        AND year(A.starting_date) = year(B.starting_date) + 1)
GROUP BY A.organization_id) AS l
INNER JOIN (
    SELECT DISTINCT A.organization_id, COUNT(A.id) AS projects_number
    FROM projects A
    INNER JOIN organizations ON A.organization_id = organizations.id
    WHERE EXISTS (
        SELECT 1 FROM projects B
        INNER JOIN organizations ON B.organization_id = organizations.id
        WHERE B.organization_id = A.organization_id
        AND year(B.starting_date) = year(A.starting_date) + 1)
GROUP BY A.organization_id) AS m
ON l.organization_id = m.organization_id
WHERE l.projects_number = m.projects_number AND l.projects_number >= 1
GROUP BY l.organization_id
ORDER BY l.projects_number DESC, l.organization_id ASC
```

3.5)



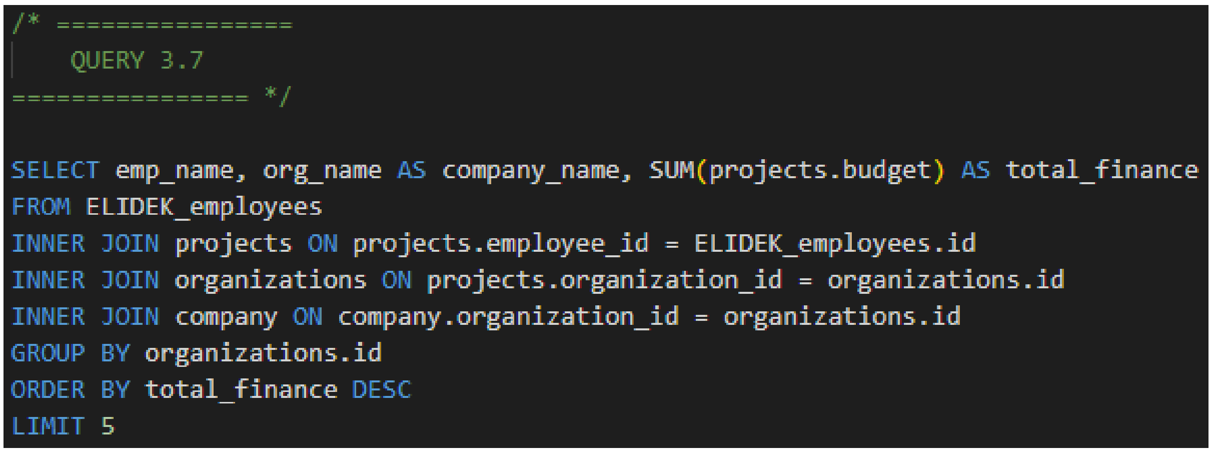
```sql
/* ================
    QUERY 3.5
================ */

SELECT DISTINCT field_pair, COUNT(field_pair) AS sum
FROM (
    SELECT CONCAT(A.field_id, ' ', B.field_id) AS field_pair, A.project_id
    FROM project_scientific_field A, project_scientific_field B
    WHERE A.field_id != B.field_id AND A.project_id = B.project_id
    ORDER BY A.project_id) AS kekw
WHERE field_pair LIKE '1_2' OR field_pair LIKE '1_3' OR field_pair LIKE '1_4' OR field_pair LIKE '2_3' OR field_pair LIKE '2_4' OR field_pair LIKE '3_4'
GROUP BY kekw.field_pair
ORDER BY sum DESC, field_pair ASC
LIMIT 3
```

3.6)



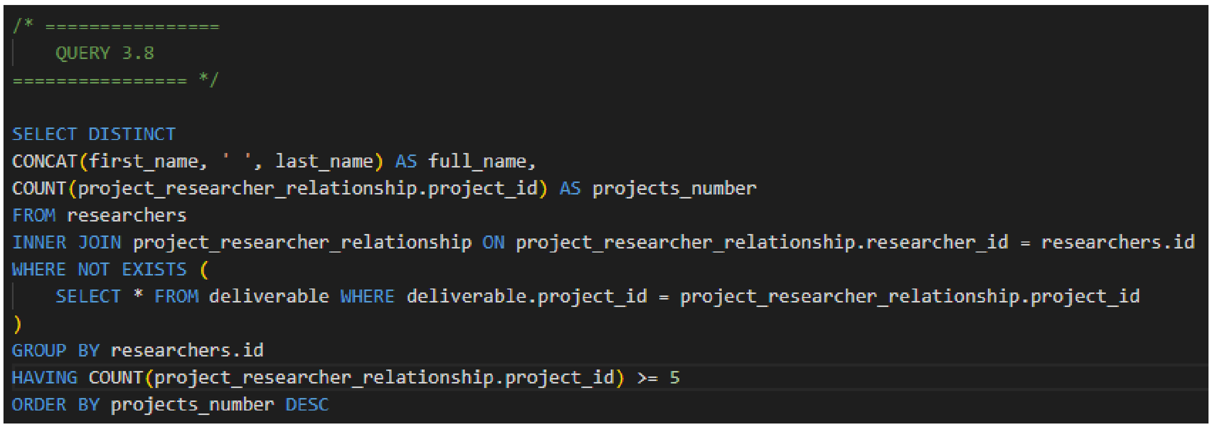
```
/* ================
    QUERY 3.6
================ */

SELECT DISTINCT
CONCAT(first_name, ' ', last_name) AS full_name,
COUNT(project_researcher_relationship.project_id) AS projects_number
FROM researchers
INNER JOIN project_researcher_relationship ON project_researcher_relationship.researcher_id = researchers.id
INNER JOIN projects ON project_researcher_relationship.project_id = projects.id
WHERE projects.end_date IS NULL AND TIMESTAMPDIFF(year, researchers.birth_date, CURRENT_DATE()) < 40
GROUP BY researchers.id
ORDER BY projects_number DESC
```

3.7)



```
/* ================
    QUERY 3.7
================ */

SELECT emp_name, org_name AS company_name, SUM(projects.budget) AS total_finance
FROM ELIDEK_employees
INNER JOIN projects ON projects.employee_id = ELIDEK_employees.id
INNER JOIN organizations ON projects.organization_id = organizations.id
INNER JOIN company ON company.organization_id = organizations.id
GROUP BY organizations.id
ORDER BY total_finance DESC
LIMIT 5
```

3.8)



```
/* ================
    QUERY 3.8
================ */

SELECT DISTINCT
CONCAT(first_name, ' ', last_name) AS full_name,
COUNT(project_researcher_relationship.project_id) AS projects_number
FROM researchers
INNER JOIN project_researcher_relationship ON project_researcher_relationship.researcher_id = researchers.id
WHERE NOT EXISTS (
    SELECT * FROM deliverable WHERE deliverable.project_id = project_researcher_relationship.project_id
)
GROUP BY researchers.id
HAVING COUNT(project_researcher_relationship.project_id) >= 5
ORDER BY projects_number DESC
```

Οδηγίες εγκατάστασης:

1) Εγκατάσταση Mysql. Εμείς χρησιμοποιήσαμε XAMPP.
2) Εγκατάσταση NodeJS.
3) Για την NodeJS χρειαζόμαστε express, mysql2, ejs, faker, express-session, connect-flash, nodemon, chalk και custom-env dependencies τα οποία εγκαθιστούμε με την εντολή *npm i [package-name]*.
4) Στο app.js αρχείο χρησιμοποιούμε *require('custom-env').env('localhost');* και δημιουργούμε αρχείο *.env.localhost* με περιεχόμενο:

    ```
    SERVER_PORT=5000
    DB_HOST=localhost
    DB_PORT=3306
    DB_USER=db-user
    DB_PASS=db-pass
    DB=db-name
    ```
5) Για την δημιουργία των δεδομένων της βάσης τρέχουμε *npm run create-data.* Στον φάκελο *dummyDataCreator* δημιουργούνται .txt αρχεία με τα δεδομένα. Στο αρχείο db-starter επεξεργαζόμαστε κατάλληλα τα file paths π.χ. *C:\Users\myUser\Desktop\Databases-NodeJS\create_tables.sql*
6) Στην Mysql τρέχουμε *source C:\myPath\Databases-NodeJS\db_starter.txt* ώστε να δημιουργήσουμε και να γεμίσουμε με τυχαία δεδομένα την βάση.
7) Τέλος, τρέχουμε *npm start* στην NodeJS και σε ενα browser ανοίγουμε http://localhost:myPort. Εμείς χρησιμοποιούμε http://localhost:5000.

Αναλυτικά βήματα εγκατάστασης στο github:
https://github.com/LefterisTournaris/Databases-NodeJS