

ΕΡΓΑΣΤΗΡΙΟ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ: PROJECT 2020-21

ΚΡΕΜΑΝΤΑΛΑ ΘΕΟΔΩΡΑ ΑΜ: 1067445

ΜΠΑΡΜΠΑΡΗ ΑΡΙΑΔΝΗ ΑΜ: 1067379

Μέρος Α: ΒΔ και SQL

Σύμφωνα με την περιγραφή και τις επιπλέον λειτουργικές απαιτήσεις που μας δίνονται στην εκφώνηση προκύπτει η εξής βάση δεδομένων:

```
CREATE DATABASE staffevaluation DEFAULT CHARSET=greek;
```

```
USE staffevaluation;
```

```
CREATE TABLE user (
```

```
    username varchar(12) NOT NULL,
```

```
    password varchar(10) DEFAULT NULL,
```

```
    name varchar(25) NOT NULL DEFAULT 'unknown',
```

```
    surname varchar(35) NOT NULL DEFAULT 'unknown',
```

```
    reg_date datetime DEFAULT NULL,
```

```
    email varchar(30) NOT NULL,
```

```
    PRIMARY KEY (username)
```

```
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE company (
```

```
    AFM char(9) NOT NULL,
```

```
    DOY varchar(15) DEFAULT NULL,
```

```
    name varchar(35) NOT NULL DEFAULT 'unknown',
```

```
    phone bigint(16) NOT NULL,
```

```
    street varchar(15) NOT NULL,
```

```
    num tinyint(4) DEFAULT NULL,
```

```
city varchar(15) NOT NULL DEFAULT 'unknown',  
country varchar(15) NOT NULL DEFAULT 'unknown',  
PRIMARY KEY (AFM)  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE manager (  
managerUsername varchar(12) NOT NULL,  
exp_years tinyint(4) DEFAULT NULL,  
firm char(9) NOT NULL,  
PRIMARY KEY (managerUsername),  
CONSTRAINT MANAGERCMP FOREIGN KEY (firm) REFERENCES company (AFM) ON  
DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT MANAGERUSR FOREIGN KEY (managerUsername) REFERENCES user  
(username) ON DELETE CASCADE ON UPDATE CASCADE  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE evaluator (  
username varchar(12) NOT NULL,  
exp_years tinyint(4) DEFAULT NULL,  
firm char(9) NOT NULL,  
PRIMARY KEY (username),  
CONSTRAINT EVALCMP FOREIGN KEY (firm) REFERENCES company (AFM) ON  
DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT EVALUSR FOREIGN KEY (username) REFERENCES user (username) ON  
DELETE CASCADE ON UPDATE CASCADE  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE employee (  
username varchar(12) NOT NULL,
```

```

exp_years tinyint(4) DEFAULT NULL,
bio text DEFAULT NULL,
sistatikes varchar(35) NOT NULL,
certificates varchar(35) NOT NULL,
awards varchar(35) NOT NULL,
firm char(9) NOT NULL,
PRIMARY KEY (username),
CONSTRAINT EMPLUSR FOREIGN KEY (username) REFERENCES user (username) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT EMPLCOMP FOREIGN KEY (firm) REFERENCES company (AFM) ON
DELETE CASCADE ON UPDATE CASCADE
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

```

CREATE TABLE administrator (
username varchar(12) NOT NULL,
PRIMARY KEY (username),
CONSTRAINT ADMINUSR FOREIGN KEY (username) REFERENCES user (username)
ON DELETE CASCADE ON UPDATE CASCADE
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

```

CREATE TABLE job (
id int(4) NOT NULL,
start_date date NOT NULL,
salary float(6,1) DEFAULT NULL,
position varchar(40) DEFAULT NULL,
edra varchar(45) DEFAULT NULL,
evaluator varchar(12) NOT NULL,
announce_date datetime DEFAULT current_timestamp(),
submission_date date NOT NULL,

```

```
PRIMARY KEY (id),  
  
CONSTRAINT JOBEVLTR FOREIGN KEY (evaluator) REFERENCES evaluator  
(username) ON DELETE CASCADE ON UPDATE CASCADE  
  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE project (  
  
    empl varchar(12) NOT NULL,  
  
    num tinyint(4) NOT NULL ,  
  
    descr text NOT NULL,  
  
    url varchar(60) DEFAULT NULL,  
  
    PRIMARY KEY (empl,num),  
  
    CONSTRAINT PROJEMPL FOREIGN KEY (empl) REFERENCES employee (username)  
ON DELETE CASCADE ON UPDATE CASCADE  
  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE degree (  
  
    titlos varchar(50) NOT NULL,  
  
    idryma varchar(40) NOT NULL,  
  
    bathmida enum('LYKEIO','UNIV','MASTER','PHD') DEFAULT NULL,  
  
    PRIMARY KEY (titlos,idryma)  
  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

```
CREATE TABLE has_degree (  
  
    degr_title varchar(50) NOT NULL,  
  
    degr_idryma varchar(40) NOT NULL,  
  
    empl_username varchar(12) NOT NULL,  
  
    etos year(4) DEFAULT NULL,  
  
    grade float(3,1) DEFAULT NULL,  
  
    PRIMARY KEY (degr_title,degr_idryma,empl_username),
```

CONSTRAINT HASDEGR FOREIGN KEY (degr_title, degr_idryma) REFERENCES degree (titlos, idryma) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT HASEMPL FOREIGN KEY (empl_username) REFERENCES employee (username) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

CREATE TABLE **antikeim** (

title varchar(36) NOT NULL,

descr tinytext DEFAULT NULL,

belongs_to varchar(36) DEFAULT NULL,

PRIMARY KEY (title),

CONSTRAINT ANTIKEIM FOREIGN KEY (belongs_to) REFERENCES antikeim (title) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

CREATE TABLE **needs** (

job_id int(4) NOT NULL,

antikeim_title varchar(36) NOT NULL,

PRIMARY KEY (job_id,antikeim_title),

CONSTRAINT JOBNEEDS FOREIGN KEY (job_id) REFERENCES job (id) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT NEEDSANTIK FOREIGN KEY (antikeim_title) REFERENCES antikeim (title) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

CREATE TABLE **requestsevaluation** (

empl_username varchar(12) NOT NULL,

```

job_id int(4) NOT NULL,

PRIMARY KEY (empl_username,job_id),

CONSTRAINT EVAEMPLOY FOREIGN KEY (empl_username) REFERENCES employee
(username) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT EVALJ FOREIGN KEY (job_id) REFERENCES job (id) ON DELETE CASCADE
ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

```

CREATE TABLE languages (

employee varchar(12) NOT NULL,

lang set('EN','FR','SP','GR') NOT NULL,

PRIMARY KEY (employee,lang),

CONSTRAINT EMPLLANG FOREIGN KEY (employee) REFERENCES employee
(username) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

```

CREATE TABLE evaluationresult (

Evid INT(4) NOT NULL ,

empl_username varchar(12) NOT NULL,

job_id int(4) NOT NULL,

grade INT(4) NOT NULL ,

comments varchar(255) NOT NULL,

PRIMARY KEY(Evid,empl_username),

CONSTRAINT EMPL_EVALR FOREIGN KEY (empl_username) REFERENCES employee
(username) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT JOB_EVALR FOREIGN KEY (job_id) REFERENCES job (id) ON DELETE
CASCADE ON UPDATE CASCADE

)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

Ακολουθούν οι επεκτάσεις της βάσης δεδομένων που χρειάζονται για το πρώτο μέρος του πρότζεκτ.

1) Διαδικασία Αξιολόγησης

Δημιουργούμε έναν νέο πίνακα με όνομα **evaluation** στον οποίο θα αποθηκεύονται οι 3 επιμέρους αξιολογήσεις :

α) **Συνέντευξη** που δίνει ο υπάλληλος στον αξιολογητή (0-4),

β) **Αξιολόγηση** που γίνεται με βάση το **report** του διευθυντή του τμήματος στο οποίο ανήκει ο εργαζόμενος (0-4)

γ) **Αξιολόγηση** των **πτυχίων** (0-2)

Στην βάση μας η α) αξιολόγηση για την συνέντευξη αποθηκεύεται στο πεδίο **aksiologisi1** του πίνακα evaluation , η β) αξιολόγηση του report αποθηκεύεται στο πεδίο **aksiologisi2** του πίνακα και η γ) αξιολόγηση των πτυχίων αποθηκεύεται στο πεδίο **aksiologisi3** του πίνακα evaluation.

```
CREATE TABLE evaluation (  
    emplusername varchar(12) NOT NULL,  
    evalusername varchar(12) NOT NULL,  
    jobid int(4) NOT NULL,  
    aksiologisi1 enum('0','1','2','3','4') DEFAULT NULL,  
    aksiologisi2 enum('0','1','2','3','4') DEFAULT NULL,  
    aksiologisi3 enum('0','1','2') DEFAULT NULL,  
    comments varchar(255) NOT NULL,  
    eval_id INT(4) NOT NULL,  
    CONSTRAINT EVL_EMP FOREIGN KEY (emplusername) REFERENCES  
employee (username) ON DELETE CASCADE ON UPDATE CASCADE,  
    CONSTRAINT EVAL_EVL FOREIGN KEY (evalusername) REFERENCES  
evaluator (username) ON DELETE CASCADE ON UPDATE CASCADE,  
    CONSTRAINT EVAL_JOB FOREIGN KEY (jobid) REFERENCES job (id) ON  
DELETE CASCADE ON UPDATE CASCADE  
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;
```

Για να υπολογίζεται **αυτόματα** το τελικό αποτέλεσμα αξιολόγησης και να αποθηκεύεται στον πίνακα evaluation result μετά την καταχώρηση των βαθμολογιών του αξιολογητή και στις 3 επιμέρους φάσεις υλοποιήσαμε έναν **trigger** ο οποίος φαίνεται παρακάτω:

Ο Trigger υπολογίζει το άθροισμα των επιμέρους αξιολογήσεων κάθε φορά που θα εισάγεται ένας βαθμός και θα το καταχωρεί στο πεδίο Trade του πίνακα **evaluationresult**.

Στον τελικό πίνακα **evaluationresult** δεν συμπεριλαμβάνονται οι ενδιάμεσες αξιολογήσεις που δεν έχουν ολοκληρωθεί (δηλαδή έχουν τιμή NULL)

DELIMITER \$

```
CREATE TRIGGER SumOfEvaluations
AFTER INSERT ON evaluation
FOR EACH ROW
BEGIN
    SET @eval_1 = NEW.aksiologisi1;
    SET @eval_2 = NEW.aksiologisi2;
    SET @eval_3 = NEW.aksiologisi3;

    IF(@eval_1 <> " AND @eval_2 <> " AND @eval_3 <> ") THEN

        SET @sum = @eval_1 + @eval_2 + @eval_3;

        INSERT INTO evaluationresult
        VALUES
        (NEW.eval_id,NEW.emplusername,NEW.jobid,@sum,NEW.comments);
    END IF;

END $
DELIMITER;
```

2) Διαδικασία υποβολής αιτήσεων

Για την συγκεκριμένη διαδικασία δημιουργήσαμε δυο νέους πίνακες με ονόματα **promotion** και **requestspromotion**.

Στον πρώτο αποθηκεύονται οι θέσεις εργασίας που ανακοινώνονται για προαγωγή από τον manager και τον evaluator της κάθε εταιρείας . Ενώ στον δεύτερο πίνακα αποθηκεύονται οι αιτήσεις των εργαζομένων που ενδιαφέρονται για τις θέσεις που είναι σε προαγωγή.


```

CREATE TABLE promotion (
manag_username varchar(12) NOT NULL,
evaluat_username varchar(12) NOT NULL,
id_job INT(4) NOT NULL,
PRIMARY KEY(manag_username,evaluat_username,id_job),
CONSTRAINT PROM_MANAG FOREIGN KEY (manag_username) REFERENCES
manager(managerUsername) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT PROM_EVAL FOREIGN KEY (evaluat_username) REFERENCES
evaluator(username) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT PROM_JO FOREIGN KEY (id_job) REFERENCES job(id) ON DELETE
CASCADE ON UPDATE CASCADE
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

```

CREATE TABLE requestspromotion (
emplo_username varchar(12) NOT NULL,
jobID int NOT NULL,
PRIMARY KEY (emplo_username,jobID),
CONSTRAINT EVAEMPLO FOREIGN KEY (emplo_username) REFERENCES
employee (username) ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT EVALJOB FOREIGN KEY (jobID) REFERENCES job (id) ON DELETE
CASCADE ON UPDATE CASCADE
) ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

Κάθε φορά που κάποιος εργαζόμενος κάνει αίτηση για μια θέση εργασίας θα πρέπει να ελέγχεται αν αυτή η εργασία βρίσκεται σε προαγωγή. Αυτό το πετυχαίνουμε με την χρήση ενός **trigger** .
Αν αυτή η θέση για την οποία ενδιαφέρεται ο εργαζόμενος δεν βρίσκεται σε προαγωγή θα εμφανίζεται το μήνυμα : 'There is no promotion for this job.';

```

DELIMITER $

```

```

CREATE TRIGGER applyingforjob_promotion

```

```

BEFORE INSERT ON requestspromotion

```

```

FOR EACH ROW

```

```

BEGIN

```

```

                IF NEW.job_id NOT IN (SELECT id_job FROM promotion WHERE
id_job=NEW.job_id)

```

```

                THEN

```

```

SIGNAL SQLSTATE VALUE '45000'

    SET MESSAGE_TEXT = 'There is no promotion for this job.';

END IF;

END $

DELIMITER ;

```

3) Διατήρηση Πίνακα Ενεργειών (log)

```

CREATE TABLE log (
username_xrhsth varchar(12) DEFAULT NULL,
Date_time datetime DEFAULT NULL,
ektelesi enum('YES','NO') NOT NULL,
typeofaction varchar(10) NOT NULL,
table_name varchar(25) NOT NULL,
CONSTRAINT LOG_USER FOREIGN KEY (username_xrhsth) REFERENCES user
(username) ON DELETE CASCADE ON UPDATE CASCADE
)ENGINE = InnoDB CHARACTER SET greek COLLATE greek_general_ci;

```

Στον πίνακα log θα αποθηκεύονται όσα ζητούνται στην εκφώνηση.
Όσον αφορά το username του χρήστη κάθε φορά που θα συνδέεται στην αρχική σελίδα σύνδεσης του Β μέρους κάποιος χρήστης το username του θα κρατείται ώστε να αποθηκευτεί στον πίνακα log.

Ακολουθούν τα ερωτήματα που ζητούνται να υλοποιηθούν στο πρώτο μέρος :

- 1) Εκτός από τα δεδομένα και τις απαιτήσεις που μας δίνονται στις εκφωνήσεις για την βάση δεδομένων κάναμε και τις **εξής παραδοχές** για να υλοποιήσουμε την βάση δεδομένων μας:
 - Ο evaluator που ανακοινώνει μια θέση εργασίας , την ανακοινώνει και για προαγωγή (στον πίνακα promotion).
 - Όλες οι θέσεις εργασίας υπάρχουν σε όλες τις εταιρείες-companies, δηλαδή είναι ιδίες για όλες τις εταιρείες.
 - Οι προαγωγές ανακοινώνονται τόσο από τον υπεύθυνο αξιολόγησης όσο και από τον διευθυντή – manager ο οποίος αποφασίζει για το ποιες θα πρέπει να προκηρυχθούν για προαγωγή (πίνακας promotion).

Το σχεσιακό διάγραμμα για την συνολική αναθεωρημένη βάση δεδομένων μετά τις επεκτάσεις είναι το ακόλουθο:

//ADD ADMINISTRATORS

INSERT INTO user VALUES

```
('xristanag','764','Xristina','Anagnwstopoulou','2011-09-01
10:00:00','anagnos@hotmail.gr'),
('mairypap','646','Mairy','Papas','2006-12-05 14:05:00','mapapado@yahoo.com'),
('iosifzax','543','Iosif','Zaxariou','2007-01-08 11:00:00','zaxar@mail.com'),
('marzax','348','Maria','Zaxariou','2011-02-20 17:00:00','zaxarm@mail.com');
```

//ADD EMPLOYEES

INSERT INTO user VALUES

```
('mnikol','m@n0lis','Manolis','Nikopoloulos','2017-11-08 21:07:12',
'nikolp@gmail.com'),
('abrown','w1lcoxon','Andrew','McBrown','2018-01-27 16:02:56',
'andrewbr@yahoo.com'),
('elenineo','369','Eleni','Neofytou','2016-01-06 14:50:00','neofytoue@gmail.com'),
('kwstasneo','2149','Neofytos','Kwstas','2005-03-14 16:10:00','neofytos@gmail.com'),
('cleogeo','upL34r','Cleomenis','Georgiadis','2018-02-13 12:23:34',
'cleom17@gmail.com'),
('zazahir23','zoolhger','Ahmet','Mobasher-Hirs','2017-05-11 14:08:23',
'ahmetTech@yahoo.com'),
('lionarF','erg2378','Freddy','Lionar','2018-10-07 20:09:10','Lionarfre@ezra.co.uk'),
('liagourma','sionpass','Maria','Liagkoumi','2018-05-22 17:03:01',
'mliagkr@gmail.com'),
('msmith','lol123','Mike','Smith','2014-02-01 17:00:00','msmith@hotmail.com'),
('jennyk','k555','Jenny','Kiriakou','2013-03-20 20:21:00','jenkir@gmail.com'),
('miltH','milt095','Miltiadis','Hristidis','2019-07-22 16:51:32','miltiadis@yahoo.com');
```

//ADD EVALUATORS

INSERT INTO user VALUES

```
('hlgeor','hlias90','Hlias','Georgiadis','2003-02-12 12:00:21','hl20@mail.com'),
('vlasster','sterr','Vlassis','Stergiou','2005-12-25 14:20:00','stervl@gmail.gr'),
('akisgoud','we341','Akis','Goudis','2008-06-17 13:15:00','ak@hotmail.com'),
('efstang','ef38','Efstathios','Anagnwstakis','2006-05-14 15:20:59','anagef@yahoo.gr'),
('adonkab','kab78','Adonis','Kabouris','2010-01-01 10:16:20','adon@mail.gr'),
('dionkok','67kok','Dionisios','Kokkinakis','2000-02-12
16:00:41','dionk@hotmail.com'),
('swtirzan','swt235','Swtiris','Zanos','2007-03-12 18:05:30','swtiris@yahoo.com');
```

INSERT INTO company VALUES

('023453344', 'C Patras', 'EXPENDITURE Ltd', 2610256321, 'Maizonos', 123, 'Patra', 'Greece'),
('023451232', 'A Patras', 'Typology Ltd', 2610231452, 'Korinthou', 56, 'Patra', 'Greece'),
('123432211', 'A Geraka', 'SoftSol A.E.', 2103452133, 'Ahepa', 44, 'Athina', 'Greece'),
('18765549', 'C Peiraia', 'Unigram', 2103452672, 'Karaiskaki', 10, 'Peiraias', 'Greece'),
('561234561', 'GS 35321 L', 'InCodeWeTrust', 1242345612, 'Oxford', 12, 'London', 'United Kingdom'),
('23122345', 'SF 1234 BG', 'SocialSc', 3200123451, 'General Sklevi', 35, 'Sofia', 'Bulgaria'),
('05694712', 'A Athinwn', 'Electrical A.A.', 2100202022, 'Kifisias', 26, 'Athina', 'Greece');

INSERT INTO manager VALUES

('nikospap', 2, '023453344'),
('panxrist', 5, '023451232'),
('Theoxrist', 1, '123432211'),
('nikipap', 6, '18765549'),
('vasmitr', 3, '561234561'),
('stavroskost', 9, '23122345'),
('ptolkor', 5, '05694712');

INSERT INTO evaluator VALUES

('hlgeor', 10, '18765549'),
('vlasster', 8, '023451232'),
('akisgoud', 11, '123432211'),
('efstang', 3, '18765549'),
('adonkab', 5, '561234561'),
('dionkok', 1, '23122345'),
('swtirzan', 7, '05694712');

INSERT INTO employee VALUES

('mnikol', 3, 'bio1', 'sistatikes1', 'certificates1', 'awards1', '023453344'),
('abrown', 4, 'bio2', 'sistatikes2', 'certificates2', 'awards2', '023451232'),
('elenineo', 2, 'bio3', 'sistatikes3', 'certificates3', 'awards3', '123432211'),
('kwstasneo', 1, 'bio4', 'sistatikes4', 'certificates4', 'awards4', '18765549'),
('cleogeo', 6, 'bio5', 'sistatikes5', 'certificates5', 'awards5', '561234561'),
('zazahir23', 1, 'bio6', 'sistatikes6', 'certificates6', 'awards6', '23122345'),
('lionarF', 8, 'bio7', 'sistatikes7', 'certificates7', 'awards7', '05694712'),
('msmith', 12, 'bio8', 'sistatikes8', 'certificates8', 'awards8', '023453344'),
('jennyk', 2, 'bio9', 'sistatikes9', 'certificates9', 'awards9', '023451232'),
('miltH', 9, 'bio10', 'sistatikes10', 'certificates10', 'awards10', '123432211'),

```
('liagourma',1,'bio11','sistatikes11','certificates11','awards11','18765549');
```

```
INSERT INTO administrator VALUES
```

```
('xristanag'),  
( 'mairypap'),  
( 'iosifzax'),  
( 'marzax');
```

```
INSERT INTO job VALUES
```

```
(001,'2019-01-01', 1800, 'data analyst', 'Patra, Greece', 'hlgeor', '2018-07-13  
10:00:00', '2018-12-20'),  
(111,'2019-02-01',1450, 'web programmer', 'Patra, Greece', 'vlasster', '2018-07-13  
11:00:00', '2019-01-10'),  
(456,'2019-02-01', 2100, 'mobile app developer', 'Patra, Greece', 'akisgoud', '2018-10-  
24 12:00:00', '2018-01-12'),  
(789,'2018-12-25', 2700, 'NLP expert', 'Peiraias, Greece', 'efstang', '2018-10-10',  
'2018-11-10'),  
(123,'2019-03-01', 2100, 'Graphics designer', 'Peiraias, Greece', 'adonkab', '2018-10-  
10', '2019-02-01'),  
(321,'2011-03-01', 2300, 'Visualization expert', 'Peiraias, Greece','dionkok', '2018-10-  
20', '2019-01-10'),  
(654,'2019-05-01', 1850, 'web and mobile app programmer', 'Athina,  
Greece','swtirzan', '2018-11-20', '2019-04-12'),  
(987,'2019-05-01', 1600, 'graphics expert', 'Athina, Greece','hlgeor', '2018-11-20',  
'2019-04-12'),  
(147,'2019-05-01', 1850, 'DB expert', 'Athina, Greece','vlasster', '2018-11-20', '2019-  
04-12'),  
(258,'2019-04-01', 2100, 'AI expert', 'Sofia, Bulgaria', 'akisgoud', '2018-11-21', '2019-  
03-10'),  
(369,'2019-02-01', 2600, 'Algorithmic efficiency expert', 'Sofia, Bulgaria', 'efstang',  
'2018-11-01', '2019-01-16'),  
(963,'2019-03-01', 2800, 'web and media programmer', 'Oxford, London', 'adonkab',  
'2018-11-01', '2019-01-03');
```

```
INSERT INTO project VALUES
```

```
('mnikol', 1, 'Minimal examples of data structures and algorithms in Python',  
'https://github.com/mnikol/algorithms'),  
( 'mnikol', 2, 'Interactive Online Platform that Visualizes Algorithms from Code',  
'https://github.com/mnikol/algorithm-visualizer'),  
( 'mnikol', 3, 'Repository which contains links and resources on different topics of  
Computer Science', 'https://github.com/mnikol/AlgoWiki'),  
( 'elenineo', 1, 'Essential Cheat Sheets for deep learning and machine learning  
researchers', 'https://github.com/elenineo/cheatsheets-ai'),
```

('elenineo', 2, 'Python sample codes for robotics algorithms.',
 'https://github.com/elenineo/PythonRobotics'),
 ('zazahir23', 1, 'Go Graphics - 2D rendering in Go with a simple
 API.', 'https://github.com/mob@s/gg'),
 ('zazahir23', 2, 'Draco is a library for compressing and decompressing 3D geometric
 meshes and point clouds. It is intended to improve the storage and transmission of
 3D graphics.', 'https://github.com/mob@s/draco'),
 ('zazahir23', 3, 'Data Discovery and Lineage for Big Data
 Ecosystem.', 'https://github.com/linkedin/WhereHows'),
 ('jennyk', 1, 'HTML5 Mobile App UI templates created using Intel App Framework.',
 'https://github.com/jenny/appframework-templates'),
 ('jennyk', 2, 'Mobile Version of Travel sample App using Couchbase Lite 2.0.',
 'https://github.com/jenny/mobile-travel-sample'),
 ('jennyk', 3, 'Appium Demo App with clearly defined Page Object Pattern for React
 Native Mobile App. Test Language - Javascript.', 'https://github.com/jenny/Appium-
 Page-Object-Model-Demo'),
 ('miltH', 1, 'WebGL2 powered geospatial visualization layers. offers an extensive
 catalog of pre-packaged visualization "layers", including ScatterplotLayer, ArcLayer,
 TextLayer, GeoJsonLayer, etc. The input to a layer is usually an array of JSON objects.
 Each layer offers highly-flexible API to customize how the data should be rendered.',
 'https://github.com/milti/deck.gl'),
 ('miltH', 2, 'Messy datasets? Missing values? missingno provides a small toolset of
 flexible and easy-to-use missing data visualizations and utilities that allows a quick
 visual summary of the completeness (or lack thereof) of the
 dataset.', 'https://github.com/milti/missingno'),
 ('miltH', 3, 'Repository to track the progress in Natural Language Processing (NLP),
 including the datasets and the current state-of-the-art for the most common NLP
 tasks', 'https://github.com/milti/NLP-progress'),
 ('miltH', 4, 'Supporting Rapid Prototyping with a Toolkit (incl. Datasets and Neural
 Network Layers)', 'https://github.com/milti/PyTorch-NLP')
 ;

INSERT INTO degree VALUES

('Lysium certificate', '2nd Lysium of Aigaleo', 'LYKEIO'),
 ('Computer and Infomatics Eng.', 'Patras University', 'UNIV'),
 ('Electrical and Computer Eng.', 'Metsovio Polytexneio', 'UNIV'),
 ('Computer Science Dipl.', 'Lancster University', 'UNIV'),
 ('Computer Vision and Modelling', 'Princeton University', 'MASTER'),
 ('Artificial Intelligence', 'Cambrigde University', 'MASTER'),
 ('Big Data and Analytics', 'Imperial College London', 'MASTER'),
 ('Advanced Rendering Techniques', 'Delft University of Technology', 'MASTER'),
 ('Computer Science and Engineering', 'Delft University of Technology', 'UNIV'),
 ('Data Science Bachelor', 'Eindhoven University of Technology', 'UNIV'),
 ('PDEng Data Science', 'Eindhoven University of Technology', 'PHD'),
 ('NLP related high efficiency algorithms', 'Patras University', 'PHD'),
 ('Big Data Structures and Algorithms', 'Technical University of Denmark', 'MASTER');

INSERT INTO has_degree VALUES

('Lysium certificate', '2nd Lysium of Aigaleo', 'mnikol', 1999, 19.2),
('Computer Science and Engineering', 'Delft University of Technology', 'mnikol', 2000, 8.2),
('PDEng Data Science', 'Eindhoven University of Technology', 'elenineo', 2006, 9),
('Electrical and Computer Eng.', 'Metsovio Polytexneio', 'kwstasneo', 1998, 7.6),
('Computer Vision and Modelling', 'Princeton University', 'kwstasneo', 2001, 8.5),
('Computer and Infomatics Eng.', ' Patras University', 'cleogeo', 2003, 8.6),
('Artificial Intelligence', ' Cambrigde University', 'zazahir23', 2008, 8),
('NLP related high efficiency algorithms', 'Patras University', 'liagourma', 2013, 9),
('Computer Science Dipl.', 'Lancster University', 'liagourma', 2001, 8.4),
('Computer Vision and Modelling', 'Princeton University', 'jennyk', 2006, 7.4),
('Data Science Bachelor', 'Eindhoven University of Technology', 'jennyk', 2004, 9.2),
('Big Data and Analytics', ' Imperial College London', 'jennyk', 2006, 8),
('Big Data Structures and Algorithms', 'Technical University of Denmark', 'miltH', 2008, 8.2);

INSERT INTO antikeim VALUES

('Computer Science', 'Root element, no more general antikeim', NULL),
('Databases', 'Level one element, child of Computer Science', 'Computer Science'),
('AI', 'Level one element, child of Computer Science', 'Computer Science'),
('Algorithms', 'Level one element, child of Computer Science', 'Computer Science'),
('Networking', 'Level one element, child of Computer Science', 'Computer Science'),
('Graphics', 'Level one element, child of Computer Science', 'Computer Science'),
('2D', 'Level two element, child of Graphics', 'Graphics'),
('3D', 'Level two element, child of Graphics', 'Graphics'),
('Animation', 'Level two element, child of Graphics', 'Graphics'),
('Programming', 'Level one element, child of Computer Science', 'Computer Science'),
('Web Programming', 'Level two element, child of Programming', 'Programming'),
('Mobile Apps', 'Level two element, child of Programming', 'Programming'),
('Relational DBs', 'Level two element, child of Databases', 'Databases'),
('Object-Oriented DBs', 'Level two element, child of Databases', 'Databases'),
('NoSQL DBs', 'Level two element, child of Databases', 'Databases'),
('Robotics', 'Level two element, child of AI', 'AI'),
('NLP', 'Level two element, child of AI', 'AI'),
('Information Retieval', 'Level three element, child of NLP', 'NLP'),
('Language analysis', 'Level three element, child of NLP', 'NLP'),
('Data structures', 'Level two element, child of Algorithms', 'Algorithms'),
('Complexity and Efficiency', 'Level two element, child of Algorithms', 'Algorithms'),


```
('Network setup and maintainance', 'Level two element, child of Networking',  
'Networking'),  
('Device connectivity', 'Level two element, child of Networking', 'Networking');  
INSERT INTO needs VALUES  
(001, 'Databases'),  
(001, 'Algorithms'),  
(111, 'Programming'),  
(456, 'Web Programming'),  
(456, 'Mobile Apps'),  
(789, 'Animation'),  
(123, 'AI'),  
(321, 'NLP'),  
(321, 'Graphics'),  
(654, 'Graphics'),  
(654, 'Algorithms'),  
(987, 'Programming'),  
(987, 'Web Programming'),  
(147, 'Mobile Apps'),  
(258, '2D'),  
(963, '3D'),  
(963, 'Databases'),  
(963, 'NoSQL DBs'),  
(456, 'AI'),  
(789, 'Complexity and Efficiency'),  
(147, 'Algorithms'),  
(258, 'Web Programming'),  
(369, 'Mobile Apps'),  
(111, 'Animation');
```

```
INSERT INTO languages VALUES  
( 'mnikol', 'EN,SP,GR'),  
( 'abrown', 'GR,EN'),  
( 'elenineo', 'EN,FR'),  
( 'kwstasneo', 'GR,EN'),  
( 'cleogeo', 'GR,FR'),  
( 'zazahir23', 'EN,FR,SP'),  
( 'lionarF', 'EN,GR'),  
( 'liagourma', 'FR'),  
( 'msmith', 'SP,FR,EN,GR'),  
( 'jennyk', 'SP,GR'),  
( 'miltH', 'EN,GR');
```

```
INSERT INTO evaluation VALUES
```

```

('mnikol','hlgeor',001,'4','3','1','good',20)
('mnikol','vlasster',111,'1',NULL,NULL,'not good',21),
('abrown','akisgoud',456,'2','3','2','good',22),
('abrown','hlgeor',001,NULL,'3','1','not good',23),
('elenineo','efstang',789,'2','4','2','good',24),
('elenineo','adonkab',123,'1',NULL,'0','not good',25),
('kwstasneo','dionkok',321,'3','2','0','good',26),
('kwstasneo','swtirzan',654,NULL,'4','2','good',27),
('cleogeo','hlgeor',987,'4','4','2','good',28),
('cleogeo','vlasster',147,'3',NULL,'1','not good',29),
('zazahir23','akisgoud',258,'4','0','0','not good',30),
('zazahir23','efstang',369,NULL,NULL,'2','not good',31),
('lionarF','adonkab',963,'2','2','2','good',32),
('lionarF','hlgeor',001,'3','1',NULL,'not good',33),
('liagourma','vlasster',111,'4','4','2','good',34),
('liagourma','akisgoud',456,'1','2',NULL,'not good',35),
('msmith','efstang',789,'4','1','1','good',36),
('msmith','adonkab',123,NULL,'2',NULL,'not good',37),
('jennyk','dionkok',321,'2','4','2','good',38),
('jennyk','swtirzan',654,NULL,'4','2','good',39),
('miltH','akisgoud',456,'4','3','2','good',40),
('miltH','adonkab',963,NULL,'3','2','good',41);

```

INSERT INTO promotion VALUES

```

('nikipap','hlgeor',272),
('vasmitr','dionkok',123),
('Theoxrist','swtirzan',789),
('nikipap','efstang',369),
('nikipap','efstang',654),
('stavroskost','adonkab',963),
('ptolkor','akisgoud',456);

```

insert into requestsevaluation VALUES

```

('mnikol', 111),
('abrown',001),
('elenineo',123),
('kwstasneo', 654),
('cleogeo',147 ),
('zazahir23',369 ),
('lionarF',001),
('liagourma',456),
('msmith',123),
('jennyk',654),
('miltH',963);

```

3) Stored Procedures

3.1) Stored procedure

DELIMITER \$

```
CREATE PROCEDURE aitiseis_aksiologiseis_evaluatorOfEmployee (IN Name
varchar(25), IN SurName varchar(35))

BEGIN

    DECLARE Username VARCHAR(12);
    SELECT username INTO Username FROM user WHERE user.name=Name AND
user.surname=SurName;

    IF(Username NOT IN (SELECT empl_username FROM requestsevaluation) AND
Username NOT IN (SELECT emplo_username FROM requestspromotion))
    THEN
        SELECT 'There are no evaluation and promotion requests for this employee.';

    ELSE

        SELECT 'Evaluation requests:';
        SELECT empl_username,job_id FROM requestsevaluation INNER JOIN
employee ON employee.username=empl_username
        INNER JOIN user ON user.username=employee.username WHERE
user.name=Name AND user.surname=SurName;

        SELECT 'Promotion requests:';
        SELECT emplo_username,jobID FROM requestspromotion INNER JOIN
employee ON employee.username=emplo_username
        INNER JOIN user ON user.username=employee.username WHERE
user.name=Name AND user.surname=SurName;
        END IF;

        IF(Username NOT IN (SELECT empl_username FROM evaluationresult)) THEN
        SELECT 'This employee hasnt been fully evaluated.';

    ELSE
        SELECT EvId,empl_username,job_id,grade,comments FROM evaluationresult
        INNER JOIN employee ON employee.username=empl_username
```

```
INNER JOIN user ON user.username=employee.username WHERE  
user.name=Name AND user.surname=SurName;
```

```
SELECT evalusername FROM evaluation INNER JOIN employee ON  
employee.username=emplusername  
INNER JOIN user ON user.username=employee.username WHERE  
user.name=Name AND user.surname=SurName;
```

```
END IF;
```

```
IF(Username NOT IN (SELECT emplusername FROM evaluation)) THEN  
SELECT 'This employee hasnt been evaluated at all.';
```

```
ELSE
```

```
SELECT 'Aksiologisi se ekseliksi.';  
SELECT  
emplusername,evalusername,jobid,aksiologisi1,aksiologisi2,aksiologisi3,comm  
ents,eval_id FROM evaluation INNER JOIN employee ON  
employee.username=emplusername  
INNER JOIN user ON user.username=employee.username WHERE  
user.name=Name AND user.surname=SurName AND (aksiologisi1 IS NULL OR  
aksiologisi2 IS NULL OR aksiologisi3 IS NULL);
```

```
END IF;
```

```
END $
```

3.2) Stored Procedure

```
DELIMITER $
```

```
CREATE PROCEDURE TelikeskaiOristikopihmenes_aksiologiseis (IN id_thesis INT, IN  
evaluator_name varchar(12))
```

```
BEGIN
```

```
DECLARE grade1 INT;
```

```
DECLARE grade2 INT;
```

```
DECLARE grade3 INT;
```

```
DECLARE finishedFlag INT;
```

```
DECLARE gcursor CURSOR FOR SELECT aksiologisi1,aksiologisi2,aksiologisi3  
FROM evaluation WHERE jobid=id_thesis AND  
evalusername=evaluator_name;
```

```

DECLARE CONTINUE HANDLER FOR NOT FOUND SET finishedFlag=1;
OPEN gcursor;
SET finishedFlag=0;

IF(id_thesis NOT IN (SELECT jobid FROM evaluation WHERE jobid=id_thesis ))
THEN
SELECT 'There is no evaluation for this job.';
END IF;

IF(evaluator_name NOT IN (SELECT evalusername FROM evaluation WHERE
evalusername=evaluator_name )) THEN
SELECT 'There is no evaluation for this job.';
END IF;

REPEAT
FETCH gcursor INTO grade1,grade2,grade3;
IF(finishedFlag=0) THEN

        IF(grade1 IS NULL OR grade2 IS NULL OR grade3 IS NULL) THEN
        SELECT 'Den iparxoun bathmoi kai stis 3 faseis aksiologisis';
        SELECT * FROM evaluation WHERE jobid=id_thesis AND
evalusername=evaluator_name AND (aksiologisi1 IS NULL OR aksiologisi2 IS
NULL OR aksiologisi3 IS NULL);
        ELSE
        SELECT 'TELIKH KAI ORISTIKOPOIHMENH AKSIOLOGISI';

        SELECT * FROM evaluationresult WHERE job_id IN (SELECT jobid
FROM evaluation WHERE jobid=id_thesis AND evalusername=evaluator_name
AND (aksiologisi1 <>" AND aksiologisi2 <>" OR aksiologisi3 <>" ));

        END IF;

END IF;
UNTIL (finishedFlag=1)
END REPEAT;
CLOSE gcursor;

END $
DELIMITER ;

```

3.3) Stored Procedure

```
DELIMITER $

CREATE PROCEDURE oristikopoihmenes_aksiologhseis(IN kwdikos_job INT(4))

BEGIN

    DECLARE COUNTER INT;

    SELECT COUNT(*) INTO COUNTER from evaluation where jobid=kwdikos_job
    AND (aksiologisi1 IS NULL OR aksiologisi2 IS NULL OR aksiologisi3 IS NULL);

    IF (kwdikos_job NOT IN (SELECT job_id FROM evaluationresult)) THEN
        SELECT 'Den iparxei aksiologisi gia auth thn thesh.';
    ELSE
        SELECT 'Oristikopoihmenoi pinakes.';
        SELECT * FROM evaluationresult WHERE job_id=kwdikos_job ORDER BY
        grade DESC;
    END IF;

    IF(kwdikos_job IN(SELECT jobid FROM evaluation WHERE (aksiologisi1 IS
    NULL OR aksiologisi2 IS NULL OR aksiologisi3 IS NULL)))
    THEN
        SELECT * FROM evaluation WHERE jobid=kwdikos_job AND (aksiologisi1 IS
        NULL OR aksiologisi2 IS NULL OR aksiologisi3 IS NULL) ORDER BY
        aksiologisi1,aksiologisi2,aksiologisi3 DESC;

    ELSE
        SELECT 'Den iparxoun aksiologiseis se ekxeliksi gia ayth thn thesi.';
    END IF;

    IF(COUNTER >0 ) THEN
        SELECT 'Aksiologisi se ekxeliksi...ekremmoun aitiseis';
        SELECT COUNTER;

    ELSE
        SELECT 'Den uparxoun upopshfioi';

    END IF;

END $
```

4) Triggers

4.1) Για το πρώτο ερώτημα κατασκευάζουμε 9 trigger , 3 για κάθε πίνακα (**job, employee, requestsevaluation**) αφού θέλουμε να ενημερώνεται ο πίνακας log για κάθε ενέργεια **εισαγωγής ,ενημέρωσης η διαγράφης.**

1. Εισαγωγή στον πίνακα job

DELIMITER \$

```
CREATE TRIGGER inserting_intojob  
BEFORE INSERT ON job  
FOR EACH ROW  
BEGIN
```

```
    IF(NEW.id IN (SELECT id FROM job))  
    THEN  
        INSERT INTO log  
        VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','INSERT','job');
```

```
    ELSE  
        INSERT INTO log  
        VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','INSERT','job');  
    END IF;
```

```
END$
```

DELIMITER ;

2. Ενημέρωση στον πίνακα job

DELIMITER \$

```
CREATE TRIGGER updatingtable_job  
BEFORE UPDATE ON job  
FOR EACH ROW  
BEGIN
```

```
    IF(NEW.id=OLD.id AND NEW.start_date=OLD.start_date AND  
    NEW.salary=OLD.salary AND NEW.position=OLD.position AND OLD.edra=NEW.edra  
    AND OLD.evaluator=NEW.evaluator
```

```
AND NEW.announce_date=OLD.announce_date AND  
NEW.submission_date=OLD.submission_date) THEN
```

```
INSERT INTO log  
VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','UPDATE','job');
```

```
ELSE  
INSERT INTO log  
VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','UPDATE','job');
```

```
END IF;  
END $
```

3. Διαγραφή στον πίνακα job

```
CREATE TRIGGER deletingfrom_job  
BEFORE DELETE ON job  
FOR EACH ROW  
BEGIN
```

```
IF(OLD.id NOT IN (SELECT id FROM job)) THEN  
INSERT INTO log  
VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','DELETE','job');
```

```
ELSE  
INSERT INTO log  
VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','DELETE','job');
```

```
END IF;  
END$
```

```
DELIMITER $
```

4. Εισαγωγή στον πίνακα employee

```
CREATE TRIGGER insertinginto_employee  
BEFORE INSERT ON employee  
FOR EACH ROW  
BEGIN
```

```
IF(NEW.username IN (SELECT username FROM employee))  
THEN
```



```

INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','INSERT','employee');

ELSE
INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','INSERT','employee');
END IF;

END$

DELIMITER ;

```

5. Ενημέρωση στον πίνακα employee

```

DELIMITER $

CREATE TRIGGER updatingtable_employee
BEFORE UPDATE ON employee
FOR EACH ROW
BEGIN

    IF(NEW.username=OLD.username AND NEW.exp_years=OLD.exp_years AND
NEW.bio=OLD.bio AND NEW.sistatikes=OLD.sistatikes AND
NEW.certificates=OLD.certificates
AND OLD.awards=NEW.awards AND OLD.firm=NEW.firm) THEN

INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','UPDATE','employee');

ELSE
INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','UPDATE','employee');
END IF;
END $

```

6. Διαγραφή στον πίνακα employee

```

CREATE TRIGGER deletingfrom_employee
BEFORE DELETE ON employee
FOR EACH ROW
BEGIN

```

```

IF(OLD.username NOT IN (SELECT username FROM employee)) THEN

INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','DELETE','employee');

ELSE
INSERT INTO log
VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','DELETE','employee');
END IF;
END $

```

7. Εισαγωγή στον πίνακα requestsevaluation

```

DELIMITER $
CREATE TRIGGER insertinginto_requestsevaluation
BEFORE INSERT ON requestsevaluation
FOR EACH ROW
BEGIN

    IF(NEW.empl_username IN (SELECT empl_username FROM requestsevaluation)
    AND NEW.job_id IN (SELECT job_id FROM requestsevaluation) )

THEN
    INSERT INTO log
    VALUES(@useronoma,CURRENT_TIMESTAMP,'NO','INSERT','requestsevaluati
on');

ELSE
    INSERT INTO log
    VALUES(@useronoma,CURRENT_TIMESTAMP,'YES','INSERT','requestsevaluati
on');
END IF;

END$

DELIMITER ;

```

8. Ενημέρωση στον πίνακα requestsevaluation

```

DELIMITER $

CREATE TRIGGER updatingtable_requestsevaluation
BEFORE UPDATE ON requestsevaluation
FOR EACH ROW

```

```

BEGIN

    IF(NEW.empl_username=OLD.empl_username AND NEW.job_id=OLD.job_id)
    THEN

        INSERT INTO log
        VALUES(@useronoma
        ,CURRENT_TIMESTAMP,'NO','UPDATE','requestsevaluation');

    ELSE
        INSERT INTO log
        VALUES(@useronoma
        ,CURRENT_TIMESTAMP,'YES','UPDATE','requestsevaluation');
    END IF;
END $

```

9. Διαγραφή στον πίνακα requestsevaluation

```

CREATE TRIGGER deletingfrom_requestsevaluation
BEFORE DELETE ON requestsevaluation
FOR EACH ROW
BEGIN

    IF(OLD.empl_username AND OLD.job_id NOT IN (SELECT empl_username FROM
requestsevaluation)) THEN
        INSERT INTO log
        VALUES(@useronoma
        ,CURRENT_TIMESTAMP,'NO','DELETE','requestsevaluation');

    ELSE
        INSERT INTO log
        VALUES(@useronoma
        ,CURRENT_TIMESTAMP,'YES','DELETE','requestsevaluation');

    END IF;
END$

```

4.2) Ο trigger που ζητείται είναι ο εξής :

```

DELIMITER $
CREATE TRIGGER cannotchangefieldsetsofcompany
BEFORE UPDATE ON company
FOR EACH ROW

```

```

BEGIN
  IF(NEW.AFM <> OLD.AFM OR NEW.DOY <> OLD.DOY OR NEW.name <>
  OLD.name) THEN
    SET NEW.AFM=OLD.AFM;
    SET NEW.DOY=OLD.DOY;
    SET NEW.name=OLD.name;
  END IF;
END $

```

4.3) Ο trigger που ζητείται είναι ο εξής :

```

DELIMITER $
CREATE TRIGGER usercannotchangefields
BEFORE UPDATE ON user
FOR EACH ROW
BEGIN
  IF (@useronoma IN (SELECT managerUsername FROM manager)) THEN
    SET NEW.username=OLD.username;
    SET NEW.name=OLD.name;
    SET NEW.surname=OLD.surname;
    SET NEW.reg_date=OLD.reg_date;

    ELSEIF(@useronoma IN (SELECT username FROM evaluator)) THEN
    SET NEW.username=OLD.username;

    ELSEIF(@useronoma IN (SELECT username FROM employee)) THEN
    SET NEW.email=OLD.email;
    SET NEW.username=OLD.username;
    SET NEW.name=OLD.name;
    SET NEW.surname=OLD.surname;
    SET NEW.reg_date=OLD.reg_date;

    ELSEIF(@useronoma IN (SELECT username FROM administrator)) THEN
    SET NEW.username=OLD.username;

  END IF;
END $

```

Μέρος B: GUIs

1.

Για το συγκεκριμένο μέρος του πρότζεκτ χρησιμοποιήσαμε την **Python** ως γλώσσα προγραμματισμού για να δημιουργήσουμε τις ζητούμενες διεπαφές.

Για να κατασκευάσουμε τα GUIs στην Python χρησιμοποιήσαμε την βιβλιοθήκη **Tkinter**. Ενώ για να πέτυχουμε την σύνδεση της Python με την βάση δεδομένων που φτιάξαμε με όνομα **staffevaluation** χρησιμοποιήσαμε την βιβλιοθήκη της Python **mysql.connector**.

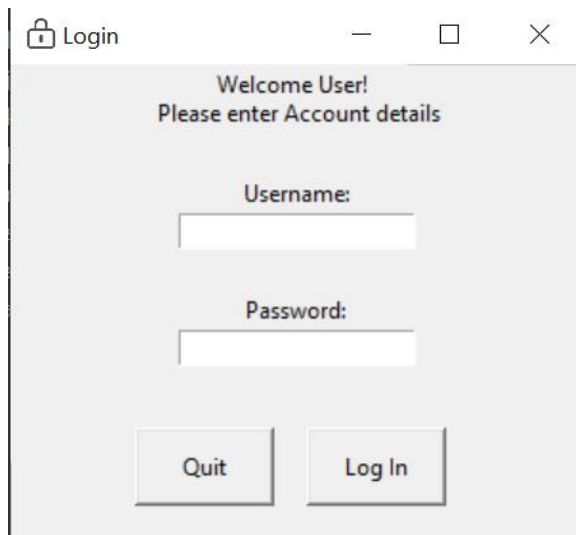
Επίσης χρησιμοποιήσαμε κάποιες εικόνες (αρχεία .ico) για κάποια από τα παράθυρα που εμφανίζονται. Έχουν συγκεκριμένο path πχ

```
root.iconbitmap("C:/Users/dorak/Downloads/lock.ico")
```

το οποίο αντιστοιχεί στον δικό μας ηλεκτρονικό υπολογιστή.

Θα συμπεριλάβουμε τα αρχεία .ico που χρησιμοποιήσαμε στο αρχείο .zip

Η αρχική σελίδα σύνδεσης για τους 4 χρήστες (manager,evaluator,employee,administrator) είναι η εξής:



Ο κάθε χρήστης εισάγει username και password και ανάλογα με την κατηγορία κάθε χρήστη θα εμφανίζεται το κατάλληλο GUI.

Με την χρήση της **stored procedure** paretousername προσπαθήσαμε να πάρουμε το username του χρήστη που συνδέεται ώστε να αποθηκεύεται στον πίνακα log κάθε

φορά που γίνεται μια ενέργεια **εισαγωγής ,ενημέρωσης η διαγράφης** αλλά δεν το καταφέραμε.

DELIMITER \$

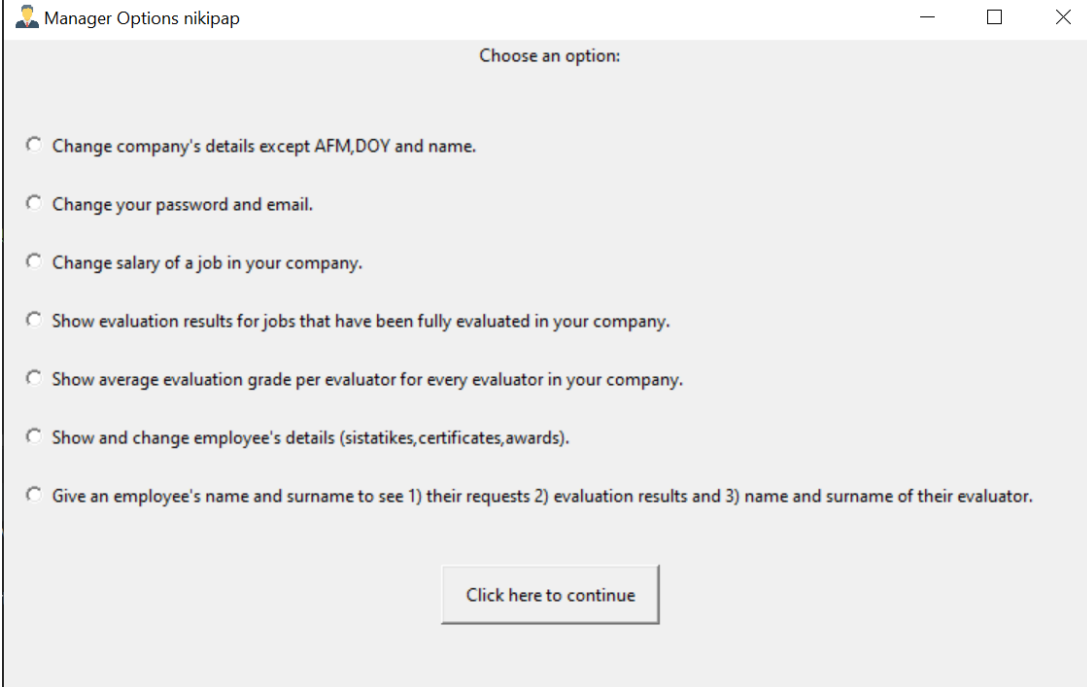
```
CREATE PROCEDURE paretusername(IN usernamexrhsth varchar(12))
```

```
BEGIN
```

```
    SET @useronoma=usernamexrhsth;
```

```
END$
```

Συγκεκριμένα αν ο χρήστης που θα συνδεθεί είναι διευθυντής/**manager** εμφανίζεται το παρακάτω GUI στην οθόνη του με τις επιλογές που μπορεί να κάνει:



Manager Options nikipap

Choose an option:

- ☐ Change company's details except AFM,DOY and name.
- ☐ Change your password and email.
- ☐ Change salary of a job in your company.
- ☐ Show evaluation results for jobs that have been fully evaluated in your company.
- ☐ Show average evaluation grade per evaluator for every evaluator in your company.
- ☐ Show and change employee's details (sistatikes,certificates,awards).
- ☐ Give an employee's name and surname to see 1) their requests 2) evaluation results and 3) name and surname of their evaluator.

[Click here to continue](#)

- 1) Αν ο manager επιλέξει να κάνει την **πρώτη** επιλογή δηλαδή να αλλάξει τα στοιχεία της εταιρείας του εκτός από ΑΦΜ , ΔΟΥ και επωνυμία εμφανίζεται το εξής παράθυρο:

Change company's details

Enter the details to be changed:

Phone:

Street:

Number:

City:

Country:

Change

Στην συνέχεια αν ο διευθυντής πατήσει το κουμπί Change θα αλλάξουν τα στοιχεία της εταιρείας του στην βάση δεδομένων με την χρήση της παρακάτω **stored procedure** με όνομα **syndesicompanyupdate** που φτιάξαμε και την καλούμε στην Python:

DELIMITER \$

```
CREATE PROCEDURE syndesicompanyupdate(IN usernamemanager varchar(12), IN phonecmp bigint, IN streetcmp varchar(15), IN numcmp INT, IN citycmp varchar(15), IN countrycmp varchar(15))
```

```
BEGIN
```

```
DECLARE etairia char(9);
```

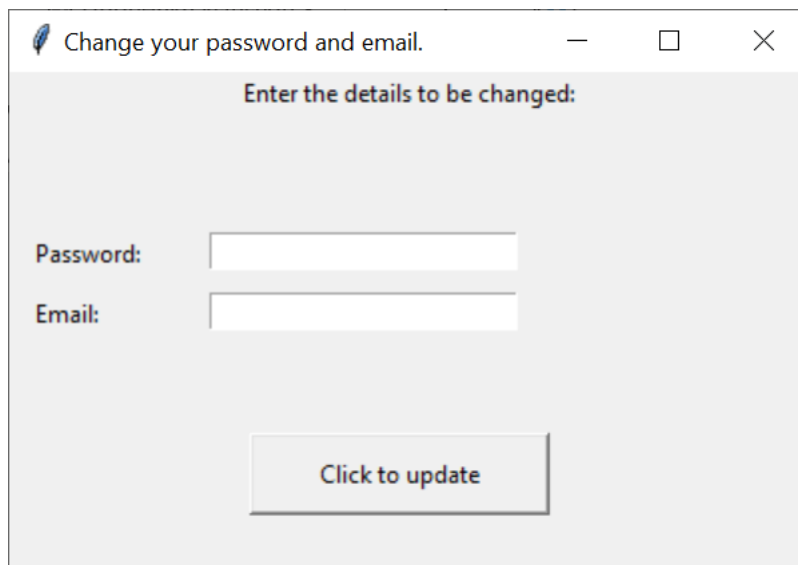
```
SELECT firm INTO etairia FROM manager WHERE  
managerUsername=usernamemanager;
```

```
UPDATE company SET phone = phonecmp , street = streetcmp , num =  
numcmp , city=citycmp, country = countrycmp
```

```
WHERE AFM=etairia;
```

```
END $
```

2) Αν επιλέξει την **δεύτερη** επιλογή εμφανίζεται αυτό το παράθυρο:



Change your password and email.

Enter the details to be changed:

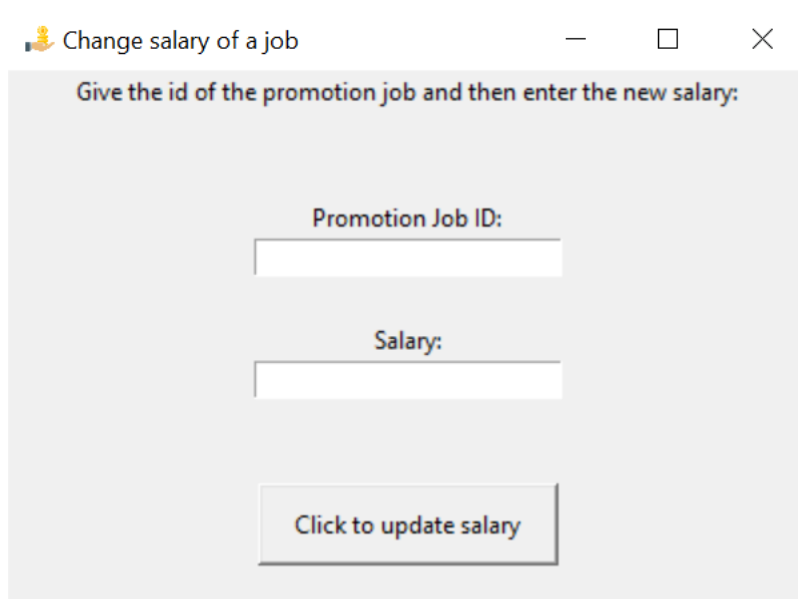
Password:

Email:

Click to update

Και στην συνέχεια αν πατήσει το κουμπί Click to update το password και το email του θα ενημερωθούν στην βάση δεδομένων και θα αλλάξουν σε αυτά που συμπλήρωσε ο ίδιος. Αυτό γίνεται με κλήση εντολής update της sql μέσω της **my_cursor.execute()** στην Python.

3) Αν επιλέξει την τρίτη επιλογή θα εμφανιστεί στην οθόνη του το εξής παράθυρο:



Change salary of a job

Give the id of the promotion job and then enter the new salary:

Promotion Job ID:

Salary:

Click to update salary

Αν ο manager συμπληρώσει τα κενά και πατήσει το κουμπί Click to update salary ο μισθός της θέσης εργασίας προαγωγής που δίνει θα αλλάξει στην βάση δεδομένων με την χρήση μιας **stored procedure** με όνομα **syndesimisthos_manager1** που φτιάξαμε και την καλούμε στην Python:

```
DELIMITER $
```

```
CREATE PROCEDURE syndesimisthos_manager1(IN username_manag varchar(12),IN idjob INT,IN salary_job float)
```

```
BEGIN
```

```
    DECLARE Job INT(4);
```

```
    SELECT id_job INTO Job FROM promotion INNER JOIN manager ON  
manag_username=managerUsername WHERE  
managerUsername=username_manag AND id_job=idjob ;
```

```
    UPDATE job SET salary = salary_job WHERE id=Job;
```

```
END $
```

- 4) Αν ο διευθυντής επιλέξει την **τέταρτη** επιλογή τότε τα αποτελέσματα της αξιολόγησης θα φανούν στο terminal της Python:

```
[(26, 'kwstasneo', 321, 5, 'good'), (34, 'liagourma', 111, 10, 'good')]
```

Τα αποτελέσματα που ζητούνται προκύπτουν από την κλήση μιας **stored procedure** που φτιάξαμε με όνομα **emfaniseapotelesmataston_manager** και την καλούμε στην Python:

```
CREATE PROCEDURE emfaniseapotelesmataston_manager (IN managusername  
varchar(12))
```

```
BEGIN
```

```
    SELECT EvId,empl_usrname,job_id,grade,comments FROM evaluationresult  
INNER JOIN employee ON username=empl_usrname INNER JOIN company ON  
AFM=employee.firm
```

```
    INNER JOIN manager ON manager.firm=AFM WHERE  
managerUsername=managusername;
```

```
END $
```

Δεν καταφέραμε να εισάγουμε τα αποτελέσματα των αξιολογήσεων σε κάποιο παράθυρο και έτσι εκτυπώνονται στην Python.

- 5) Αν πατήσει το **πέμπτο** στην σειρά κουμπί εμφανίζονται τα αποτελέσματα δηλαδή ο μέσος όρος βαθμού αξιολόγησης ανά αξιολογητή στην εταιρεία του για όλους τους αξιολογητές στην Python.

Πχ για τον manager nikipap:

```
Average Evaluation grade per evaluator in your company:
[('efstang', 6.666666666666667)]
[('efstang', 6.666666666666667)]
[('efstang', 6.666666666666667)]
[('hlgeor', 6.0)]
[('hlgeor', 6.0)]
```

Τα παραπάνω αποτελέσματα προκύπτουν από την κλήση μιας stored procedure :

```
CREATE PROCEDURE emfanise_averagevathmo_anaevaluator(IN manag_urname
varchar(12))
```

```
BEGIN
```

```
    DECLARE average_vathmos INT;
```

```
    DECLARE aksiologitis varchar(12);
```

```
    DECLARE finished_flag INT;
```

```
    DECLARE evalcursor CURSOR FOR SELECT evalusername FROM evaluation
INNER JOIN evaluator ON evaluator.username=evalusername INNER JOIN company
ON AFM=evaluator.firm
```

```
    INNER JOIN manager ON manager.firm=AFM WHERE
managerUsername=manag_urname;
```

```
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished_flag=1;
```

```
    OPEN evalcursor;
```

```
    SET finished_flag=0;
```

```
REPEAT
```

```

        FETCH evalcursor INTO aksiologitis;

        IF(finished_flag=0) THEN

            Select
evalusername,(SUM(aksiologisi1+aksiologisi2+aksiologisi3)/COUNT(aksiologitis)) as
AverageGrade

                from evaluation where evalusername=aksiologitis group by
aksiologitis;

            END IF;

        UNTIL (finished_flag=1)

        END REPEAT;

        CLOSE evalcursor;

END $

```

- 6) Αν πατήσει το **έκτο** στην σειρά κουμπί θα εμφανιστεί ο φάκελος του employee στην Python :

Αυτοι είναι οι εργαζομενοι στην εταιρεια του και εμφανίζονται με την βοθηεια της κλησης της stored procedure:

```

CREATE PROCEDURE emfanisefakeloemployee(IN user_manager varchar(12))
BEGIN
    DECLARE compan char(9);

    SELECT firm INTO compan FROM manager WHERE
managerUsername=user_manager;

    SELECT * from employee WHERE employee.firm =compan;

END $

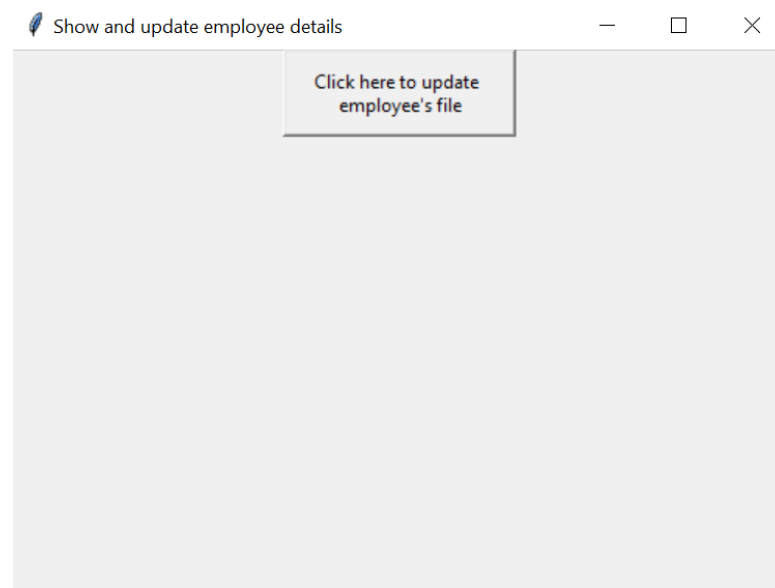
```

```

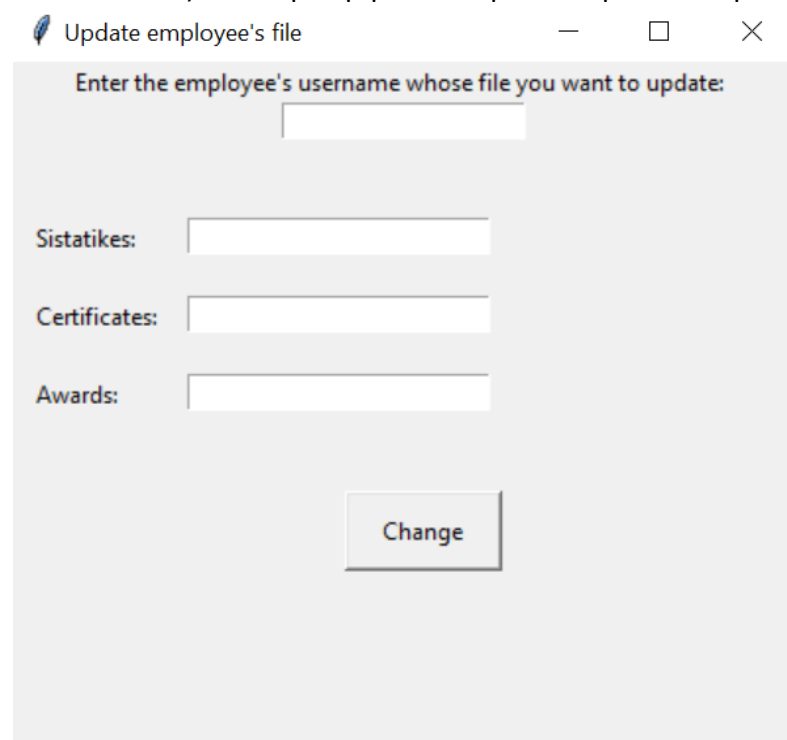
[('kwstasneo', 1, 'bio4', 'sistatikes4', 'certificates4', 'awards4', '18765549'),
('liagourma', 1, 'bio11', 'sistatikes11', 'certificates11', 'awards11', '18765549')]

```

Εμφανίζεται το παράθυρο:



Και πατώντας στο συγκεκριμένο κουμπί ανοίγει νέο παράθυρο :



Ο manager συμπληρώνει τα κενά στο παράθυρο και ενημερώνονται τα στοιχεία του employee που θέλει ο manager στην βάση δεδομένων πατώντας το κουμπί Change με την χρήση μιας stored procedure που δημιουργήσαμε με όνομα **allaksestoixeiaemployee** και την καλούμε στην Python:

```
CREATE PROCEDURE allaksestoixeiaemployee(IN manager_user varchar(12),IN  
emplusername varchar(12),IN sistat varchar(35),IN certific varchar(35),IN vraveia  
varchar(35))
```

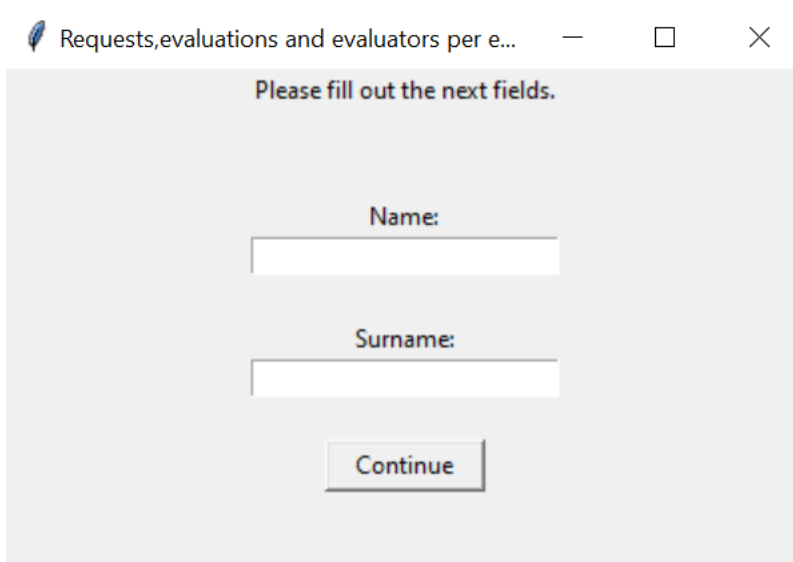
```
BEGIN
```

```
call emfanisefakeloemployee(manager_user);
```

```
UPDATE employee set sistatikes = sistat, certificates = certific , awards= vraveia  
WHERE employee.username=emplusername;
```

```
END $
```

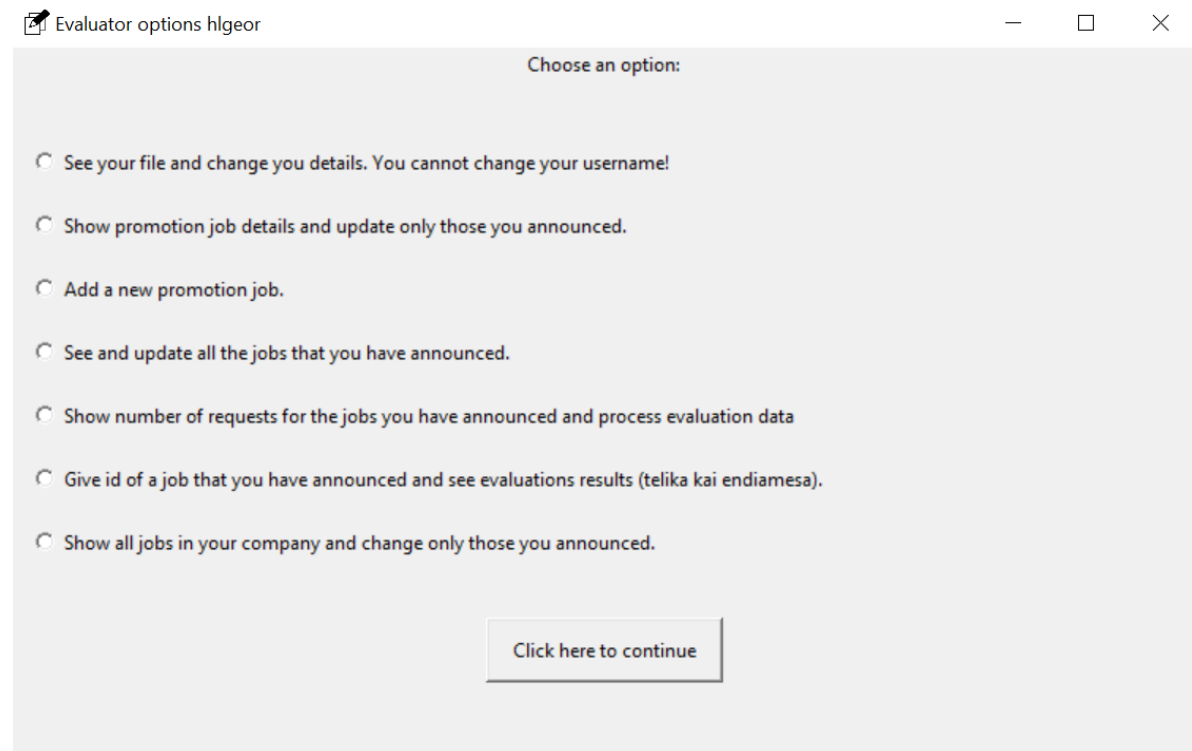
7) Αν επιλέξει την **εβδομή** επιλογή τότε εμφανίζεται ένα νέο παράθυρο:



Ο manager δίνει το ονοματεπώνυμο ενός employee και με την κλήση της **stored procedure 3.1** εμφανίζονται τα ζητούμενα αποτελέσματα στην Python πατώντας το κουμπί Continue.

```
[('Evaluation requests:',)]  
[('cleogeo', 147)]  
[('Promotion requests:',)]  
[('cleogeo', 1)]  
[(28, 'cleogeo', 987, 10, 'good')]  
[('hlgeor',), ('vlasster',)]  
[('Aksiologisi se ekseliksi.',)]  
[('cleogeo', 'vlasster', 147, '3', None, '1', 'not good', 29)]  
|
```

Αν ο χρήστης που θα συνδεθεί την αρχική σελίδα σύνδεσης ανήκει στην κατηγορία **evaluator** (αξιολογητή) εμφανίζεται το παρακάτω GUI στην οθόνη του με τις επιλογές που μπορεί να κάνει:



- 1) Αν ο evaluator επιλέξει το πρώτο κουμπί θα εμφανιστούν στην Python τα στοιχεία του λογαριασμού του:

```
('hlgeor', 'hl20', 'Hlias', 'Georgiadis', datetime.datetime(2003, 2, 12, 12, 0, 21), 'hl20@mail.com')  
( 'hlgeor', 10, '18765549')
```

Αυτό το πετυχαίνουμε με την χρήση μιας **stored procedure** που φτιάξαμε με όνομα **blepeifakelo** την οποία καλούμε στην Python :

DELIMITER \$

```
CREATE PROCEDURE blepeifakelo (IN employeeusername varchar(12))
```

```
BEGIN
```

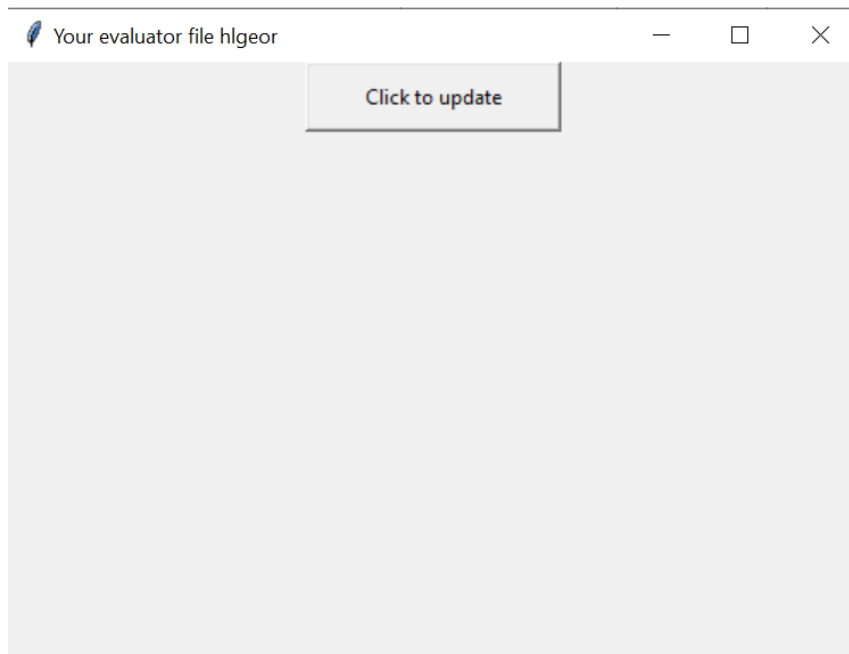
```
    SELECT employee.username,exp_years,bio,sistatikes,certificates,awards,firm  
FROM employee INNER JOIN user ON user.username=employee.username WHERE  
user.username=employeeusername;
```

```
    SELECT username,password,name,surname,reg_date,email FROM user  
WHERE user.username=employeeusername;
```

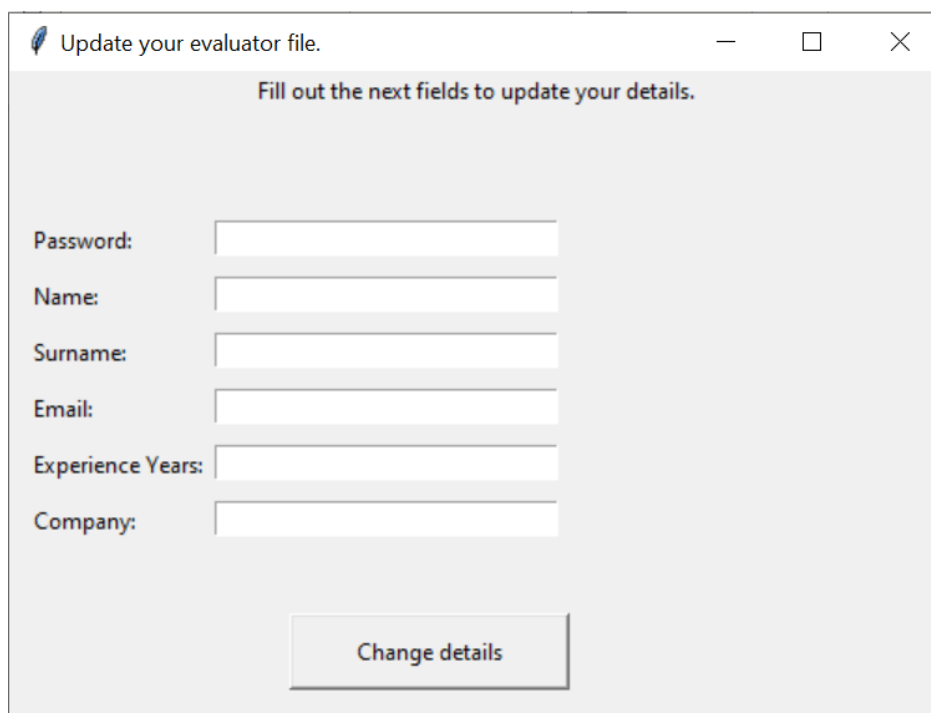
```
END $
```

DELIMITER ;

Εμφανίζεται στην συνέχεια το παρακάτω παράθυρο στην οθόνη του evaluator που έχει συνδεθεί στο σύστημα:



Αν πατήσει πάνω στο κουμπί με όνομα Click to update που εμφανίζεται ένα νέο παράθυρο με σκοπό να αλλάξει αν θέλει τα στοιχεία στον λογαριασμό του:

A screenshot of a software window titled "Update your evaluator file.". The window has a title bar with minimize, maximize, and close buttons. Below the title bar, there is a subtitle "Fill out the next fields to update your details.". The main area contains a form with the following fields: "Password:", "Name:", "Surname:", "Email:", "Experience Years:", and "Company:". Each field has a corresponding text input box. At the bottom center of the window, there is a button labeled "Change details".

Αν ο evaluator πατήσει το κουμπί Change details τα στοιχεία του θα αλλάξουν στην βάση δεδομένων με την χρήση της εντολής **my_cursor.execute()** στην Python με την οποία εκτελούμε τα update που χρειαζόμαστε.

2) Αν ο υπεύθυνος αξιολόγησης επιλέξει το δεύτερο στην σειρά κουμπί τότε:

Εμφανίζονται τα ζητούμενα αποτελέσματα στην Python στο terminal με την χρήση μιας stored procedure που φτιάξαμε με όνομα **evaluator_seejobpromotions** που καλούμε στην Python:

```
CREATE PROCEDURE evaluator_seejobpromotions(IN eval_username varchar(12))
BEGIN
DECLARE etaireia char(9);

SELECT    firm      INTO    etaireia      FROM    evaluator      WHERE
evaluator.username=eval_username;

SELECT
id,start_date,salary,position,edra,evaluator,announce_date,submission_date
FROM job INNER JOIN evaluator ON evaluator.username=evaluator WHERE
firm=etaireia AND id IN (SELECT id_job from promotion);

END$
```

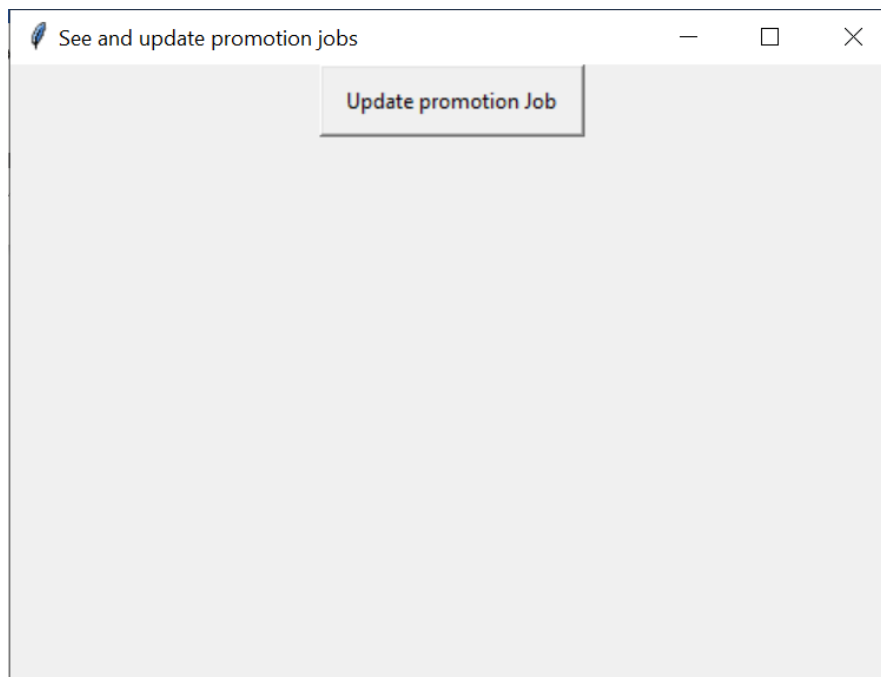
π.χ. [(1, datetime.date(2019, 1, 1), 1800.0, 'data analyst', 'Patra, Greece', 'hlgeor',
datetime.datetime(2018, 7, 13, 10, 0), datetime.date(2018, 12, 20)),

(272, datetime.date(2018, 7, 22), 2389.0, 'ML expert', 'Thessaloniki,Greece',
'hlgeor', datetime.datetime(2021, 2, 11, 20, 52, 56), datetime.date(2019, 10, 29)),

(369, datetime.date(2019, 2, 1), 2600.0, 'Algorithmic efficiency expert', 'Sofia,
Bulgaria', 'efstang', datetime.datetime(2018, 11, 1, 0, 0), datetime.date(2019, 1,
16)),

(789, datetime.date(2018, 12, 25), 2700.0, 'NLP expert', 'Peiraias, Greece',
'efstang', datetime.datetime(2018, 10, 10, 0, 0), datetime.date(2018, 11, 10))]

Εμφανίζεται επίσης και ένα παράθυρο στην οθόνη του που φαίνεται παρακάτω:



Αν ο αξιολογητής πατήσει το κουμπί που φαίνεται στο πιο πάνω παράθυρο εμφανίζεται ένα νέο που είναι το ακόλουθο:

A screenshot of a web application window. The title bar at the top reads "Update promotion jobs you have announced." and includes standard window control icons. The main content area has a light gray background. At the top, it says "Fill out the next fields to update a Promotion Job." followed by "Enter the id of the Promotion Job you want to update:". Below this is a text input field. Further down, there are five labeled text input fields: "Start date:", "Salary:", "Position:", "Edra:", and "Submission date:". At the bottom center, there is a button labeled "Update".

Ο χρήστης θα δίνει τον κωδικό της θέσης προαγωγής την οποία θέλει να επεξεργαστεί και στην συνέχεια θα συμπληρώνει τα νέα στοιχεία. Τα στοιχεία θα

ενημερώνονται στην βάση δεδομένων staffevaluation με την χρήση της stored procedure **evaluator_updatejobpromotion** την οποία καλούμε στην Python.

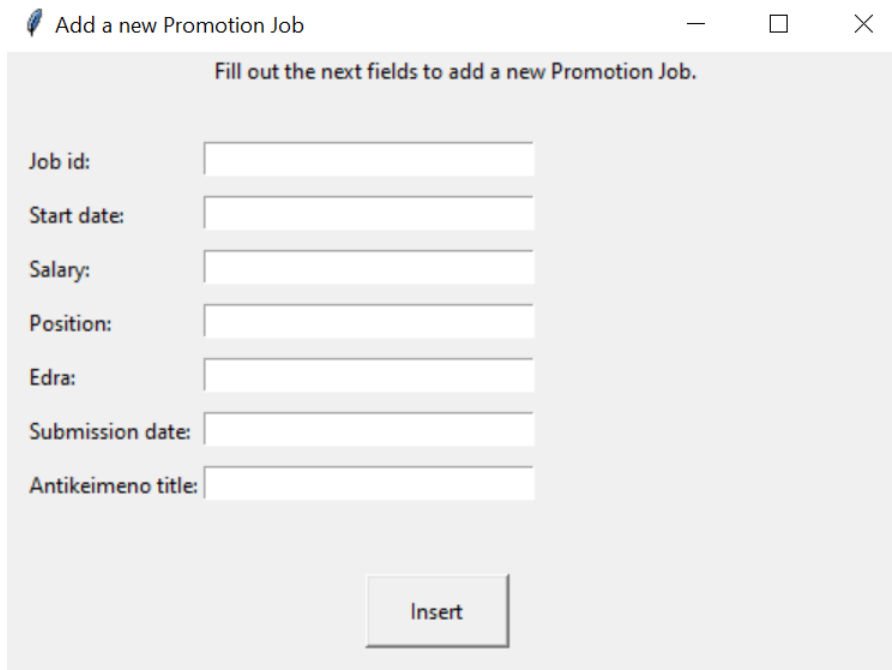
```
CREATE PROCEDURE evaluator_updatejobpromotion(IN PJid int,IN pjstart_date  
date,IN pjsalary float,IN pjposition varchar(40),IN pjedra varchar(45),IN  
eval_username varchar(12) ,IN pjsubmission_date date)
```

```
BEGIN
```

```
UPDATE job SET start_date=pjstart_date ,salary=pjsalary, position=pjposition,  
edra=pjedra, announce_date=CURRENT_TIMESTAMP,  
submission_date=pjsubmission_date WHERE id=PJid AND  
evaluator=eval_username;
```

```
END $
```

- 3) Αν επιλέξει την τρίτη επιλογή τότε εμφανίζεται ένα παράθυρο που φαίνεται παρακάτω :



— □ ×

Fill out the next fields to add a new Promotion Job.

Job id:

Start date:

Salary:

Position:

Edra:

Submission date:

Antikeimeno title:

Insert

Ο χρήστης συμπληρώνει τα κενά για να εισάγει μια νέα θέση προαγωγής και πατώντας το κουμπί insert τα νέα στοιχεία εισάγονται στην βάση δεδομένων. Αυτό

γίνεται με την χρήση μιας **stored procedure** με όνομα **insertjobpromotion** που την καλούμε στην Python.

```
CREATE PROCEDURE insertjobpromotion(IN id int,IN start_date date,IN salary
float,IN position varchar(40),IN edra varchar(45),IN eval_username varchar(12) ,IN
submission_date date,IN antikeim_title varchar(36))

BEGIN

    DECLARE etaireia char(9);

    DECLARE manager varchar(12);

    SELECT firm INTO etaireia FROM evaluator WHERE
evaluator.username=eval_username;

    SELECT managerUsername into manager from manager where firm=etaireia;

    INSERT INTO job

    VALUES(id,start_date,salary,position,edra,eval_username,CURRENT_TIMEST
AMP,submission_date);

    INSERT INTO promotion

    VALUES(manager,eval_username,id);

    INSERT INTO needs

    VALUES(id,antikeim_title);

END $
```

- 4) Σε περίπτωση που ο evaluator επιλέξει την τέταρτη επιλογή θα μπορεί να δει τις θέσεις εργασίες που έχει ανακοινώσει ο ίδιος στην Python με κλήση μιας εντολής select στην sql.

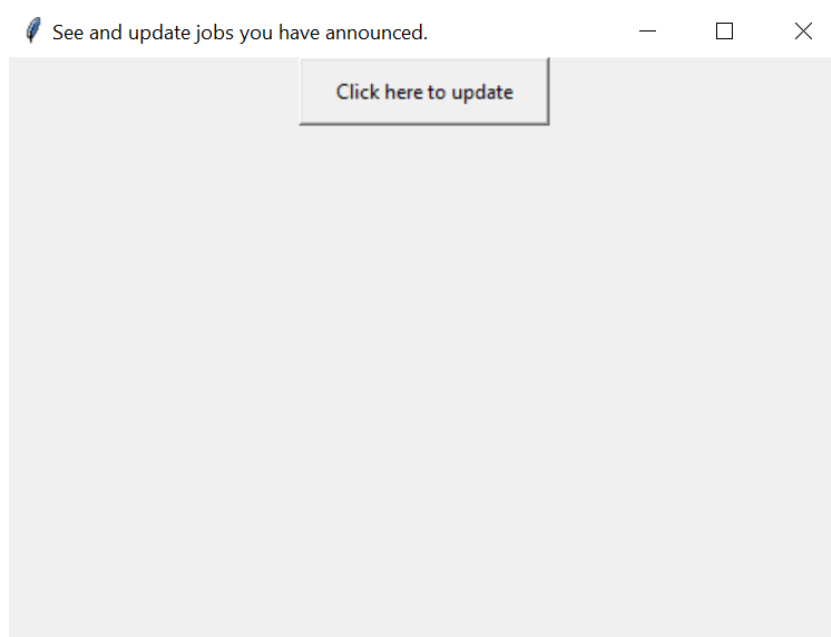
Π.χ. (1, datetime.date(2019, 1, 1), 1800.0, 'data analyst', 'Patra, Greece', 'hlgeor', datetime.datetime(2018, 7, 13, 10, 0), datetime.date(2018, 12, 20))

(272, datetime.date(2018, 7, 22), 2389.0, 'ML expert', 'Thessaloniki,Greece', 'hlgeor', datetime.datetime(2021, 2, 11, 20, 52, 56), datetime.date(2019, 10, 29))

(987, datetime.date(2019, 5, 1), 1600.0, 'graphics expert', 'Athina, Greece', 'hlgeor', datetime.datetime(2018, 11, 20, 0, 0), datetime.date(2019, 4, 12))

Τα παραπάνω αποτελέσματα φαίνονται στην Python για τον evaluator με username hlgeor που μπήκε στο σύστημα.

Εμφανίζεται ένα παράθυρο στην οθόνη του αξιολογητή που είναι το εξής :



Αν πατήσει πάνω στο κουμπί που φαίνεται στο παράθυρο ανοίγει ένα νέο παράθυρο και είναι αυτό που ακολουθεί:

A screenshot of a web application window titled "Update jobs you have announced." with standard window controls. The main content area is a light gray rectangle. At the top, it says "Fill out the next fields to update a Job." followed by "Enter the id of the job you want to update:". Below this is a text input field. Further down, there are five labeled input fields: "Start date:", "Salary:", "Position:", "Edra:", and "Submission date:". At the bottom center, there is an "Update" button.

Έτσι ο υπεύθυνος αξιολόγησης μπορεί να επεξεργαστεί τις θέσεις εργασίες που έχει προσθέσει στο σύστημα (συμπληρώνοντας τα κενά και πατώντας το κουμπί Update).

Οι αλλαγές των jobs στην βάση γίνονται με μια **stored procedure** **updatejobs_youhaveannounced** την οποία καλούμε στην Python.

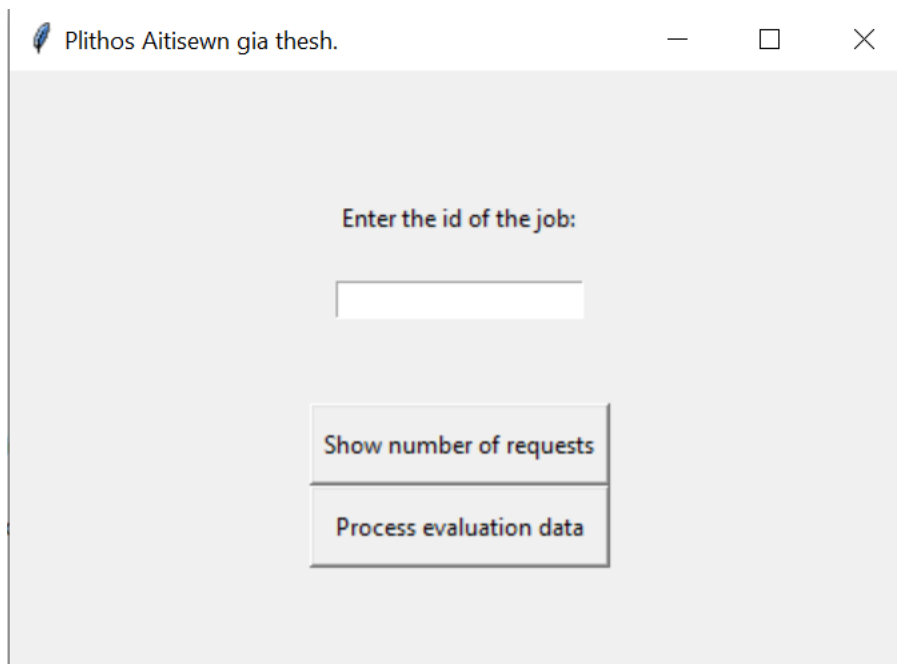
```
CREATE PROCEDURE updatejobs_youhaveannounced(IN Jid int,IN jstart_date date,IN
jsalary float,IN jposition varchar(40),IN jedra varchar(45),IN eval_username
varchar(12) ,IN jsubmission_date date)
```

```
BEGIN
```

```
UPDATE job SET start_date=jstart_date ,salary=jsalary, position=jposition,
edra=jedra, announce_date=CURRENT_TIMESTAMP,
submission_date=jsubmission_date WHERE id=Jid AND evaluator=eval_username;
```

```
END $
```

5) Αν επιλέξει το πέμπτο κουμπάκι στην σειρά ανοίγει το παρακάτω παράθυρο:



The screenshot shows a web application window with the title "Plithos Aitisewn gia thesh.". Inside the window, there is a text prompt "Enter the id of the job:" followed by a text input field. Below the input field, there are two buttons stacked vertically: "Show number of requests" and "Process evaluation data".

Ο evaluator θα δίνει τον κωδικό μιας θέσης εργασίας που έχει αναρτήσει ο ίδιος και πατώντας κάποιο από τα 2 κουμπιά που φαίνονται θα μπορεί να δει το πλήθος των αιτήσεων για αυτή την θέση ή να επεξεργαστεί δεδομένα αξιολογήσεις για αυτή την θέση.

Πατώντας το κουμπί Show number of requests εμφανίζονται στην Python τα αποτελέσματα δηλαδή ο αριθμός των αιτήσεων πχ για την job κωδικό 1:

```
[('No. of Evaluation Requests',)]
[(2,)]
[('No. of Promotion Requests',)]
[(2,)]
```

Αυτό το πετυχαίνουμε με την κλήση μιας **stored procedure** στην Python με όνομα showrequestcount :

```
CREATE PROCEDURE showrequestcount(IN thesi_ergasias INT)
```

```
BEGIN
```

```
    SELECT 'No. of Evaluation Requests';
```

```
    SELECT COUNT(empl_username) AS 'No. of Evaluation Requests' FROM
requestsevaluation WHERE job_id=thesi_ergasias;
```

```
    SELECT 'No. of Promotion Requests';
```

```
    SELECT COUNT(emplo_username) AS 'No. of Promotion Requests' FROM
requestspromotion WHERE jobID=thesi_ergasias;
```

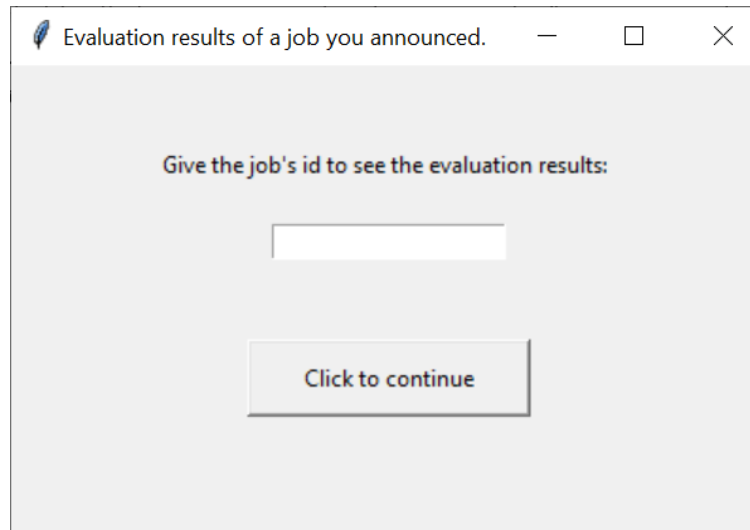
```
END $
```

Αν πατήσει το κουμπί Process evaluation data τότε εμφανίζεται το παράθυρο:

Έτσι ο evaluator μπορεί να αλλάξει τα δεδομένα αξιολόγησης στον πίνακα evaluation πατώντας το κουμπί Process Evaluation Data και ταυτόχρονα όταν

πατήσει το κουμπί καλείται και η **stored 3.2** με την οποία οριστικοποιείται το αποτέλεσμα.

- 6) Όταν ο αξιολογητής επιλέξει το έκτο στην σειρά κουμπί θα εμφανιστεί το εξής παράθυρο:



The image shows a web browser window with the title "Evaluation results of a job you announced." Inside the window, there is a message "Give the job's id to see the evaluation results:" followed by a text input field. Below the input field is a button labeled "Click to continue".

Ο evaluator θα δίνει το id μιας θέσης εργασίας που έχει αναρτήσει ο ίδιος και με την κλήση της **stored procedure 3.3** (πατώντας το κουμπί Click to continue) θα εμφανίζονται στην Python τα ενδιάμεσα και τελικά αποτελέσματα αξιολόγησης:

Πχ για την δουλειά με id 1:

```
[('Οριστικοποιημένοι πίνακες.',)]  
[(20, 'mnikol', 1, 8, 'good')]  
[(('abrown', 'hlgeor', 1, None, '3', '1', 'good', 23), ('lionarF', 'hlgeor', 1, '3', '1', None, 'not good', 33))]  
[('Aksiologisi se ekseleksi...ekremmoun aitiseis',)]  
[(2,)]
```

- 7) Αν πατήσει το τελευταίο κουμπί επίλογων τότε στην python θα δει τα ζητούμενα αποτελέσματα δηλαδή όλες τις θέσεις που έχουν αναρτηθεί στην εταιρεία του .

Πχ για τον evaluator hlgeor:

```
[(369, datetime.date(2019, 2, 1), 2600.0, 'Algorithmic efficiency expert',  
'Sofia, Bulgaria', 'efstang', datetime.datetime(2018, 11, 1, 0, 0),  
datetime.date(2019, 1, 16)),
```

(789, datetime.date(2018, 12, 25), 2700.0, 'NLP expert', 'Peiraias, Greece',
'efstang', datetime.datetime(2018, 10, 10, 0, 0), datetime.date(2018, 11, 10)),

(1, datetime.date(2019, 1, 1), 1800.0, 'data analyst', 'Patra, Greece', 'hlgeor',
datetime.datetime(2018, 7, 13, 10, 0), datetime.date(2018, 12, 20)),

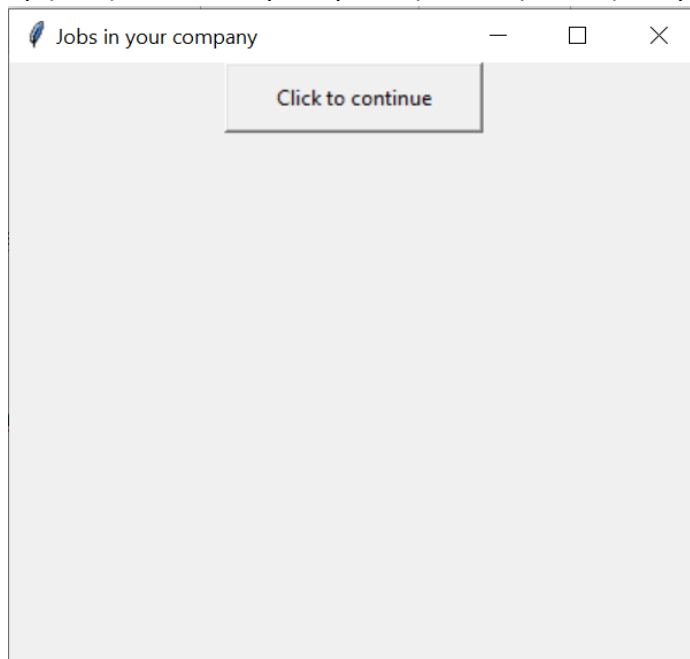
(272, datetime.date(2018, 7, 22), 2389.0, 'ML expert', 'Thessaloniki,Greece',
'hlgeor', datetime.datetime(2021, 2, 11, 20, 52, 56), datetime.date(2019, 10,
29)),

(987, datetime.date(2019, 5, 1), 1600.0, 'graphics expert', 'Athina, Greece',
'hlgeor', datetime.datetime(2018, 11, 20, 0, 0), datetime.date(2019, 4, 12))]

Η εκτύπωση αυτή έγινε με την κλήση της **stored procedure**
jobsinevaluatorscompany την οποία καλούμε στην Python.

```
CREATE PROCEDURE jobsinevaluatorscompany(IN eval_username  
varchar(12))  
BEGIN  
    DECLARE etairia char(9);  
    SELECT firm INTO etairia FROM evaluator WHERE  
username=eval_username;  
    SELECT  
id,start_date,salary,position,edra,evaluator,announce_date,submission_date  
FROM job INNER JOIN evaluator ON evaluator.username=evaluator WHERE  
firm=etairia;  
  
END $
```


Εμφανίζεται ένα παράθυρο στην οθόνη του αξιολογητή:



Όταν πατήσει το κουμπί που υπάρχει στο παράθυρο αυτό εμφανίζεται ένα νέο στο οποίο εισάγει τα στοιχεία που θέλει να αλλάξει για τις θέσεις που αυτός έχει αναρτήσει.

A screenshot of a web application window titled "Update a job you have announced". The window has a light gray background and standard window controls. The main content area contains the following elements:

- Text: "Fill out the next fields to update a Job."
- Text: "Enter the id of the job you want to update:"
- A single-line text input field.
- A label "Start date:" followed by a single-line text input field.
- A label "Salary:" followed by a single-line text input field.
- A label "Position:" followed by a single-line text input field.
- A label "Edra:" followed by a single-line text input field.
- A label "Submission date:" followed by a single-line text input field.
- A button labeled "Update" at the bottom center.

Πατώντας το κουμπί update τα στοιχεία του δοσμένου από τον ίδιο job αλλάζουν στην βάση δεδομένων.

Αν ο χρήστης που θα συνδεθεί είναι ο υπάλληλος /**employee** εμφανίζεται το παρακάτω GUI στην οθόνη με τις επιλογές που μπορεί να κάνει:

Employee Options elenineo

Choose one option:

- ☐ See your file and change your password and bio.
- ☐ Request for an available promotion job.
- ☐ Show all requests(promotion and evaluation).
- ☐ Delete an evaluation request if the job is still available.

[Click here to continue](#)

- 1) Αν επιλέξει την πρώτη επιλογή ώστε να δει και να επεξεργαστεί μερικά από τα στοιχεία του εμφανίζεται το εξής:

```
[('elenineo', 2, 'bio3', 'sistatikes3', 'certificates3', 'awards3', '123432211')]  
[('elenineo', '369', 'Eleni', 'Neofytou', datetime.datetime(2016, 1, 6, 14, 50), 'neofytoue@gmail.com')]
```

Τα παραπάνω στοιχεία είναι παράδειγμα από τον χρήστη elenineo που εισήλθε στο σύστημα, και εμφανίζονται στο terminal της python. Επίσης τα στοιχεία εμφανίζονται με την χρήση μίας **stored procedure** :

DELIMITER \$

CREATE PROCEDURE blepeifakelo (IN employeeusername varchar(12))

BEGIN

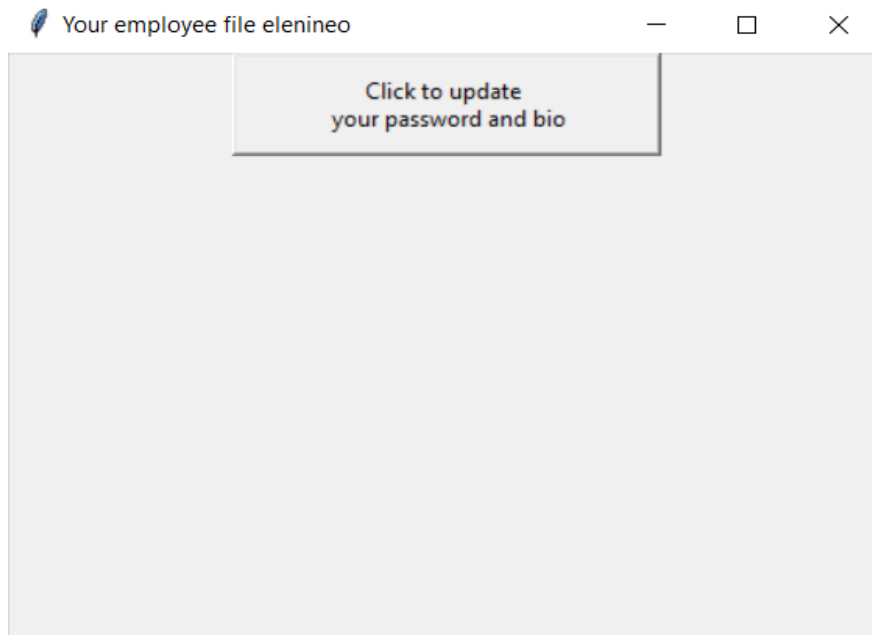
SELECT employee.username,exp_years,bio,sistatikes,certificates,awards,firm
FROM employee INNER JOIN user ON user.username=employee.username WHERE
user.username=employeeusername;

SELECT username,password,name,surname,reg_date,email FROM user
WHERE user.username=employeeusername;

END \$

DELIMITER ;

Για να αλλάξει τα στοιχεία του εμφανίζεται ένα νέο παράθυρο



Και πατώντας το κουμπί Click to update your password and bio εμφανίζεται το παράθυρο για να εισάγει το νέο κωδικό και βιογραφικό του.

A screenshot of a web application window titled 'Change password and bio'. The title bar includes a feather icon, the title text, and window controls. The main content area has a light gray background. At the top of the content area, it says 'Enter the details to be changed in your profile:'. Below this, there are two input fields. The first is labeled 'Password:' and the second is labeled 'Bio:'. At the bottom center of the form, there is a button labeled 'Change'.

Αυτά αλλάζουν στην βάση δεδομένων με την χρήση του κουμπιού Change.

- 2) Αν επιλέξει την δεύτερη επιλογή να κάνει δηλαδή αίτηση για θέση/θέσεις προαγωγής εμφανίζεται το εξής:

Εισάγοντας ένα id μιας θέσης προαγωγής και πατώντας το κουμπί καλείτε μια **stored procedure**

```
DELIMITER $
CREATE PROCEDURE requestforjob_promotion(IN username_employee
varchar(12),IN id_thesispromotion INT)
BEGIN
    IF( username_employee NOT IN (SELECT emplo_username FROM
requestspromotion WHERE jobID=id_thesispromotion) AND
id_thesispromotion NOT IN(SELECT jobID FROM requestspromotion WHERE
emplo_username= username_employee )) THEN
        INSERT INTO requestspromotion
VALUES(username_employee,id_thesispromotion);
    ELSE
        SELECT 'You have requested for this job.';
    END IF;
END $
DELIMITER ;
```

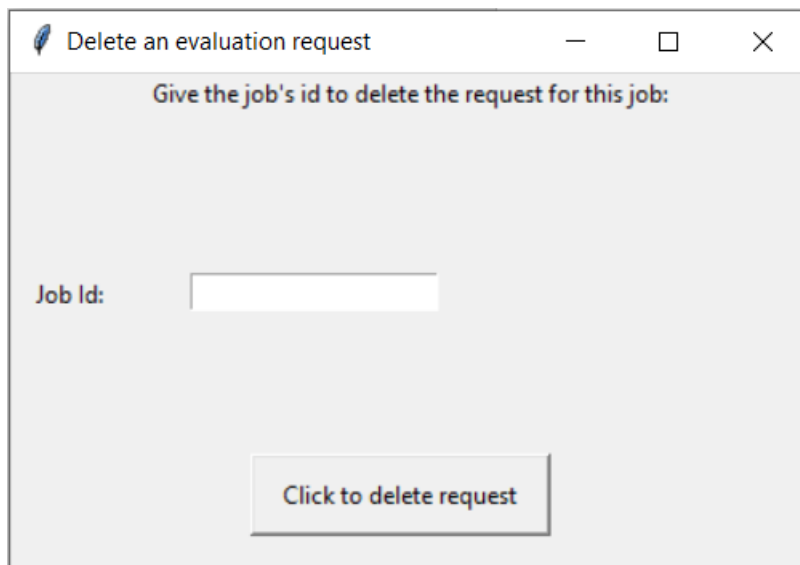
Και αυτόματα εισάγεται το όνομα του **employee** και το **id** της θέσης στον πίνακα requestpromotion αν δεν έχει ήδη κάνει αίτηση αυτός ο employee. Αν έχει κάνει εμφανίζει το μήνυμα στην Python **'You have requested for this job.'**

- 3) Αν επιλέξει την τρίτη επιλογή να δει όλες τις αιτήσεις που έχει υποβάλει στο σύστημα εμφανίζονται τα αποτελέσματα στο terminal της Python:

```
Evaluation requests:
('elenineo', 123)
Promotion requests:
('elenineo', 1)
('elenineo', 111)
('elenineo', 123)
('elenineo', 456)
('elenineo', 654)
('elenineo', 789)
('elenineo', 963)
```

Τα παραπάνω στοιχεία είναι παράδειγμα από τον χρήστη elenineo που εισήλθε στο σύστημα.

- 4) Αν επιλέξει την τέταρτη επιλογή για να αποσύρει μια υποψηφιότητα για μια θέση εμφανίζεται το εξής:

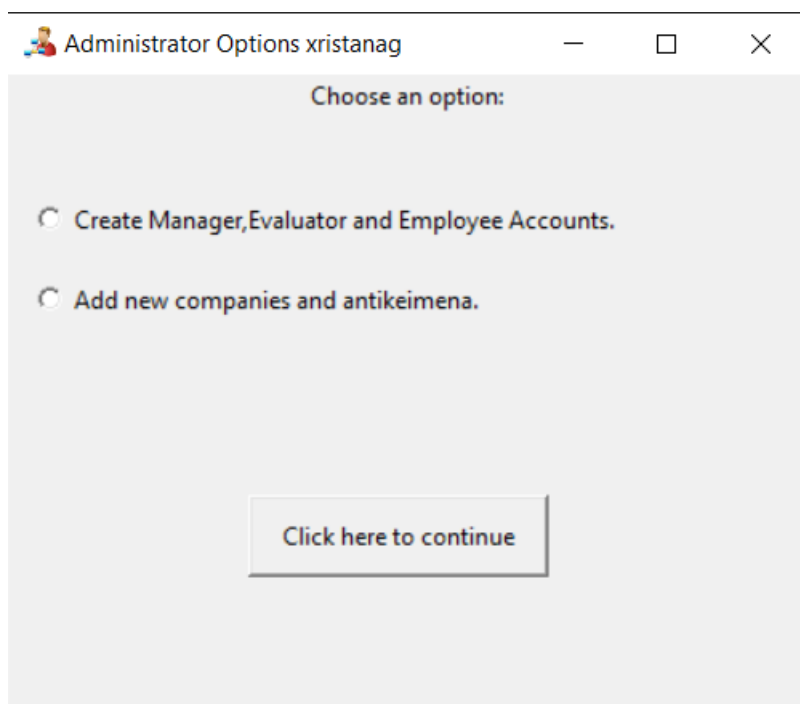


The screenshot shows a web application window with the title "Delete an evaluation request". Inside the window, there is a prompt: "Give the job's id to delete the request for this job:". Below this prompt, there is a text input field labeled "Job Id:". At the bottom of the window, there is a button labeled "Click to delete request".

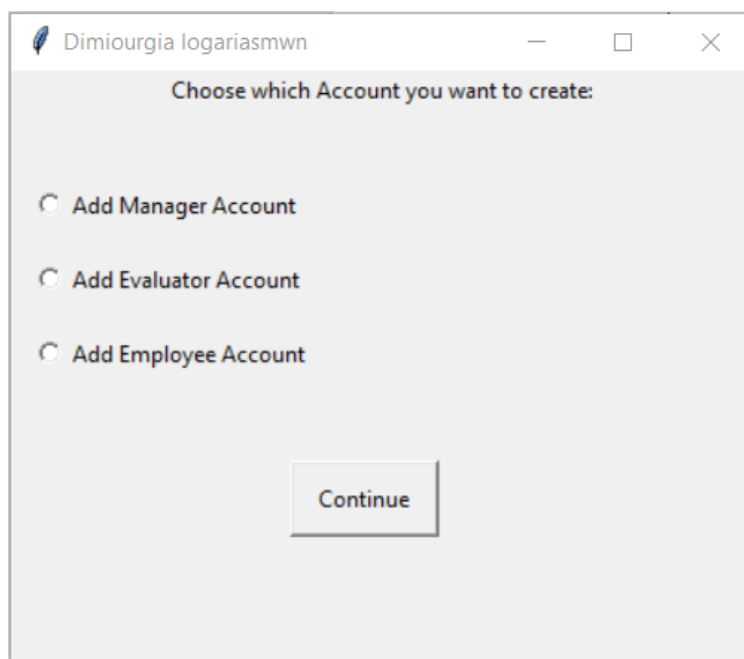
Ο χρήστης εισάγει το id της θέσης που επιθυμεί να διαγράψει την αίτηση και στην συνέχεια πατάει το κουμπί Click to delete request.

Αν ο χρήστης που θα συνδεθεί είναι ο διαχειριστής/**administrator** εμφανίζεται το παρακάτω GUI στην οθόνη με τις επιλογές που μπορεί να κάνει:

- 1) Αν επιλέξει την πρώτη επιλογή για να δημιουργήσει νέος λογαριασμούς στο σύστημα για Διευθυντές Αξιολογητές και Υπαλλήλους εμφανίζεται το εξής:



Αν επιλέξει την πρώτη επιλογή εμφανίζεται:



Εάν επιλέξει να προσθέσει manager δηλαδή το πρώτο εμφανίζεται το παράθυρο :

Add Manager Account

Fill out the next fields to insert a new Manager Account.

Username:

Password:

Name:

Surname:

Email:

Years of experience:

Company:

Και συμπληρώνει στα κενά τα στοιχεία ενός νέου manager και πατώντας το κουμπί Insert καλείται μια **stored procedure**:

DELIMITER \$

```
CREATE PROCEDURE admininsert_manager(IN man_username varchar(12),IN
password varchar(10),IN name varchar(25),IN surname varchar(35) ,IN email
varchar(30),IN exp_years INT,IN firm char(9))
```

```
BEGIN
```

```
    INSERT INTO user
```

```
    VALUES(man_username,password,name,surname,CURRENT_TIMESTAMP,em
ail);
```

```
    INSERT INTO manager
```

```
    VALUES(man_username,exp_years,firm);
```

```
END $
```

```
DELIMITER ;
```

Εάν επιλέξει να δημιουργήσει έναν evaluator δηλαδή την δεύτερη επιλογή εμφανίζεται :

Add Evaluator Account

Fill out the next fields to insert a new Evaluator Account.

Username:

Password:

Name:

Surname:

Email:

Years of experience:

Company:

Και συμπληρώνει στα κενά τα στοιχεία ενός νέου evaluator και πατώντας το κουμπί Insert καλείται μια **stored procedure**:

DELIMITER \$

```
CREATE PROCEDURE admininsert_evaluator(IN eval_username varchar(12),IN
eval_password varchar(10),IN eval_name varchar(25),IN eval_surname
varchar(35) ,IN eval_email varchar(30),IN eval_exp_years INT,IN eval_firm
char(9))
```

```
BEGIN
```

```
    INSERT INTO user
```

```
    VALUES(eval_username,eval_password,eval_name,eval_surname,CURRENT_
TIMESTAMP,eval_email);
```

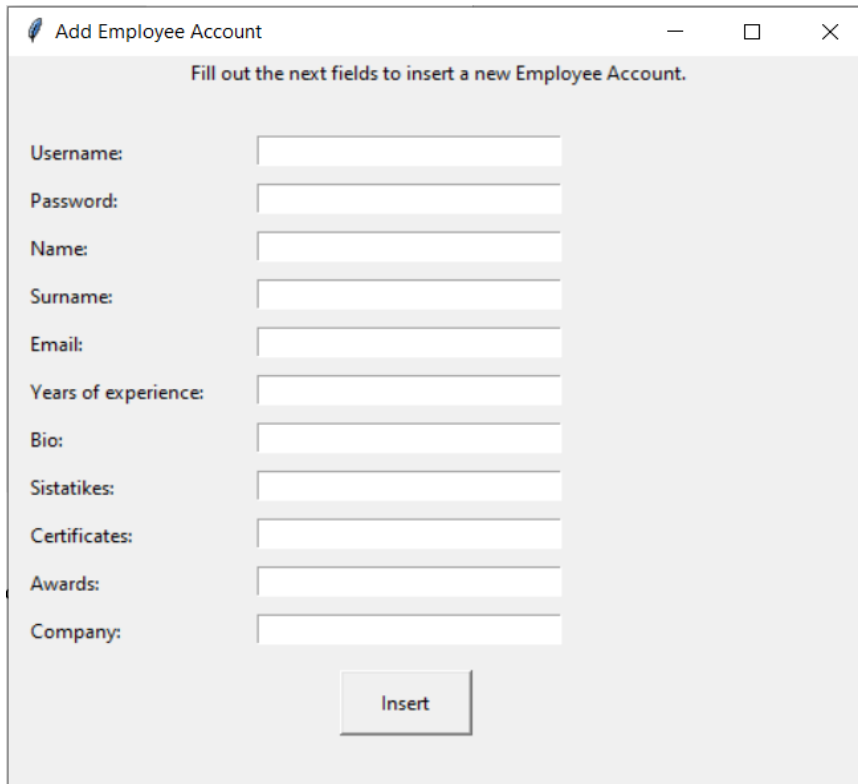
```
    INSERT INTO evaluator
```

```
    VALUES(eval_username,eval_exp_years,eval_firm);
```

```
END $
```

```
DELIMITER ;
```


Αντίστοιχα εάν επιλέξει να δημιουργήσει έναν νέο employee δηλαδή την τρίτη επιλογή εμφανίζεται το εξής:



Και συμπληρώνει στα κενά τα στοιχεία ενός νέου employee και με το πάτημα του κουμπιού Insert καλείται μια **stored procedure**:

DELIMITER \$

```
CREATE PROCEDURE admininsert_employee(IN e_username varchar(12),IN  
e_password varchar(10),IN e_name varchar(25),IN e_surname varchar(35) ,IN  
e_email varchar(30),IN e_exp_years INT,IN e_bio text,IN e_sistatikes  
varchar(35),IN e_certificates varchar(35),IN e_awards varchar(35),IN e_firm  
char(9))
```

```
BEGIN
```

```
    INSERT INTO user
```

```
    VALUES(e_username,e_password,e_name,e_surname,CURRENT_TIMESTAMP,  
P,e_email);
```

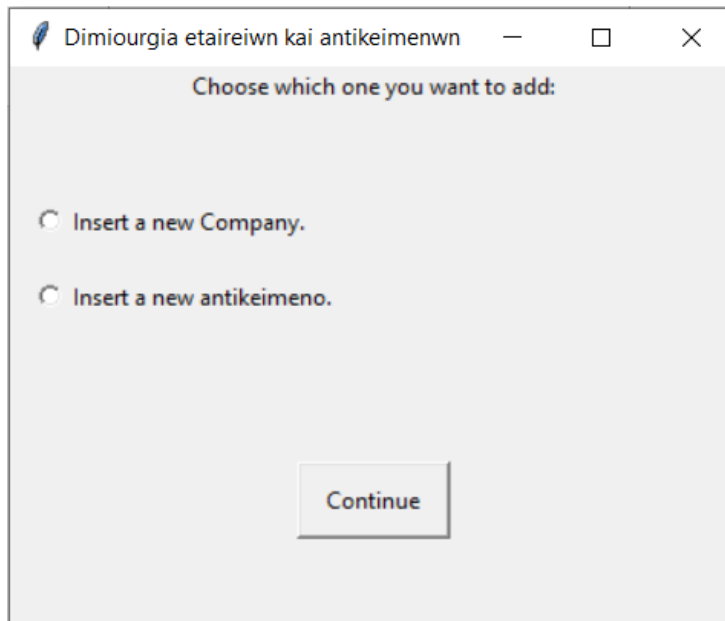
```
    INSERT INTO employee
```

```
    VALUES(e_username,e_exp_years,e_bio,e_sistatikes,e_certificates,e_awards,  
e_firm);
```

END \$

DELIMITER ;

- 2) Αν επιλέξει την δεύτερη επιλογή δηλαδή για να δημιουργήσει νέες εταιρείες, αντικείμενα και τομείς.



Dimiourgia etaireiwn kai antikeimenwn

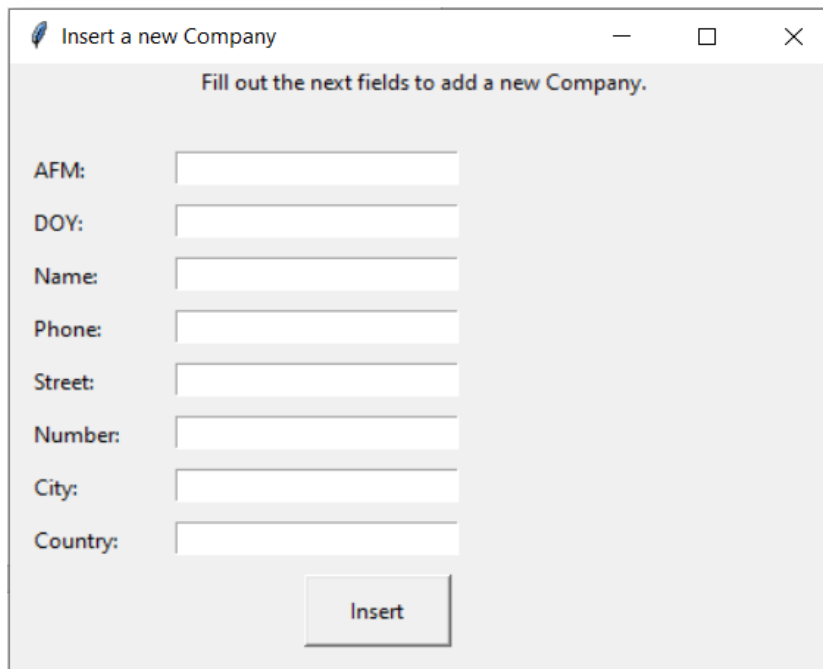
Choose which one you want to add:

☐ Insert a new Company.

☐ Insert a new antikeimeno.

Continue

Εάν επιθυμεί να εισάγει εταιρεία επιλέγει το πρώτο και εμφανίζεται:



Insert a new Company

Fill out the next fields to add a new Company.

AFM:

DOY:

Name:

Phone:

Street:

Number:

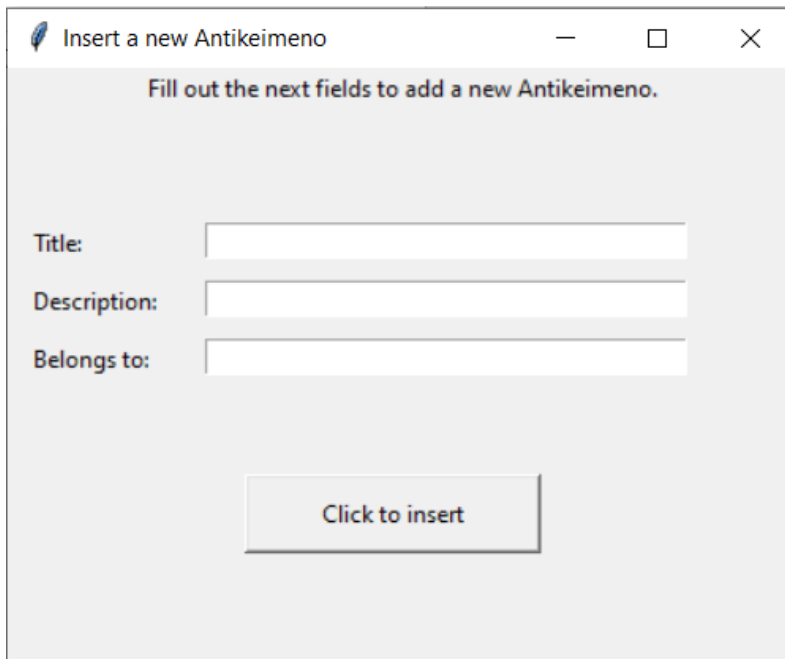
City:

Country:

Insert

Εισάγει τα νέα στοιχεία στα κενά , πατάει το κουμπί Insert και η νέα εταιρεία - company εισάγεται στην βάση δεδομένων.

Από την άλλη αν επιθυμεί να εισάγει αντικείμενα δηλαδή την δεύτερη επιλογή εμφανίζεται:



Insert a new Antikeimeno

Fill out the next fields to add a new Antikeimeno.

Title:

Description:

Belongs to:

Click to insert

Συμπληρώνει τα κενά με στοιχεία για την εταιρεία πατάει το κουμπί Click to insert και έτσι το νέο Antikeimeno εισάγεται στην βάση δεδομένων.