

Ζύμνης Μάρκελλος ΑΜ:1059562 ΈΤΟΣ:Δ'

Σκύρλα Αγάθη ΑΜ:1064888 ΈΤΟΣ:Δ'

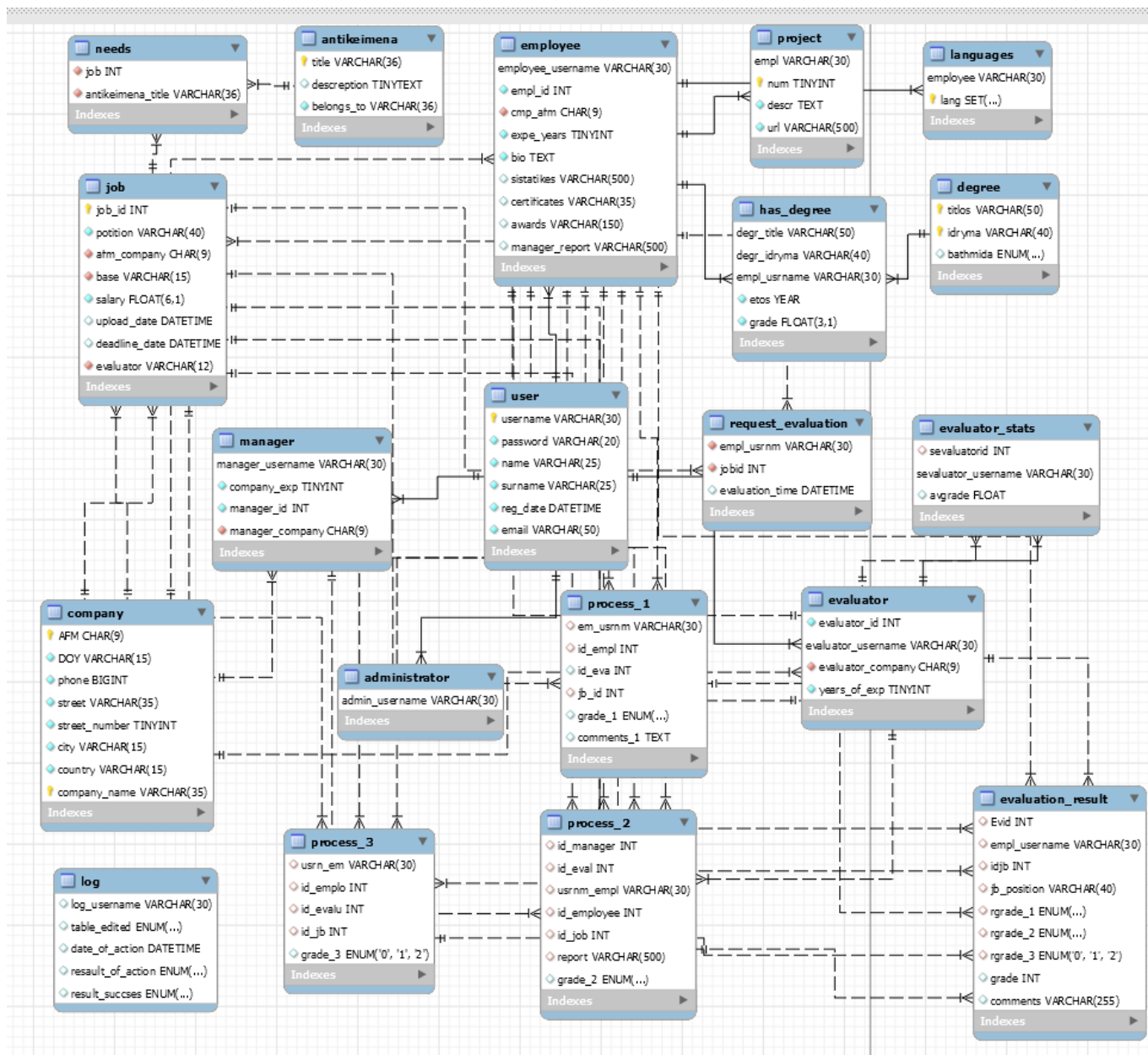
Project Βάσεων Δεδομένων

➤ ΔΙΕΥΚΡΙΝΙΣΕΙΣ :

1)ΤΑ ΣΧΟΛΙΑ ΤΗΣ ΤΕΚΜΗΡΙΩΣΗΣ ΤΟΥ ΚΩΔΙΚΑ ΜΑΣ ΕΙΝΑΙ ΜΕ ΧΡΩΜΑ ΣΙΕΛ.

2)Οποιαδήποτε αλλαγή από το αρχικό διάγραμμα ή από τις οδηγίες έχει δίπλα EXTRA , το οποίο σημαίνει ότι το έχουμε βάσει εμείς για την υλοποίηση της βάσης.

1. ΣΧΕΣΙΑΚΟ ΔΙΑΓΡΑΜΜΑ ΒΑΣΗΣ:



Κώδικας Βάσης από MySQL :

Πίνακες για την Βάση:

Δημιουργία της Βάσης μας : StaffEvaluation

DROP DATABASE IF EXISTS StaffEvaluation; *Αν υπάρχει η βάση ήδη κάνει DROP την υπάρχον Βάση.*

CREATE DATABASE StaffEvaluation;

USE StaffEvaluation;

CREATE TABLE `user`

(

username VARCHAR(30) NOT NULL,

`password` VARCHAR(20) NOT NULL,

`name` VARCHAR(25) NOT NULL,

surname VARCHAR(25) NOT NULL,

reg_date DATETIME NOT NULL,

email VARCHAR(50) NOT NULL,

PRIMARY KEY (username))engine=InnoDB; *Βάζουμε πρωτεύον κλειδί το username.*

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE company

(

AFM CHAR(9) NOT NULL,

DOY VARCHAR(15) NOT NULL,

phone BIGINT(12) NOT NULL,

street VARCHAR(35) NOT NULL,

street_number TINYINT(4) NOT NULL,

city VARCHAR(15) NOT NULL,

country VARCHAR(15) NOT NULL,

company_name VARCHAR(35) NOT NULL,

PRIMARY KEY (AFM,company_name), *Βάζουμε πρωτεύων κλειδί το ΑΦΜ και το όνομα της εταιρείας.*

KEY(city) *Θέτουμε σαν απλό κλειδί την πόλη της εταιρείας, ώστε έπειτα να μπορεί να κάνει Constraint με την δουλειά.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE employee

(

employee_username VARCHAR(30) NOT NULL,

empl_id INT(4) NOT NULL auto_increment , (extra)Βάζουμε για κάθε υπάλληλο ένα μοναδικό ID για το σύστημα μας , το οποίο με κάθε νέο υπάλληλο θα αυξάνετε κατά 1 αυτόματα .

cmp_afm CHAR(9) NOT NULL, (extra)Με το cmp_afm θέτουμε το ΑΦΜ της εταιρείας που δουλεύει ο κάθε υπάλληλος,

expe_years TINYINT(4) NOT NULL,

bio TEXT NOT NULL ,

sistatikes VARCHAR(500) ,

certificates VARCHAR(35),

awards VARCHAR(150),

manager_report VARCHAR(500), (extra)Βάζουμε ένα report στον employee, το οποίο μέσα στο GUI θα το επεξεργάζεται ο manager για να συνδεθεί με το process_2(αξιολόγηση με βάση το report του manager).

PRIMARY KEY (employee_username), Πρωτεύον κλειδί είναι το όνομα του υπαλλήλου.

UNIQUE KEY(empl_id), Μοναδικό κλειδί είναι το ID του. Δεν πρέπει δύο υπάλληλοι να έχουν ίδιο ID.

KEY(manager_report), Θέτουμε σαν κλειδί το manager_report ώστε έπειτα να κάνει Constraint με το process_2 στο report.

CONSTRAINT EMPL_USER FOREIGN KEY (employee_username) REFERENCES `user`(username) ON DELETE CASCADE ON UPDATE CASCADE , Κάνουμε CONSTRAINT στο employee_username από τον user (username), ώστε να αναγνωρίζει το username από τον χρήστη.

CONSTRAINT COMP_AFM FOREIGN KEY (cmp_afm) REFERENCES company(AFM) ON DELETE CASCADE ON UPDATE CASCADE Κάνουμε CONSTRAINT στο cmp_afm από το company (AFM), ώστε να αναγνωρίζει το ΑΦΜ από τον πίνακα εταιρεία.

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα. Αλλά θεωρήσαμε ότι τα sistatikes, certificates, awards, manager_report μπορούν να είναι NULL, καθώς κάποιος εργαζόμενος μπορεί να μην έχει κάποιο από αυτά τα πεδία.

CREATE TABLE evaluator

(

evaluator_id INT(4) NOT NULL AUTO_INCREMENT, (extra)Βάζουμε μοναδικό ID σε κάθε evaluator για να αναγνωρίζει το σύστημα μας, το οποίο με κάθε εισαγωγή καινούργιου evaluator θα αυξάνετε αυτόματα κατά 1.

evaluator_username VARCHAR(30) NOT NULL,

evaluator_company CHAR(9) NOT NULL, Ως evaluator_company θέσαμε το ΑΦΜ κάθε εταιρίας που ανήκει ο κάθε evaluator(ουσιαστικά είναι το firm απλά αλλάξαμε το όνομα για μας βοηθήσει).

years_of_exp TINYINT(4) NOT NULL,

PRIMARY KEY (evaluator_username), Πρωτεύον κλειδί θέτουμε το evaluator_username.

UNIQUE KEY(evaluator_id), Όπως αναφέραμε θέτουμε μοναδικό κλειδί το ID του evaluator για να κάνει Constraint με άλλους πίνακες.

CONSTRAINT EVALUATOR_USERNAME FOREIGN KEY (evaluator_username) REFERENCES `user`(username) ON DELETE CASCADE ON UPDATE CASCADE , Κάνουμε Constraint το evaluator_username του evaluator με το username από τον πίνακα user.

CONSTRAINT EVALUATOR_COMPANY FOREIGN KEY (evaluator_company) REFERENCES company(AFM) ON DELETE CASCADE ON UPDATE CASCADE Κάνουμε Constraint το evaluator_company με το company AFM ώστε να αναγνωρίζει το ΑΦΜ της εταιρίας από τον πίνακα εταιρία .

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα

CREATE TABLE job

(

job_id INT(4) NOT NULL AUTO_INCREMENT, Θέτουμε ID μοναδικό για κάθε δουλειά το οποίο με κάθε νέα εισαγωγή νέας δουλειάς θα αυξάνετε αυτόματα κατά 1.

position VARCHAR(40) NOT NULL, Ουσιαστικά το όνομα της θέσης εργασίας .

afm_company CHAR(9) NOT NULL, (extra)Ως afm_company θέσαμε το ΑΦΜ κάθε εταιρίας που ανήκει η κάθε δουλειά, για να βλέπουμε που ανήκει κάθε δουλειά σε ποια εταιρεία.

base VARCHAR(15) NOT NULL, Θεωρήσαμε σαν base την πόλη που βρίσκεται κάθε δουλειά(ουσιαστικά είναι η edra αλλά αλλάξαμε το όνομα για λόγους βοηθητικούς).

salary FLOAT(6,1) NOT NULL,

upload_date DATETIME,

deadline_date DATETIME,

evaluator VARCHAR(12) NOT NULL, Ποιος evaluator θα αξιολογήσει τους υποψήφιους για την συγκεκριμένη δουλειά.

PRIMARY KEY (job_id), Θέτουμε ως πρωτεύον κλειδί το ID κάθε δουλειάς.

CONSTRAINT JOB_AFM FOREIGN KEY(afm_company) REFERENCES company(AFM) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint afm_company με το AFM από τον πίνακα company, ώστε να αναγνωρίζει το ΑΦΜ της εταιρίας από τον πίνακα εταιρία .

CONSTRAINT JOB_CITY FOREIGN KEY (base) REFERENCES company(city) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το base στο city από τον πίνακα company, ώστε να αναγνωρίζει την πόλη στην οποία ανήκει η δουλειά που ανακοινώνεται δηλαδή είναι η πόλη της εταιρείας στην οποία ανήκει.

CONSTRAINT EVALUATOR_IN_CHARGE FOREIGN KEY (evaluator) REFERENCES evaluator(evaluator_username) ON DELETE CASCADE ON UPDATE CASCADE Κάνουμε Constraint το evaluator στο evaluator_username από τον πίνακα evaluator, ώστε να αναγνωρίζει το username του evaluator ο οποίος έχει αναλάβει την αξιολόγηση της συγκεκριμένης δουλειάς.

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE antikeimena

(

title VARCHAR(36) NOT NULL,

description TINYTEXT,

belongs_to VARCHAR(36) NOT NULL,

PRIMARY KEY (title) *Βάζουμε πρωτεύον κλειδί τον τίτλο όπως μας έχει δωθεί.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα. Το description δεν είναι απαραίτητο να γεμίσει.

CREATE TABLE needs

(

job INT(4) NOT NULL, *Ουσιαστικά εδώ έχουμε το ID της δουλειάς που θα κάνουμε Constraint για να το αναγνωρίζει από το πίνακα job.*

antikeimena_title VARCHAR(36) NOT NULL,

CONSTRAINT JOB_IN_NEED FOREIGN KEY (job) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE, *Κάνουμε Constraint το job στο job_id από το job πίνακα ,δηλαδή για να πάρουμε το job_id από τον πίνακα job.*

CONSTRAINT ANTIKEIMENO_JOB FOREIGN KEY (antikeimena_title) REFERENCES antikeimena(title) ON DELETE CASCADE ON UPDATE CASCADE *Κάνουμε Constraint το antikeimena_title από τον πίνακα antikeimena στο title, για να παίρνει τον τίτλο του αντικειμένου,δηλαδή από το needs, όπου ανήκει η κάθε δουλειά.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE degree (

titlos VARCHAR(50) NOT NULL,

idryma VARCHAR (40) NOT NULL,

bathmida ENUM ('LYKEIO', 'UNIV', 'MASTER', 'PHD'),

PRIMARY KEY (titlos, idryma) *Θέτουμε ως πρωτεύον κλειδί τα titlos,idryma.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE has_degree (

degr_title VARCHAR(50) NOT NULL,

degr_idryma VARCHAR (40) NOT NULL,

empl_username VARCHAR(30) NOT NULL,

etos YEAR(4) NOT NULL,

grade FLOAT(3,1) NOT NULL,

PRIMARY KEY (degr_title, degr_idryma, empl_username), *Θέτουμε ως πρωτεύοντα κλειδιά τα: degr_title, degr_idryma, empl_username.*

CONSTRAINT HAS_DEGR FOREIGN KEY (degr_title, degr_idryma) REFERENCES degree(titlos, idryma) ON DELETE CASCADE ON UPDATE CASCADE, *Κάνουμε Constraint τα degr_title, degr_idryma στα titlos, idryma από τον πίνακα degree, ώστε να παίρνει τον τίτλο του πτυχίου και το ίδρυμα από το οποίο έχει αποφοιτήσει ο κάθε υπάλληλος.*

CONSTRAINT HAS_EMPL FOREIGN KEY (empl_username) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE *Κάνουμε Constraint το empl_username στο employee-username από τον πίνακα employee, ώστε το has_degree να παίρνει το όνομα του κάθε υπαλλήλου.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE project(

empl VARCHAR(30) NOT NULL,

num TINYINT(4) NOT NULL,

descr TEXT NOT NULL,

url VARCHAR(500) NOT NULL,

PRIMARY KEY (empl, num),

CONSTRAINT PROJ_EMPL FOREIGN KEY (empl) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE *Κάνουμε Constraint το empl στο employee_username από τον πίνακα employee, ώστε το κάθε project να έχει το όνομα του υπαλλήλου από τον οποίο έχει υλοποιηθεί.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE languages(

employee VARCHAR(30) NOT NULL,

lang SET('EN', 'FR', 'SP', 'GR'),

PRIMARY KEY (employee, lang), *Θέτουμε ως πρωτεύον κλειδί τα employee και lang όπως μας έχει δοθεί.*

CONSTRAINT EMPL_LANG FOREIGN KEY (employee) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

CREATE TABLE administrator *EXTRA πίνακας, ο οποίος είναι για τον admin του συστήματος .*

(

admin_username VARCHAR(30) NOT NULL, *Βάζουμε το username του admin.*

PRIMARY KEY(admin_username), *Θέτουμε ότι είναι πρωτεύον κελιδί το admin_username.*

CONSTRAINT ADMIN_USRNM FOREIGN KEY (admin_username) REFERENCES `user`(username) ON DELETE CASCADE ON UPDATE CASCADE *Κάνουμε Constraint το admin_username στο username από τον πίνακα user, ώστε να προέρχεται από εκείνο τον πίνακα.*

)engine=InnoDB;

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

```
CREATE TABLE manager
```

```
(
```

```
manager_username VARCHAR(30) NOT NULL,
```

```
company_exp TINYINT(4) NOT NULL,
```

manager_id INT(4) NOT NULL AUTO_INCREMENT, (extra)Θέτουμε ότι ο κάθε μάνατζερ έχει ένα μοναδικό ID το οποίο με κάθε νέα εισαγωγή καινούργιου μάνατζερ θα αυξάνετε αυτόματα, από το προηγούμενο ID κατά 1.

```
manager_company CHAR(9) NOT NULL,
```

PRIMARY KEY (manager_username), Θέτουμε ως πρωτεύον κλειδί το manager_username όπως μας έχει δωθεί.

UNIQUE KEY(manager_id), Θέτουμε το manager_id μοναδικό κλειδί καθώς κάθε μάνατζερ έχει μοναδικό ID.

CONSTRAINT MNGER_USRNNM FOREIGN KEY (manager_username) REFERENCES `user`(username) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το manager_username στο username από τον πίνακα user, ώστε κάθε ψευδώνυμο των μάνατζερ να το δείχνει από τον πίνακα user.

CONSTRAINT MNGER_COMPANY FOREIGN KEY (manager_company) REFERENCES company(AFM) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το manager_company στο AFM από τον πίνακα company, ώστε να βλέπουμε σε ποια εταιρεία ανήκει ο κάθε μάνατζερ.

```
)engine=InnoDB;
```

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

```
CREATE TABLE request_evaluation
```

```
(
```

```
empl_usrnm VARCHAR(30) NOT NULL,
```

```
jobid INT(4) NOT NULL,
```

CONSTRAINT JOB_ID FOREIGN KEY(jobid) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE,

Κάνουμε Constraint το jobid στο job_id από τον πίνακα job, ώστε να υπάρχει σε κάθε νέα αίτηση αξιολόγησης το μοναδικό ID κάθε δουλειάς.

CONSTRAINT EMPL_EVAL FOREIGN KEY (empl_usrnm) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE

Κάνουμε Constraint το empl_usrnm στο employee_username από τον πίνακα employee, ώστε κάθε αίτηση να περιέχει το ψευδώνυμο κάθε υπαλλήλου που κάνει αίτηση.

```
)engine=InnoDB;
```

Βάζουμε NOT NULL, καθώς αυτά τα στοιχεία θεωρούμε ότι είναι απαιτούμενα.

```
CREATE TABLE process_1 EXTRA πίνακας για την πρώτη φάση αξιολόγησης δηλαδή την συνέντευξη κάθε υποψηφίου.
```

```
(
```

```
em_usrnm VARCHAR(30), Εδώ έχουμε το username κάθε υποψηφίου(employee) που έχει κάνει αίτηση.
```

id_empl INT(4) , Το ID κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης .

id_eva INT(4) , Το ID κάθε evaluator(αξιολογητής) που έχει αναλάβει την αξιολόγηση.

jb_id INT(4) ,Το ID κάθε δουλειάς στην οποία έχουν κάνει αίτηση αξιολόγησης, οι υποψήφιοι-υπάλληλοι.

grade_1 ENUM('0','1','2','3','4'), Ο βαθμός της πρώτης φάσης(συνέντευξη) που μπορεί να είναι από 0-4.

comments_1 TEXT, Σχόλια του αξιολογητή.

KEY (grade_1), Θέτουμε ως κλειδί το βαθμό της πρώτης φάσης αξιολόγησης, grade_1.

CONSTRAINT PRO_USRNM FOREIGN KEY(em_usrnm) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το em_usrnm στο employee_username από τον πίνακα employee, ώστε η κάθε συνέντευξη να έχει το username του υπαλλήλου/υποψηφίου.

CONSTRAINT PROC_JBID FOREIGN KEY(jb_id) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE, , Κάνουμε Constraint το jb_id στο job_id από τον πίνακα job, ώστε να έχει το μοναδικό ID της κάθε δουλειάς στην οποία έγινε η πρώτη φάση της αξιολόγησης.

CONSTRAINT ID_EMPLO FOREIGN KEY (id_empl) REFERENCES employee(empl_id) ON DELETE CASCADE ON UPDATE CASCADE , Κάνουμε Constraint το id_empl στο empl_id από τον πίνακα employee, ώστε να έχει το μοναδικό ID του κάθε υποψηφίου/υπαλλήλου για την πρώτη φάση της αξιολόγησης.

)engine=InnoDB;

ΔΕΝ ΒΑΖΟΥΜΕ NOT NULL ΚΑΘΩΣ ΜΠΟΡΕΙ ΝΑ ΕΙΝΑΙ ΚΕΝΑ ΤΑ ΣΤΟΙΧΕΙΑ .

CREATE TABLE process_2 EXTRA πίνακας για την δεύτερη φάση της αξιολόγησης δηλαδή την αξιολόγηση με βάση του report του manager για τον κάθε υποψήφιο/υπάλληλο.

(

id_manager INT(4) , Το ID κάθε μάνατζερ που ανήκει στην εταιρεία που έχει γίνει η αίτηση.

id_eval INT(4) , Το ID κάθε evaluator(αξιολογητής) που έχει αναλάβει την αξιολόγηση .

usrnm_empl VARCHAR(30) , Εδώ έχουμε το username κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης.

id_employee INT(4) , Το ID κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης.

id_job INT(4) , Το ID κάθε δουλειάς στην οποία έχουν κάνει αίτηση αξιολόγησης, οι υποψήφιοι-υπάλληλοι.

anafora TEXT, Η αναφορά του manager εταιρείας, όπου εργάζεται ο κάθε υπάλληλος.

grade_2 ENUM('0','1','2','3','4'), Ο βαθμός της δεύτερης φάσης(συνέντευξη) που μπορεί να είναι από 0-4

KEY (grade_2), Θέτουμε ως κλειδί το βαθμό της δεύτερης φάσης αξιολόγησης, grade_2

CONSTRAINT ID_MANA FOREIGN KEY (id_manager) REFERENCES manager(manager_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το id_manager στο manager_id από τον πίνακα manager, ώστε να έχει το μοναδικό ID του κάθε manager για την δεύτερη φάση της αξιολόγησης.

CONSTRAINT ID_EVALU FOREIGN KEY(id_eval) REFERENCES evaluator(evaluator_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το id_eval στο evaluator_id από τον πίνακα evaluator, ώστε να έχει το μοναδικό ID του κάθε αξιολογητή για την δεύτερη φάση της αξιολόγησης.

CONSTRAINT PROC_USRNM FOREIGN KEY(usrnm_empl) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το usrnm_empl στο employee_username από τον πίνακα employee, ώστε η κάθε δεύτερη φάση αξιολόγησης να έχει το username του υπαλλήλου/υποψηφίου.

CONSTRAINT ID_EMPLOYEE FOREIGN KEY(id_employee) REFERENCES employee(empl_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το id_employee στο empl_id από τον πίνακα employee, ώστε να έχει το μοναδικό ID του κάθε υποψήφιου/υπαλλήλου για την δεύτερη φάση της αξιολόγησης.

CONSTRAINT PRO_JOBID FOREIGN KEY(id_job) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE
Κάνουμε Constraint το id_job στο job_id από τον πίνακα job, ώστε να έχει το μοναδικό ID της κάθε δουλειάς, που έγινε η δεύτερη φάση της αξιολόγησης.

)engine=InnoDB;

ΔΕΝ ΒΑΖΟΥΜΕ NOT NULL ΚΑΘΩΣ ΜΠΟΡΕΙ ΝΑ ΕΙΝΑΙ ΚΕΝΑ ΤΑ ΣΤΟΙΧΕΙΑ .

CREATE TABLE process_3 EXTRA πίνακας για την τρίτη φάση της αξιολόγησης Την αξιολόγηση των πτυχίων, συστατικών επιστολών, τον αριθμό των project που έχει υλοποιήσει και τις διακρίσεις που έχει πάρει.

(

usrn_em VARCHAR(30) , Εδώ έχουμε το username κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης.

id_emplo INT(4) , Το ID κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης.

id_evalu INT(4) , Το ID κάθε evaluator(αξιολογητής) που έχει αναλάβει την αξιολόγηση .

id_jb INT(4) , Το ID κάθε δουλειάς στην οποία έχουν κάνει αίτηση αξιολόγησης, οι υποψήφιοι-υπάλληλοι.

grade_3 ENUM('0','1','2'), Ο βαθμός της δεύτερης φάσης(συνέντευξη) που μπορεί να είναι από 0-2

KEY (grade_3), Θέτουμε πρωτεύον κλειδί το βαθμό της δεύτερης φάσης αξιολόγησης, grade_2

CONSTRAINT PROCE_USRNM FOREIGN KEY(usrn_em) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το usrn_em στο employee_username από τον πίνακα employee, ώστε η κάθε τρίτη φάση αξιολόγησης να έχει το username του υπαλλήλου/υποψηφίου.

CONSTRAINT ID_EVALUA FOREIGN KEY(id_evalu) REFERENCES evaluator(evaluator_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το id_evalu στο evaluator_id από τον πίνακα evaluator, ώστε να έχει το μοναδικό ID του κάθε αξιολογητή για την τρίτη φάση της αξιολόγησης.

CONSTRAINT ID_EMPLOY FOREIGN KEY (id_emplo) REFERENCES employee(empl_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το id_emplo στο empl_id από τον πίνακα employee, ώστε να έχει το μοναδικό ID του κάθε υποψήφιου/υπαλλήλου για την τρίτη φάση της αξιολόγησης.

CONSTRAINT PROCE_JOBID FOREIGN KEY(id_jb) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE
Κάνουμε Constraint το id_jb στο job_id από τον πίνακα job, ώστε να έχει το μοναδικό ID της κάθε δουλειάς, που έγινε η τρίτη φάση της αξιολόγησης.

)engine=InnoDB;

ΔΕΝ ΒΑΖΟΥΜΕ NOT NULL ΚΑΘΩΣ ΜΠΟΡΕΙ ΝΑ ΕΙΝΑΙ ΚΕΝΑ ΤΑ ΣΤΟΙΧΕΙΑ .

CREATE TABLE evaluation_result

(

Evid INT(4) , Το ID κάθε evaluator(αξιολογητής) που έχει αναλάβει την αξιολόγηση .

empl_username VARCHAR(30) ,extra Έχουμε το username κάθε υποψηφίου(employee) που έχει κάνει αίτηση αξιολόγησης

idjb INT(4) , Το ID κάθε δουλειάς στην οποία έχουν κάνει αίτηση αξιολόγησης, οι υποψήφιοι-υπάλληλοι.

rgrade_1 ENUM('0','1','2','3','4'), extra σκεφτήκαμε για να βλέπει ο κάθε αξιολογητής την τελική θα ήταν καλό να έχουμε μαζέψει όλους τους βαθμούς από τις προηγούμενες φάσης αξιολόγησης κι έπειτα με την βοήθεια του procedure να υπολογίζονται όλα μαζί στο τελικό βαθμό garde. Δηλαδή εδώ έχουμε το βαθμό της φάσης 1.

rgrade_2 ENUM('0','1','2','3','4'), extra σκεφτήκαμε για να βλέπει ο κάθε αξιολογητής την τελική θα ήταν καλό να έχουμε μαζέψει όλους τους βαθμούς από τις προηγούμενες φάσης αξιολόγησης κι έπειτα με την βοήθεια του procedure να υπολογίζονται όλα μαζί στο τελικό βαθμό garde. Δηλαδή εδώ έχουμε το βαθμό της φάσης 2.

rgrade_3 ENUM('0','1','2'), extra σκεφτήκαμε για να βλέπει ο κάθε αξιολογητής την τελική θα ήταν καλό να έχουμε μαζέψει όλους τους βαθμούς από τις προηγούμενες φάσης αξιολόγησης κι έπειτα με την βοήθεια του procedure να υπολογίζονται όλα μαζί στο τελικό βαθμό garde. Δηλαδή εδώ έχουμε το βαθμό της φάσης 3.

grade INT(4) , Ο τελικός βαθμός αξιολόγησης με maximum 10.

comments VARCHAR(255) , Τα σχόλια του τελικά αξιολογητή για τον υποψήφιο/υπάλληλο.

CONSTRAINT EVA_ID FOREIGN KEY (Evid) REFERENCES evaluator(evaluator_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το Evid στο evaluator_id από τον πίνακα evaluator, ώστε να έχει το μοναδικό ID του κάθε αξιολογητή για τα αποτελέσματα της αξιολόγησης.

CONSTRAINT EMPL_EVALR FOREIGN KEY (empl_username) REFERENCES employee(employee_username) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το empl_username στο employee_username από τον πίνακα employee, ώστε το κάθε αποτέλεσμα αξιολόγησης να έχει το username του υπαλλήλου/υποψηφίου.

CONSTRAINT EVAR_JOBID FOREIGN KEY(idjb) REFERENCES job(job_id) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το idjb στο job_id από τον πίνακα job, ώστε να έχει το μοναδικό ID της κάθε δουλειάς, για τα αποτελέσματα της αξιολόγησης.

CONSTRAINT RGRADE_1 FOREIGN KEY (rgrade_1) REFERENCES process_1(grade_1) ON DELETE CASCADE ON UPDATE CASCADE, , Κάνουμε Constraint το rgrade_1 στο grade_1 από τον πίνακα process_1, ώστε να παίρνει τον βαθμό της πρώτης φάσης αξιολόγησης, για να τον εμφανίζει στο τελικό πίνακα μαζί με όλους τους άλλους.

CONSTRAINT RGRADE_2 FOREIGN KEY (rgrade_2) REFERENCES process_2(grade_2) ON DELETE CASCADE ON UPDATE CASCADE, Κάνουμε Constraint το rgrade_2 στο grade_2 από τον πίνακα process_2, ώστε να παίρνει τον βαθμό της δεύτερης φάσης αξιολόγησης, για να τον εμφανίζει στο τελικό πίνακα μαζί με όλους τους άλλους.

CONSTRAINT RGRADE_3 FOREIGN KEY (rgrade_3) REFERENCES process_3(grade_3) ON DELETE CASCADE ON UPDATE CASCADE Κάνουμε Constraint το rgrade_3 στο grade_3 από τον πίνακα process_3, ώστε να παίρνει τον βαθμό της τρίτης φάσης αξιολόγησης, για να τον εμφανίζει στο τελικό πίνακα μαζί με όλους τους άλλους.

)engine=InnoDB;

ΔΕΝ ΒΑΖΟΥΜΕ NOT NULL ΚΑΘΩΣ ΜΠΟΡΕΙ ΝΑ ΕΙΝΑΙ ΚΕΝΑ ΤΑ ΣΤΟΙΧΕΙΑ.

CREATE TABLE log EXTRA πίνακας για τις ενέργειες του κάθε user(χρήστη) μέσα στο σύστημα στους πίνακες job, employee και request-evaluation.

(

log_username VARCHAR(30) , Το username του χρήστη που κάνει κάποια ενέργεια.

table_edited ENUM('job','employee','request_evaluation'), Ποιος από αυτούς τους τρεις πίνακες ενημερώθηκε.

date_of_action DATETIME , Ημερομηνία και ώρα που άλλαξε κάτι σε έναν από τους πίνακες.

resault_of_action ENUM('INSERT','UPDATE','DELETE'), Τι από τα τρία αυτά έγινε εισαγωγή, ενημέρωση, διαγραφή.

result_succses ENUM ('SUCCSED','FAILED') Αν αυτό που άλλαξε έγινε με επιτυχία ή απορρίφθηκε .

)engine=InnoDB;

ΔΕΝ ΒΑΖΟΥΜΕ NOT NULL ΚΑΘΩΣ ΜΠΟΡΕΙ ΝΑ ΕΙΝΑΙ ΚΕΝΑ ΤΑ ΣΤΟΙΧΕΙΑ.

```
CREATE TABLE evaluator_stats EXTRA πίνακας για ώστε να μπορούμε να κρατάμε το μέσο όρο των evaluators.
(
sevaluatorid INT(4), Μεταβλητή για το ID του evaluator.
sevaluator_username VARCHAR(30), Μεταβλητή για το username του evaluator.
avgrade FLOAT, Μεταβλητή για το βαθμό του evaluator.
PRIMARY KEY (sevaluator_username), Θέτουμε ως πρωτεύον κλειδί το sevaluator_username.
CONSTRAINT SUSERNAME FOREIGN KEY (sevaluator_username) REFERENCES evaluator(evaluator_username) ON DELETE
CASCADE ON UPDATE CASCADE, Κάνουμε Constraint to sevaluator_username στο evaluator_username από το πίνακα
evaluator για να πάρουμε το username του εξιολογητη.
CONSTRAINT sevid FOREIGN KEY (sevaluatorid) REFERENCES evaluator(evaluator_id) ON DELETE CASCADE ON UPDATE
CASCADE Κάνουμε Constraint to sevaluatorid στο evaluator_id από το πίνακα evaluator, για να πάρουμε το ID του
εξιολογητή.
)engine=InnoDB;
```

INSERTS:

➤ ΔΙΕΥΚΡΙΝΙΣΕΙΣ:

- 1) Υποθέσαμε, ότι το πτυχίο κάθε υπαλλήλου θα βρίσκεται σε ένα αρχείο pdf, το οποίο θα το στέλνουν υποθετικά αυτοί(υποθετικό scanned πτυχίο).
- 2) Υποθέσαμε, ότι όσοι υπάλληλοι έχουν projects αναρτημένα στο διαδίκτυο, έχουν έναν υποθετικό σύνδεσμο, στον οποίο ανοίγει το project(πάντα υποθετικά).
- 3) Υποθέσαμε, ότι κάθε ID είναι μοναδικό και για κάθε είδος χρήστη θα ξεκινά διαφορετικά, δηλαδή για κάθε employee ξεκινάει με 15, για κάθε manager με 10, για κάθε evaluator με 23. Όπως και για κάθε εταιρεία το ΑΦΜ τους θα ξεκινάει από 101 και για κάθε δουλειά το ID θα ξεκινάει με 20.

Συνολικά έχουμε προσθέσει 40 users, από τους οποίους έχουμε θέσει ως: 1 admin, 3 manager, 6 evaluator, 30 employees. Σκεφτήκαμε, ότι τρεις εταιρείες είναι αρκετές ώστε να καλύψουν αρκετούς ρόλους για τους users. Αποφασίσαμε ότι 10 employees είναι αρκετοί για καλύψουν κάθε εταιρεία. Έχουμε θέσει 8 αιτήσεις για αξιολόγηση, από τις οποίες μόνο δύο έχουν φτάσει στα αποτελέσματα αξιολόγησης(για την βοήθεια να βλέπουμε τι γίνεται στα procedures). Υποθέτουμε, ότι τις άλλες θα τις αξιολογούμε από το GUI, συν έξτρα που θα φτιάξουμε μέσα σε αυτό. Γενικώς, σε κάθε employee(υπάλληλο) βάλαμε αρκετά δεδομένα, ώστε να καλύψουν τις απαιτήσεις κάθε εταιρείας και κάθε αξιολογητή. Να μπορούν να υπάρχουν υπάλληλοι σε 'κατώτερη' δουλειά, ώστε να πάρουν προαγωγή, αλλά υπάρχουν και μερικοί οι οποίοι είναι σε ανώτερη θέση μέσα κάθε στην εταιρεία, ώστε να μην χρειάζεται να κάνουν όλοι οι υπάλληλοι της κάθε εταιρείας αίτηση για αξιολόγηση, και να βλέπουμε την ιεραρχία.

```
/*USER*/
```

```
INSERT INTO `user`(username,`password`,`name`,surname,reg_date,email) VALUES
("katerinaki","spitimou","Katerina","Nikolaou",'2020-10-05 01:34:20',"nikakaterina@gmail.com"),
("mhstakil","nystazw9","Dhnhtra","Lazarou",'2019-05-13 15:45:34',"dhmhl@yahoo.gr"),
("kwstakis93","peinawpol","Kwnstantinos","Karakwstas",'2018-03-16 16:32:26',"kwsk93@gmail.com"),
("nikolakis","80b12nikos","Nikolas","Pappas",'2016-11-17 17:27:15',"nikospap@yahoo.com"),
```

("agathwn","thelwYpno9","Agatha","Skylra",'2019-10-01 02:35:46',"agathwnsk@gmail.com"),

("yianstamel","sky!96A","lwannhs","Stamelos",'2018-01-26 16:48:36',"yianstamel@yahoo.gr"),

("maroutsos","maria97","Maria","Skylra",'2017-11-06 18:27:18',"mariask@gmail.com"),

("manoumanou99","kafes99","Emmanouela","Pappa",'2019-05-12 09:25:36',"manou99@yahoo.gr"),

("tonyzaf","anTo9","Antwnhs","Zafeirelhs",'2016-06-15 12:36:25',"tonyzaf@gmail.com"),

("savvas94","paizwlol","Savvas","Kalamis",'2016-04-03 15:18:41',"savvka94@yahoo.gr"),

("markzymn","brm87!","Markellos","Zymnhs",'2019-07-21 19:38:12',"markzymn@gmail.com"),

("siliagk","ch!lia9","Basilikh Leukothea","Gkeka",'2018-10-25 10:23:41',"siliagk@gmail.com"),

("mixaltop","tOm!X76","Mixahl","Toptshs",'2013-09-01 14:32:12',"mixatop@gmail.com"),

("yiannisxA","m8ctP28!","lwannhs","Xaztimixalhs",'2016-05-26 22:31:20',"yiannisxa@yahoo.gr"),

("gianStam","x4L45!s","lwannhs","Stamoulhs",'2020-12-28 05:32:12',"gianstam@gmail.com"),

("chrisboul","chris99A","Christos","Boulafenthhs",'2015-06-28 20:04:26',"chrisboul99@gmail.com"),

("anastasila","anast987","Anastasia","Laskaridh",'2014-09-19 18:36:25',"anastasialas@yahoo.gr"),

("andrpi","1245!","Andreas","Pikas",'2013-04-25 11:07:46',"andreaspi@gmail.com"),

("elengeo","sdfllif3","Eleni","Georgiadou",'2017-06-30 17:15:06',"elengeo@gmail.com"),

("eyangeo","eya19!A","Eyanthia","Georgiadou",'2018-10-23 16:28:52',"eyxageo@gmail.com"),

("kwstas96","KwstasBro","Kostantinos","Nikolaou",'2018-05-01 12:46:45',"kwstasnik@gmail.com"),

("dimitrispap","Dimitris85","Dimitris","Papadopoulos",'2018-05-30 07:25:15',"dimitris.papadopoulos@gmail.com"),

("marakioik","Maria94","Maria","Oikonomou",'2018-07-22 09:25:16',"maria.oikonomou@gmail.com"),

("nikosgat","Nikos74","Nikos","Gatos",'2019-06-14 15:21:36',"nikos.gatos@gmail.com"),

("elenisar","Eleni89","Eleni","Saranti",'2020-01-23 12:23:15',"eleni.saranti@gmail.com"),

("mariadok","Maria95","Maria","Dokouli",'2019-06-11 10:31:29',"maria.dok@gmail.com"),

("elenipapaxri","Eleni84","Eleni","Papaxristou",'2014-11-21 13:25:01',"elenipapaxristou@yahoo.gr"),

("georgioik","Giorgos96","Georgios","Oikonomou",'2020-04-22 10:00:01',"georgioikonomou@gmail.com"),

("basilikigeo","Vasiliki91","Vasiliki","Georgiou",'2017-11-22 17:025:11',"vasiliki.georgiou@yahoo.gr"),

("dimitriskous","Dimitris88","Dimitrios","Koustos",'2016-01-30 12:27:21',"dimitris.koustos@gmail.com"),

("kwstasthom","Kwstas92","Kostantinos","Thomaidis",'2018-04-14 18:25:36',"kostantinos.thommaidis@gmail.com"),

("giannisadam","Giannis79","Ioannis","Adamopoulos",'2013-11-06 22:45:01',"ioannis.adamopoulos@gmail.com"),

("athinamar","Athina92","Athina","Martaki",'2016-06-04 11:15:09',"athinamartaki@yahoo.gr"),

("nikosthan","Nikos87","Nikolaos","Thanasouras",'2018-07-14 13:42:16',"nikos.thanasouras@gmail.com"),

("georgiasaim","Georgia96","Georgia","Saimi",'2017-10-06 16:22:31',"georgia.saimi@gmail.com"),

("sofialap","Sofia95","Sofia","Lapari",'2019-04-21 17:45:32',"sofia.lapari@gmail.com"),

("anastasiarep","Anastasia87","Anastasia","Repopoulou",'2015-06-21 21:15:41',"anastasia.reppoulou@gmail.com"),

("evaggeliavotsi","Evaggelia94","Evaggelia","Votsi",'2016-10-12 09:36:25',"evaggelia.votsi@gmail.com"),

```
("ioannasake","Ioanna89","Ioanna","Sakelaridi",'2017-02-03 08:28:12',"ioanna.sakelaridi@gmail.com"),
("xristospap","Xristos97","Xristos","Papagiannis",'2020-04-02 11:32:40',"xristos.papagiannis@yahoo.gr");
SELECT * FROM `user`;
```

```
/*ADMIN*/
```

```
INSERT INTO administrator(admin_username) VALUES ("katerinaki");
SELECT * FROM administrator;
```

```
/*COMPANY*/
```

```
INSERT INTO company(AFM,DOY,phone,street,street_number,city,country,company_name) VALUES
('101235645','Pagkrati',2106589454,"Neoptolaimou",12,"Athens","Greece","Logisthrio Papakwsta"),
('101456789','Xolargou',2104638622,'Pentelis',74,'Athens','Greece','Finacial Advice'),
('101879123','A Thessalonikis',2310542106,'Ermou',33,'Thessaloniki','Greece','Financials A.E.');
```

```
SELECT * FROM company;
```

```
/*MANAGER*/
```

```
INSERT INTO manager(manager_username,company_exp,manager_id,manager_company) VALUES
("dimitrispap",6,1000,'101235645'),
("markzymn",7,1001,'101456789'),
("agathwn",5,1002,'101879123');
```

```
SELECT * FROM manager;
```

```
/*EVALUATOR*/
```

```
INSERT INTO evaluator(evaluator_id,evaluator_username,evaluator_company,years_of_exp) VALUES
(2359,"nikolakis",101235645,4),
(2360,"manoumanou99",101879123,2),
(2361,"yianstamel",101235645,6),
(2362,"savvas94",101879123,5),
(2301,"kwstas96",101456789,4),
(2302,"marakioik",101456789,2);
```

```
SELECT * FROM evaluator;
```

```
/*EMPLOYEE*/
```

INSERT INTO

employee(employee_username,empl_id,cmp_afm,expe_years,bio,sistatikes,certificates,awards,manager_report) VALUES

("tonyzaf",1548,'101235645',4,"Antony Zafeiris Zafeirelis has an Accounting deploma, he is a an experienced accouantant (with over 4 years of experience) ","Dear employer, Antonis Zafeiris Zafeirelis has worked in our company for over 4 years as an Accountant he is a very capable and cooperative person. With respect Agatha Skyrla. ","certif_antonyzaf.pdf",""),

("mhtsakil",1549,'101879123',2,"Dhnhtra Lazarou is an Economics degree holder,she also mastered in Accounting ,and worked as a manager at a Financials company for 2 years.","Dear employer, Dimitra Lazarou is a manager in our company for over 2 years with great experience in her sector. ","certif_dhnhtralaz.pdf",""),

("siliagk",1550,'101879123',3,"Vasiliki Leukothea Gkeka has an Accounting deploma also holds an Economics deploma as well as she has mastered the Financials and Economics.She has experience in the Accounting as she worked as an Assistant Manager for over 3 years.","Dear employer, Vasiliki Leykothea Gkeka has worked in our company as an Assisatnt Manager for over 3 years.She is very capable and very smart her skills can contribute in your company.With respect ","certif_vasilikigk.pdf","Glen McLaughlin Prize for Research in Accounting Ethics(2018)",""),

("maroutsos",1551,'101235645',3,"Maria Skyrla has an Economics deploma,she has experience in Economics department for over 3 years. She specializes as an Financial Advisor with experience in that sector of 3 years.","Dear employer, Maria Skyrla has worked in our company for over 3 years , with great cooperation and good skills as an Financial Advisor. ","cert_mar_sk.pdf",""),

("mixaltop",1552,'101879123',8,"Mixahl Toptsis has a PHD in Accounting, he has a master Economics as well as he has experience over 8 years in the sector of Accounting and Econmics. ","Dear employer, Mixahl Toptsis is a great has worked as an Assistant Manager in our company for over 5 years and as a Accountant in another company for over 3 years. His skills are efficiantly in our company. ","certif_mt.pdf","Journal of Management Accounting Research Best Paper Award(2017) & Outstanding Behavioral Accounting Researcher Award(2019)",""),

("yiannisxA",1553,'101879123',4,"Iwannhs Xatzhmixalhs is a High School graduate . He worked as an Assistant accountant for over 4 years. ","","certi_iwnxa.pdf",""),

("elenisar",1554,'101235645',1,"Elenh Saranti is a graduate of High School of Nicosia, She has experience in Economics Department as an Assistant Accountant for 1 year. ","","certific_elensar",""),

("eyangeo",1055,'101235645',2,"Eyanthia Georgiadou has an Accounting and Bussiness Master. She worked as an assistant accountant for over 2 years.","Dear employer, Eyanthia Georgiadou is an exchellent Assistant Accountant in our company for over 2 years.Her work has been beneficial to our company. ","certifi_eyangeo.pdf",""),

("chrisboul",1556,'101235645',9,"Christos Boulafentis has PHD in Ecomonics and Statistics, he has won awards for his contribution to the University of Patras which he has graduated with honors.He has worked as an Assistant Manager in two different companies and he has experience over 8 years.","Dear employer, Christos Boulfentis has worked in our company for over 5 years as an Assitant manager and before that for 3 years as an Accountant.He is very tallented and hard-working person. ","chris_boul_certif.pdf","Accounting Research Best Paper Award(2016) & Outsanding Educator Award(2014)",""),

("andrpi",1557,'101235645',7,"Andreas Pikas holds a deploma of Accounting he is very experiened in his sector as well as he has experience of 7 years as an Accountant. ","Dear employer, Andreas Pikas is a very punctual and hard-working Accountant in our company for over 7 years. ","certif_andpik",""),

("kwstakis93",1558,'101235645',2,"Kwnstantinos Karakwstas is a graduate of the Economics department, he has experience as an Assistant Accountant for over 2 years. ","Dear employer Kwnstantinos Karakwstas is an excellent Assistant Accounter in our company for over 2 years.He is a deligent and respectfull person. ","karkwst_certific.pdf",""),

("elengeo",1559,'101235645',3,"Elenh Georgiadou has a master in Accounting as well as she has been many times honored for her skills though her young age she has managed to contribute in the Accounting sector with her skills with over 3 yers of expereince as an Accounter. ","Dear employer, Eleni Georgiadou is a deligdent hard-working person.Though her little experience she has contributed to our company and to Accounting sector. ","","Accounting Best Paper Award(2018)",""),

("anastasila",1560,'101235645',6,"Anastasia Laskaridi is a former manager and a graduate of Economics University of Athens, with over 6 years of experience. ","Dear employer, Anastasia Laskaridi is a former manager in our company. She is a very hard-working and willing person. ","anastsialask_certificate.pdf",""),

("nikosgat",1561,'101235645',1,"Nikolaos Gatos has recently graduated University of Aegean in the departments of Maths and Statistics and worked as an Assistans Accountant for 1 year.", "Dear employer, Nikolaos Gatos is a very skillfull and willing person working as an Accountant in our company.", "nikgat_certif.pdf", "", ""),

("gianStam",1562,'101235645',1,"Iwannhs Stamoulhs is a High School Graduate , he has expereince of 1 year as an Assistant Accountant.", "", "certifi_giannstam.pdf", "", ""),

("mariadok",1501,101456789,1,"Maria Dokouli is a High School Graduate.She has experience in Assistant Accounting for 1 year.", "", "crtif_mariadok.pdf", "", ""),

("elenipapaxri",1502,101456789,2,"Eleni Papaxristou has a Financials Deploma.Shw has 2 years of a experience as an Accountant.", "Dear employer, Maria Papaxristou is a very capable employee.She has two years of experience as an Accountant in our company.She is a willing and hard-working person.", "certifi_elenpapa.pdf", "", ""),

("georgeoik",1503,101456789,2,"Gergios Oikonomou is a graduate of University of Pereus.Hw has a Financials Deploma and experience as an Accountant for over 6 years", "Dear employer, Georgios Oikonomou is a very hard-working ajmd willing Accountant .He has worked in our company for over three years and in other companies for over 4 years.Very good skills and great personality.", "certif_geooik.pdf", "Accounting Best Paper Award(2015)", ""),

("basilikigeo",1504,101456789,4,"Vasiliki Georgiou has a Financials Deploma.Shw has four years of a experience as an Accountant.", "", "vasilkgeor.pdf", "", 'Exchellent in general'),

("dimitriskous",1505,101456789,2,"Dimitris Koustos is a High School graduate, he has experience as an Assistant for 2 years.", "", "crt_vasilkou.pdf", "", 'Good employee'),

("kwstasthom",1506,101456789,1,"Kostantinos Thomaidis is a graduate from University of Patras in Finanacials.He has experience as an Accountant for 1 year.", "", "certifi.pdf", "", 'Average employee not the best.'),

("giannisadam",1507,101456789,3,"Ioannis Adamopoulos is a graduate from University of Patras in Economics and Financials.He has worked as an Accountant for over 3 years.", "Dear employer, Ioannis Adamopoulos has worked in our comooany for over 3 years as an Accountant.He has fulfilled his duties with honors.He is very hard-working and willing", "adm_certi.pdf", "", 'Average/not intersting'),

("athinamar",1508,101456789,4,"Athina Martaki has a Financials Deploma from Universty of Patras.She has experience as an Assistant Accountant for over 4 years.", "Dear employer , Athina Martaki n exchellent Assistant Accountant in our company for over 4 years.Her work has been beneficial to our company ", "athnmar_certificate.pdf", "", ""),

("nikosthan",1509,101456789,2,"Nikolaos Thanasouras has a degree in Economics.He has experience of 2 years as an Accountant.", "", "cerifi_nikthan.pdf", "", ""),

("georgiasaim",1510,101456789,2,"Georgia Saimi is a graduate from ASOE she has a degree in Financials and she has worked as an assistant accountant for over 2 years.", "Dear employer, Gerorgia Saimi is a very good and hard-working Assistant Accountant.She has worked in our company for over 2 years and her work is very beneficial to our company.", "cerifi_georgs.pdf", "", 'Average in total'),

("sofialap",1511,101879123,3,"Sofia Lapari has a degree in Financials.She has experien ce as an Accountant for over three years.", "", "certif_soflap.pdf", "", ""),

("anastasiarep",1512,101879123,1,"Anastasia Repopoulou is a graduate from ASOE she has a degree in Financials and she has worked as an assistant accountant for over 1 year", "", "certif_anast.pdf", "", ""),

("evaggeliavotsi",1513,101879123,4,"Evaggelia Votsi is an Accountant with experience for over 4 years.She has a degree from ASOE in Fianancials.", "Dear employer, Evaggelia Votsi is a very willing and hard-working person in our company.She has worked for us for over 4 years.", "certif_evagvots.pdf", "", ""),

("ioannasake",1514,101879123,5,"Ioanna Sakelaridi is an Assistant Manage for over 4 years.She has worked as an Accountant for 1 year and then she was pormoted to Assistant Manager.She has a degree in Finncials.", "Dear employer. Ioanna Sakelaridi has worked in our comooany for 5 years she has been very willing and hard-working form the start.", "certificate_ionsak.pdf", "", 'Exchellent and good skills'),

```
("xristospap",1515,101879123,2,"Xristos Papagiannis has a Master in Financials from ASOE.He has experience for 2 years as an Accountant","","certiif_xripap.pdf","","");
```

```
SELECT * FROM employee;
```

```
/*DEGREE*/
```

```
INSERT INTO degree(titlos,idryma,bathmida) VALUES
```

```
("Accounting and Bussiness","University of Athens",'UNIV'),
```

```
("Accounting and Financials","University of Nicosia",'MASTER'),
```

```
("Accounting and Bussiness","University of Cambridge",'MASTER'),
```

```
("Economics and Statistics","University of Patras",'UNIV'),
```

```
("Economics and Bussiness","University of Thessaloniki",'PHD'),
```

```
("Vocational High School Deploma","2nd Vocational High School of Athens",'LYKEIO'),
```

```
("High School Diploma","1st High School of Nicosia",'LYKEIO'),
```

```
("Accounting and Bussiness","University of Thessaloniki",'MASTER'),
```

```
("Economics and Statistcs","University of Patras",'PHD'),
```

```
("Accounting and Financials","University of Athens",'UNIV'),
```

```
("Economics and Accounting","University of Nicosia",'UNIV'),
```

```
("Accounting and Financials","University of Oxford",'MASTER'),
```

```
("Economics and Statistics","University of Athens",'UNIV'),
```

```
("Mathematics and Statistics","University of Aeagean",'UNIV'),
```

```
("High School Diploma","High School of Mesolonghi",'LYKEIO'),
```

```
("Highschool Degree","3o Lykeio Kifisias",'LYKEIO'),
```

```
("Highschool Degree","3o Lykeio Patras",'LYKEIO'),
```

```
("Financial Sience","ASOE",'UNIV'),
```

```
("Financial Sience","University of Peireus",'UNIV'),
```

```
("Financial Sience","University of Patras",'UNIV'),
```

```
("Economics and Team Managment","ASOE",'MASTER');
```

```
SELECT * FROM degree;
```

```
/*HAS_DEGREE*/
```

```
INSERT INTO has_degree(degr_title,degr_idryma,empl_usurname,etos,grade) VALUES
```

```
("Accounting and Bussiness","University of Athens","tonyzaf",2016,'8.5'),
```

```
("Accounting and Financials","University of Nicosia","mhtsakil",2017,'7.5'),
```

```
("Accounting and Bussiness","University of Cambridge","siliagk",2016,'9.5'),
```

```
("Economics and Statistics","University of Patras","maroutsos",2015,'7.5'),
```


("Economics and Bussiness","University of Thessaloniki","mixaltop",2010,'9'),
 ("Vocational High School Deploma" ,"2nd Vocational High School of Athens","yiannisxA",2013,'10'),
 ("High School Diploma","1st High School of Nicosia","elenisar",2019,'10'),
 ("Accounting and Bussiness","University of Thessaloniki","eyangeo",2017,'8.5'),
 ("Economics and Statistcs","University of Patras","chrisboul",2012,'10'),
 ("Accounting and Financials","University of Athens","andrpi",2010,'8'),
 ("Economics and Accounting","University of Nicosia","kwstakis93",2017,'8.5'),
 ("Accounting and Financials","University of Oxford","elengeo",2015,'9'),
 ("Economics and Statistics","University of Athens","anastasila",2012,'8.5'),
 ("Mathematics and Statistics","University of Aeagean","nikosgat",2019,'6.5'),
 ("High School Diploma","High School of Mesolonghi","gianStam",2019,'10'),
 ("Highschool Degree","3o Lykeio Kifisias","mariadok",2013,'15.2'),
 ("Financial Sience","University of Peireus","elenipapaxri",2006,'7.2'),
 ("Financial Sience","University of Peireus","georgeoik",2018,'6'),
 ("Financial Sience","University of Peireus","basilikigeo",2013,'5.4'),
 ("Highschool Degree","3o Lykeio Patras","dimitriskous",2005,'16.0'),
 ("Financial Sience","University of Patras","kwstasthom",2014,'6.3'),
 ("Financial Sience","University of Patras","giannisadam",2001,'5.8'),
 ("Financial Sience","University of Patras","athinamar",2014,'7.5'),
 ("Financial Sience","University of Patras","nikosthan",2009,'9.2'),
 ("Financial Sience","ASOE","georgiasaim",2018,'7.6'),
 ("Financial Sience","ASOE","sofialap",2017,'8'),
 ("Financial Sience","ASOE","anastasiarep",2009,'7.1'),
 ("Financial Sience","ASOE","evaggeliavotsi",2016,'6.4'),
 ("Financial Sience","ASOE","ioannasake",2011,'8.1'),
 ("Financial Sience","ASOE","xristospap",2019,'9');

SELECT * FROM has_degree;

/*PROJECT*/

INSERT INTO project(empl,num,descr,url) VALUES

("tonyzaf",1,"This project is about Financial Distress Accounting","https://www.researchgate.net/publication/342663780Financial_Distress_Accounting.pdf"),

("siliagk",2,"This porject is about Accounting in Shipping companies, The project is about Invenstemnts in Greece and the Implications.", "https://www.researchgate.net/publication/342665896Accounting_in_Shipping_companies.pdf,https://www.resrarchnet.net/publications/5896Invenstemnts_Greece_Implications.pdf"),

("mixaltop",1,"This project is about Banking Services and Community Law.", "https://www.projectsonline.net/public/1025896Banking_Services_Community_Law.pdf"),

("eyangeo",1,"This project is about Single Europeans Loans.", "https://www.github.net/publications/254815621Single_Europeans_Loans.pdf"),

("chrisboul",2,"This project is about Financial Investigation of Accounting Statemants of Industrial Enterprise. This project is about Bussiness Communication between Organizations of Different Cultures", "https://www.researchgate.net/publication/345874126Financial_Investigation.pdf, https://www.resrarchnet.net/publications/58458796Bussiness_Communications.pdf"),

("elengeo",1,"This project is about Employee Insurance.", "https://star-tree.eu/images/publications/project/Employee_Insurance.pdf"),

("andrpi",1,"This project is about The Role of Multinational Enterprises and the World Bank in the Developing World.", "https://dialls2020.eu/project-publications/154872Andreas_Pikas_Multinational_Enterprises.pdf"),

("anastasila",1,"This project is about The Concept Role of the Leader in a Company.", "https://www.projectsonline.net/public/1024755Anastasia_Laskaridi_Leader_Company.pdf"),

("sofialap",1,"This project is about Banking Services and Community Law.", "https://www.projectsonline.net/public/102584Sofia_Lapari_Banking_Services.pdf"),

("anastasiarep",1,"The project is about Invenstemnts in Germany and Impications.", "https://www.projectsonline.net/public/1025896_Anastasia-Repopoulou_Germany_Investments.pdf"),

("evaggeliavotsi",2,"This porject is about Accounting in Financial companies,This projet is about Jail Expenses.", "https://www.projectsonline.net/public/1024587_Evaggelia_Votsi_Fiancial_Companies.pdf,https://dialls2020.eu/project-publications/154125_Evaggelia_Votsi_Jail_Expenses.pdf"),

("ioannasake",1,"This project is about the concept of a Manager in a company.", "https://www.worldprojectsonline.net/public/10236589Ioanna_Sakelaridi_Manager_in_a_company.pdf"),

("xristospap",1,"This project is about the World Economic Crisis.", "https://www.onlineprojects/public/10236984_Xristos_Papagiannis_World_Economic_Crisis.pdf");

SELECT * FROM project;

/*LANGUAGES*/

INSERT INTO languages(employee,lang) VALUES

("tonyzaf",'EN,GR'),

("mhtsakil",'EN,GR'),

("siliagk",'GR,EN,FR'),

("maroutsos",'GR,EN,FR,SP'),

("mixaltop",'GR,EN'),

("yiannisxA",'GR'),

("elenisar",'GR,EN'),

("eyangeo",'GR,FR,EN'),

("chrisboul",'GR,EN,FR'),

("andrpi",'GR,EN'),

("kwstakis93",'GR,EN'),

```

("elengeo",'GR,FR,EN'),
("anastasila",'GR,EN'),
("nikosgat",'GR,FR'),
("gianStam",'GR'),
("mariadok",'EN'),
("elenipapaxri",'EN'),
("georgioik",'EN,FR'),
("basilikigeo",'EN'),
("dimitriskous",'EN,GR'),
("kwstasthom",'EN'),
("giannisadam",'EN'),
("athinamar",'EN'),
("nikosthan",'EN,GR'),
("georgiasaim",'EN,FR'),
("sofialap",'EN'),
("anastasiarep",'EN,FR'),
("evaggeliavotsi",'EN'),
("ioannasake",'EN,GR'),
("xristospap",'EN');
SELECT * FROM languages;

```

```

/*JOB*/

```

```

INSERT INTO job(job_id,potition,afm_company,base,salary,upload_date,deadline_date,evaluator) VALUES
(2001,"Assistant Accountant",'101235645','Athens','980','2021-01-04 12:05:45','2021-02-05 12:00:00','yianstamel"),
(2002,"Accountant",'101879123','Thessaloniki','1350','2021-01-05 11:25:31','2021-03-23 23:00:00','savvas94"),
(2003,"Assistant Accountant",'101879123','Thessaloniki','934','2021-01-12 15:02:25','2021-02-12
23:00:00','manoumanou99"),
(2004,"Assistant Manager",'101235645','Athens','1425','2021-01-23 12:00:00','2021-02-23 23:00:00','nikolakis"),
(2005,"Accountant",'101235645','Athens','1225','2021-01-03 11:25:45','2021-03-03 23:00:00','yianstamel"),
(2006,"Assistant Manager",'101879123','Thessaloniki','1956','2021-01-15 12:00:00','2021-02-15
23:00:00','manoumanou99"),
(2007,"Accountant",'101879123','Thessaloniki','1268','2021-01-23 23:00:00','2021-03-23 23:00:00','manoumanou99"),
(2099,"Advisor",'101456789','Athens','1500','2021-01-02 13:10:04','2021-01-20 12:00:00','marakioik"),
(2098,"Advisor",'101456789','Athens','1500','2021-01-02 14:10:04','2021-01-30 12:00:00','marakioik"),
(2097,"Assistant Advisor",'101456789','Athens','1200','2021-01-05 09:10:04','2021-02-01 12:00:00','marakioik"),

```

```
(2096,"Secretary",'101456789','Athens','1000','2021-01-04 10:10:04','2021-02-06 12:00:00','kwstas96"),
(2095,"Assistant Manager",'101456789','Athens','2000','2021-01-09 15:10:04','2021-01-25 12:00:00','kwstas96"),
(2094,"Advisor",'101879123','Thessaloniki','1500','2021-01-11 10:10:21','2021-01-30 12:00:00','manoumanou99");
SELECT * FROM job;
#DELETE FROM job;
```

```
/*ANTIKEIMENA*/
```

```
INSERT INTO antikeimena(title,description,belongs_to) VALUES
```

```
("Accounting","Average knowledge of accounting for an Assistant Accountant or an average accountant.", "Accounting"),
("Accounting and Economics","Good knowledge of Accounting and Economics.", "Accounting"),
("Economics and Business","Very good knowledge of Economics to help a business flourish.", "Economics"),
("Economics and Statistics","Perfect knowledge of Economics and Statistics", "Economics"),
("Accounting and Financials","The perfect combination for an Accountant", "Accounting"),
("Accounting and Business","Very good knowledge of accounting to help a business or keep the bookkeeping of a business.", "Accounting"),
('Financial Knowledge','Advanced','Financial'),
('Financial Knowledge Average','Average','Financial'),
('Computer Knowledge','Average','Computer'),
('Team Management Experience','Advanced','Management'),
('Negotiation Skills','Advanced','Management'),
('Organization Skills','Experienced','Organization'),
('Communication Skills','Advanced','Communication');
SELECT * FROM antikeimena;
```

```
/*NEEDS*/
```

```
INSERT INTO needs(job,antikeimena_title) VALUES
```

```
('2001',"Accounting"),
('2002',"Accounting and Financials"),
('2003',"Economics and Statistics"),
('2004',"Economics and Business"),
('2005',"Accounting and Financials"),
('2006',"Accounting and Financials"),
('2007',"Economics and Business"),
(2099,'Financial Knowledge'),
(2099,'Communication Skills'),
```

```

(2099,'Computer Knowledge'),
(2098,'Financial Knowledge'),
(2098,'Communication Skills'),
(2098,'Computer Knowledge'),
(2094,'Financial Knowledge'),
(2094,'Communication Skills'),
(2094,'Computer Knowledge'),
(2097,'Financial Knowledge Avarege'),
(2097,'Computer Knowledge'),
(2096,'Computer Knowledge'),
(2096,'Organaization Skills'),
(2095,'Financial Knowledge'),
(2095,'Negotiation Skills'),
(2095,'Team Managment Experience');
SELECT * FROM needs;

```

```

/*REQUEST_EVALUATION*/

```

```

INSERT INTO request_evaluation(empl_usrnm,jobid) VALUES
("basilikigeo",2095),
("dimitriskous",2095),
("kwstasthom",2095),
("georgiasaim",2095),
("basilikigeo",2098),
("giannisadam",2097),
("kwstasthom",2098),
("ioannasake",2094);
SELECT * FROM request_evaluation;

```

```

/*PROCESS1*/

```

```

INSERT INTO process_1(em_usrnm,id_empl,id_eva,jb_id,grade_1,comments_1) VALUES
("basilikigeo",1504,2301,2095,3,'Very good ibterview/Good bio/Years of experience.'),
("dimitriskous",1505,2301,2095,4,'Very good ibterview/Good bio/Years of experience'),
("kwstasthom",1506,2302,2095,3,'Good in total'),
("georgiasaim",1508,2302,2095,NULL,""),

```

```

("basilikigeo",1504,2302,2098,4,'Excellent bio/years of exp/talkative'),
("ioannasake",1514,2360,2094,4,'Outstanding behavior.Good bio'),
("giannisadam",1507,2302,2097,2,'Average in total'),
("kwstasthom",1506,2302,2098,2,'Average in total');

SELECT * FROM process_1;

```

```

/*PROCESS2*/

```

```

INSERT INTO process_2(id_manager,id_eval,usnm_empl,id_employee,id_job,report,grade_2) VALUES

(1001,2301,"basilikigeo",1504,2095,'Exchellent in general',4),
(1001,2301,"dimitriskous",1505,2095,'Good employee',3),
(1001,2301,"kwstasthom",1506,2095,NULL,NULL),
(1001,2301,"georgiasaim",1508,2095,'Average in total',2),
(1002,2360,"ioannasake",1514,2094,'Exchellent and good skills',4),
(1001,2302,"giannisadam",1507,2097,'Average/not intersting',2),
(1001,2302,"kwstasthom",1506,2098,'Average employee not the best.',2),
(1001,2302,"basilikigeo",1504,2098,NULL,2);

SELECT * FROM process_2;

```

```

/*PROCESS3*/

```

```

INSERT INTO process_3(usrn_em,id_emplo,id_evalu,id_jb,grade_3) VALUES

("basilikigeo",1504,2301,2095,NULL),
("dimitriskous",1505,2301,2095,NULL),
("kwstasthom",1506,2302,2095,1),
("georgiasaim",1508,2302,2095,1),
("ioannasake",1514,2360,2094,2),
("basilikigeo",1504,2302,2098,2),
("kwstasthom",1506,2302,2098,2),
("giannisadam",1507,2302,2097,NULL);

SELECT * FROM process_3;

```

```

/*EVALUATION_RESULT*/

```

```

INSERT INTO evaluation_result(Evid,empl_username,idjb,jb_position,rgrade_1,rgrade_2,rgrade_3,grade,comments)
VALUES

(2301,'basilikigeo',2095,"Assistant Manager",3,4,2,9,'Very good candidate for the position.'),
(2301,"dimitriskous",2095,"Assistant Manager",4,4,2,10,'Very good candidate for the position.'),

```

```
(2302,"kwstasthom",2095,"Assistant Manager",3,3,NULL,NULL,'Average candidate not the best but not the worst.'),
(2302,"georgiasaim",2095,"Assistant Manager",NULL,NULL,1,NULL,'Not the best candidate.'),
(2360,'ioannasake',2094,"Advisor",4,4,2,NULL,"Exchellent candidate for the position."),
(2302,"giannisadam",2097,"Assistant Advisor",2,2,NULL,NULL,NULL),
(2302,"kwstasthom",2098,"Advisor",2,2,2,NULL,'Average in general.'),
(2302,'basilikigeo',2098,"Advisor",4,2,NULL,NULL,NULL);

SELECT * FROM evaluation_result;
```

```
INSERT INTO evaluator_stats(sevaluatorid,sevaluator_username,avgrade) VALUES
```

```
(2359,"nikolakis",NULL),
(2360,"manoumanou99",NULL),
(2361,"gianstamel",NULL),
(2362,"savvas94",NULL),
(2301,"kwstas96",NULL),
(2302,"marakioik",NULL);
```

Procedures:

- Πρώτο Procedure, στο οποίο πρέπει να δίνουμε το όνομα και το επώνυμο ενός υπαλλήλου και να βλέπουμε ποιες αξιολογήσεις έχει κάνει και αν και ποιες έχουν αποτέλεσμα τελικό αξιολόγησης, αλλιώς να εμφανίζει το κατάλληλο μήνυμα. Εδώ σκεφτήκαμε, πως θα χρειαστεί ένας cursor δηλαδή ένας δείκτης για να πηγαίνει στα στοιχεία του πίνακα evaluation_result και να δείχνει εάν υπάρχει βαθμός και δεν είναι NULL, ώστε να εμφανίζει τις τελικές αξιολογήσεις. Με την χρήση ενός LOOP δηλαδή ενός βρόχου, βάζουμε τον δείκτη να πηγαίνει και να δείχνει πολλές φορές ώστε να μην προσπερνά κανένα IF (συνθήκη), διότι θέλουμε να εμφανίζει κι το μήνυμα ότι υπάρχει κι αξιολόγηση που να μην έχει φτάσει την τελική φάση.

```
/*FIRST*/
```

```
DROP PROCEDURE IF EXISTS requests;
```

```
DELIMITER $
```

```
CREATE PROCEDURE requests(IN onoma VARCHAR(25), IN epitheto VARCHAR(25))
```

```
BEGIN
```

```
DECLARE usnm varchar(30) ; Δήλωση μεταβλητής, για να βρούμε το username του υπαλλήλου.
```

```
DECLARE evusername VARCHAR(30); Δήλωση μεταβλητής για να βρούμε το username του evaluator.
```

```
DECLARE GResult INT(4); Δήλωση μεταβλητής για να βρούμε το αποτέλεσμα της τελικής αξιολόγησης.
```

```
DECLARE done INT DEFAULT 0 ; Δήλωση μεταβλητής για συνθήκη τέλους κέρσρα.
```

```
DECLARE noresultcurs CURSOR FOR SELECT grade,empl_username FROM evaluation_result; Δηλώνουμε τον δείκτη μας, ώστε να δείχνει στον βαθμό και το username από το evaluation_result .
```

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

```
SELECT username INTO usnm FROM `user` WHERE name=onoma AND surname=epitheto; Για όταν δίνουμε το όνομα και το επίθετο , κατευθείαν να βρίσκει ότι το username του χρήστη/υπαλλήλου .
```

```
SELECT * FROM request_evaluation WHERE empl_usnm=usnm; Να βρίσκει τις αξιολογήσεις του συγκεκριμένου υπαλλήλου.
```

```
OPEN noresultcurs; Έπειτα ανοίγουμε τον cursor και εκτελεί τα αναφερόμενα .
```

```
notfinished_loop: LOOP
```

```
FETCH noresultcurs INTO GResult,evusername;
```

```
IF done THEN
```

```
LEAVE notfinished_loop;
```

```
END IF;
```

```
IF GResult IS NOT NULL AND evusername=usnm THEN Συνθήκη για το αν ο τελικός βαθμός δεν είναι NULL τότε να εκτυπώνει την τελική αξιολόγηση και το username του αξιολογητή, αλλιώς να εκτυπώνει ότι η αξιολόγηση βρίσκεται σε εξέλιξη(δηλαδή το κατάλληλο μήνυμα).Και να εκτυπώνει τους βαθμούς μέχρι ώρα δηλαδή grade_1 ,grade_2, grade_3 για να βλέπουμε μέχρι που έχει αξιολογηθεί ο συγκεκριμένος υπάλληλος.
```

```
SELECT * FROM evaluation_result WHERE empl_username=usnm and grade is not null;
```

```
SELECT name,surname FROM user INNER JOIN evaluator ON evaluator_username=username  
INNER JOIN evaluation_result ON evaluator_id=Evid WHERE empl_username=usnm; ;
```

```
ELSEIF GResult IS NULL AND evusername=usnm THEN
```

```
SELECT'REQUEST IN PROGRESS';
```

```
SELECT idjb,rgrade_1,rgrade_2, rgrade_3,comments FROM evaluation_result WHERE empl_username=usnm AND  
grade IS NULL;
```

```
END IF;
```

```
END LOOP;
```

```
CLOSE noresultcurs;
```

```
END$
```

```
DELIMITER ;
```

- Δεύτερο Procedure, στο οποίο πρέπει να δίνουμε το ID της δουλειάς στην οποία ο υπάλληλος έχει κάνει αίτηση κι το ID του αξιολογητή, ώστε να μπορούμε να περάσουμε τον τελικό βαθμός προσθέτοντας τους grade_1, grade_2, grade_3 και να βγάξει το κατάλληλο μήνυμα ή να εμφανίζει ότι δεν έχει ολοκληρωθεί η αξιολόγηση. Εδώ σκεφτήκαμε πως θα πρέπει να χρησιμοποιήσουμε δείκτη και βρόχο πάλι μαζί για να μπορεί πηγαίνει στα στοιχεία Evid,idjb,rgrade_1,rgrade_2, rgrade_3, grade,empl_username του πίνακα evaluation_result και να δείχνει εάν οι βαθμοί δεν είναι NULL, τότε με την χρήση της IF(συνθήκη) να κάνει UPDATE (ανανέωση) το πίνακα evaluation_result και να προσθέτει τους τελικούς βαθμούς στο grade. Και να εκτυπώνει τα το ID της δουλειάς, το ID αξιολογητή και το username του υπαλλήλου. Εάν κάποιος από τους τρεις βαθμούς είναι NULL τότε εκτυπώνει το κατάλληλο μήνυμα.


```
/*SECOND*/
```

```
DROP PROCEDURE IF EXISTS complition_check;
```

```
DELIMITER $
```

```
CREATE PROCEDURE complition_check(IN job_idP2 INT(4), IN evaluator_idP2 INT(4))
```

```
BEGIN
```

```
//Τρία πρώτα DECLARE είναι δήλωση μεταβλητών για τα grade_1, grade_2, grade_3 τα οποία παίρνει από τον πίνακα evaluation_result σαν rgrade_1, rgrade_2, rgrade_3 .
```

```
DECLARE G12 INT(4);
```

```
DECLARE G22 INT(4);
```

```
DECLARE G32 INT(4);
```

```
DECLARE GResult2 INT(4); Δήλωση μεταβλητής για τον τελικό βαθμό.
```

```
DECLARE done2 INT DEFAULT 0 ; Δήλωση μεταβλητής για τον δείκτη τότε πρέπει να τελειώσει.
```

```
// Δήλωση μεταβλητών για το ID της δουλειάς και του ID αξιολογητή.
```

```
DECLARE JOB_IDD2 INT(4);
```

```
DECLARE EJOB_IDD2 INT(4);
```

```
DECLARE EEV_IDDD2 INT(4);
```

```
DECLARE EV_IDD2 INT(4) ;
```

```
DECLARE EUSRNAME2 VARCHAR(30); Δήλωση μεταβλητής για το ψευδώνυμο του υπαλλήλου.
```

```
DECLARE finalizingcursor CURSOR FOR SELECT Evid,idjb,rgrade_1,rgrade_2, rgrade_3, grade,empl_username FROM evaluation_result; Δηλώνουμε τον δείκτη μας, για τα Evid,idjb,rgrade_1,rgrade_2, rgrade_3, grade,empl_username από τον πίνακα evaluation_result.
```

```
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done2=1;
```

```
SELECT job_idP2 INTO EJOB_IDD2; Βάζουμε το job_idP2 από το IN στο declare που κάναμε πιο πάνω.
```

```
SELECT evaluator_idP2 INTO EEV_IDDD2; Βάζουμε το evaluator_idP2 από το IN στο declare που κάναμε πιο πάνω.
```

```
OPEN finalizingcursor;
```

```
loopi : LOOP
```

```
FETCH finalizingcursor INTO EV_IDD2,JOB_IDD2,G12,G22,G32,GResult2,EUSRNAME2;
```

```
IF done2 THEN
```

```
LEAVE loopi;
```

```
END IF;
```

```
Η συνθήκη που αναφέραμε πιο πάνω.
```

```
IF G12 IS NOT NULL AND G22 IS NOT NULL AND G32 IS NOT NULL AND GResult2 IS NULL AND EJOB_IDD2=JOB_IDD2
AND EEV_IDDD2=EV_IDD2 THEN
```

```
UPDATE evaluation_result SET grade = G12 + G22 + G32 WHERE idjb=JOB_IDD2 AND
Evid=EV_IDD2 AND empl_username=EUSRNAME2;
```

```
SELECT * FROM evaluation_result WHERE idjb=JOB_IDD2 AND Evid=EV_IDD2 AND empl_username=EUSRNAME2;
```

```
END IF;
```

```
END LOOP;
```

```
CLOSE finalizingcursor;
```

```
SELECT 'THE FINISHED EVALUATIONS ARE FINAL OR NO FINISHED EVALUATION FOUND';
```

```
END$
```

```
DELIMITER ;
```

- Τρίτο procedure, στο οποίο δίνουμε το ID μίας δουλειάς και μας εμφανίζει τις οριστικοποιημένες αξιολογήσεις της δουλειάς, αλλιώς σε φθίνουσα σειρά, αλλιώς να εμφανίζει πόσες εκκρεμούν. Εδώ σκεφτήκαμε, πως και εδώ θα χρησιμοποιήσουμε δείκτη (cursor) και βρόχο. Ο δείκτης θα δείχνει μέσα στο βρόχο στο ID της δουλειάς και στο τελικό αποτέλεσμα της evaluation_result . Με συνθήκη IF δείχνουμε αν ο τελικός βαθμός είναι Null, εάν είναι NULL, τότε προσθέτουμε σε έναν counter 1(αυτές είναι οι μη οριστικοποιημένες αξιολογήσεις). Έπειτα, με συνθήκη IF, ελέγχουμε αν υπάρχουν αξιολογήσεις, αν δεν υπάρχουν εμφανίζεται το κατάλληλο μήνυμα, αλλιώς αν υπάρχουν αξιολογήσεις, τότε εμφανίζει τις τελικές αξιολογήσεις σε φθίνουσα σειρά και τις μη τελικές αξιολογήσεις σε φθίνουσα σειρά και με την χρήση μίας μεταβλητής υπολογίζουμε πόσες ακόμη μένουν για αξιολόγηση. Αλλιώς εμφανίζει τις τελικές αξιολογήσεις, εφόσον καμία αξιολόγηση από τη συγκεκριμένη δουλειά δεν είναι υπό επεξεργασία, σε φθίνουσα σειρά .

```
/*THIRD*/
```

```
DROP PROCEDURE IF EXISTS valid_requests;
```

```
DELIMITER $
```

```
CREATE PROCEDURE valid_requests(IN id_j INT(4))
```

```
BEGIN
```

```
// Δήλωση μεταβλητών για τα ID της δουλειάς και το αποτέλεσμα.
```

```
DECLARE JOB_IDD3 INT(4);
```

```
DECLARE EJOB_IDD3 INT(4);
```

```
DECLARE GRESULT3 INT(4);
```

```
DECLARE not_found3 INT; Δήλωση μεταβλητής για συνθήκη τέλους κέρσορα.
```

```
DECLARE counter int DEFAULT 0;
```

```
DECLARE MISSNIG INT(4);
```

```
DECLARE gcursor CURSOR FOR SELECT idjb,grade FROM evaluation_result; Δηλώνουμε τον δείκτη μας, στα idjb,grade από τον πίνακα evaluation_result.
```

```

DECLARE CONTINUE HANDLER FOR NOT FOUND SET not_found3=1;

SELECT id_j INTO JOB_IDD3;

OPEN gcursor;

loopy3 : LOOP
  FETCH gcursor INTO EJOB_IDD3,GRESULT3;

  IF not_found3 THEN
    LEAVE loopy3;
  END IF;
  IF JOB_IDD3=EJOB_IDD3 AND GRESULT3 IS NULL THEN
    SET counter = counter + 1 ;
  END IF;
END LOOP;
CLOSE gcursor;

IF (SELECT COUNT(*) FROM request_evaluation WHERE JOB_IDD3=jobid )=0 THEN
  SELECT 'NO AVAILABLE APLICATIONS';
ELSEIF counter>0 THEN
  SELECT * FROM evaluation_result WHERE idjb=JOB_IDD3 AND grade is not null ORDER BY (grade) DESC;
  SELECT COUNT(*) FROM evaluation_result WHERE idjb=JOB_IDD3 AND grade is null INTO MISSNIG;
  SELECT 'EVALUATION IN PROGRESS, STILL' , MISSNIG , 'MISSING EVALUATIONS';
ELSE
  SELECT 'FINALL RESULTS';
  SELECT * FROM evaluation_result WHERE idjb=JOB_IDD3 ORDER BY (grade) DESC;
END IF;
END$
DELIMITER ;

```

Triggers:

- Τα πρώτα εννέα Triggers αφορούν τα job, employee και request_evaluation σε θέμα INSERT UPDATE και DELETE . Για κάθε ένα πίνακα δημιουργήσαμε 3 Triggers που το κάθε ένα έχει τις ενέργειες που προ-αναφέραμε. Με την χρήση του IF :
 - Στο πίνακα job, δημιουργήσαμε τα triggers του, ώστε κάθε φορά που βάζουμε καινούργια δουλειά, ή κάνουμε ενημέρωση τον πίνακα ή διαγράφουμε κάποια δουλειά να εμφανίζει

Success ή Fail, ανάλογα εάν αυτό που κάναμε πέτυχε ή απέτυχε. Έτσι θα ενημερώνεται ο πίνακας log με τις ενέργειες μας.

- Στο πίνακα employee με την ίδια λογική πάλι, κάθε φορά που βάζουμε καινούργιο υπάλληλο, ή κάνουμε ενημέρωση τον πίνακα ή διαγράφουμε κάποιον υπάλληλο, να εμφανίζει Success ή Fail, ανάλογα εάν αυτό που κάναμε πέτυχε ή απέτυχε. Έτσι θα ενημερώνεται ο πίνακας log με τις ενέργειες μας.
- Στο πίνακα request_evaluation , δημιουργήσαμε τα triggers του, ώστε κάθε φορά που βάζουμε καινούργια αίτηση, ή κάνουμε ενημέρωση τον πίνακα ή διαγράφουμε κάποια αίτηση να εμφανίζει Success ή Fail, ανάλογα εάν αυτό που κάναμε πέτυχε ή απέτυχε. Έτσι θα ενημερώνεται ο πίνακας log με τις ενέργειες μας.

```
/*FIRST 9 TRIGGERS OF EXERCISE 1*/
```

```
/*JOB*/
```

```
DROP TRIGGER IF EXISTS trig_log_jobin;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_log_jobin BEFORE INSERT ON job FOR EACH ROW
```

```
BEGIN
```

```
DECLARE datetrig1 DATETIME;
```

```
DECLARE succsestrig1 ENUM ('SUCCSED','FAILED');
```

```
IF new.job_id IS NOT NULL THEN
```

```
    SET succsestrig1='SUCCSED';
```

```
ELSE
```

```
    SET succsestrig1='FAILED';
```

```
END IF;
```

```
SET datetrig1=now();
```

```
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
```

```
    #TO USERNAME THA EINAI AUTO ME TO OPOIO THA KANOUME LOGIN STO APP,NOMIZW DEN  
    MPOUROUME NA TO GRAPSOUME MEXRI NA ARXISOUME TO GUI
```

```
('job',datetrig1,'INSERT',succsestrig1);
```

```
END$
```

```
DELIMITER ;
```

```
SHOW TRIGGERS LIKE 'job';
```

```
DROP TRIGGER IF EXISTS trg_log_jobup;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_log_jobup BEFORE UPDATE ON job FOR EACH ROW
```

```

BEGIN
DECLARE datetrig2 DATETIME;
DECLARE succsestrig2 ENUM ('SUCCSED','FAILED');
IF old.job_id<>new.job_id OR old.afm_company<>new.afm_company OR old.upload_date<>new.upload_date THEN
    SET new.job_id=old.job_id , new.afm_company=old.afm_company , new.upload_date=old.upload_date ;
END IF;

IF new.potition=old.potition AND new.base=old.base AND new.salary=old.salary AND
new.deadline_date=old.deadline_date AND new.evaluator=old.evaluator THEN

    SET succsestrig2='FAILED';
ELSE
    SET succsestrig2='SUCCSED';
END IF;

SET datetrig2=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('job',datetrig2,'UPDATE',succsestrig2);
END$

DELIMITER ;
SHOW TRIGGERS LIKE 'job';

```

```

DROP TRIGGER IF EXISTS trig_log_jobdel;
DELIMITER $
CREATE TRIGGER trig_log_jobdel AFTER DELETE ON job FOR EACH ROW
BEGIN
DECLARE datetrig3 DATETIME;
DECLARE succsestrig3 ENUM ('SUCCSED','FAILED');
IF old.job_id IS NOT NULL THEN
    SET succsestrig3='SUCCSED';
ELSE
    SET succsestrig3='FAILED';
END IF;

SET datetrig3=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('job',datetrig3,'DELETE',succsestrig3);
END$

DELIMITER ;

```

```
SHOW TRIGGERS LIKE 'job';
```

```
/*EMPLOYEE*/
```

```
DROP TRIGGER IF EXISTS trig_log_emplin;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_log_emplin AFTER INSERT ON employee FOR EACH ROW
```

```
BEGIN
```

```
DECLARE datetrig4 DATETIME;
```

```
DECLARE succsestrig4 ENUM ('SUCCSED','FAILED');
```

```
IF new.empl_id IS NOT NULL THEN
```

```
    SET succsestrig4='SUCCSED';
```

```
ELSE
```

```
    SET succsestrig4 ='FAILED';
```

```
END IF;
```

```
SET datetrig4=NOW();
```

```
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
```

```
('employee',datetrig4,'INSERT',succsestrig4);
```

```
END$
```

```
DELIMITER ;
```

```
SHOW TRIGGERS LIKE 'employee';
```

```
DROP TRIGGER IF EXISTS trig_log_emplup;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_log_emplup BEFORE UPDATE ON employee FOR EACH ROW
```

```
BEGIN
```

```
DECLARE datetrig5 DATETIME;
```

```
DECLARE succsestrig5 ENUM ('SUCCSED','FAILED');
```

```
IF new.employee_username<>old.employee_username OR new.empl_id<>old.empl_id OR new.cmp_afm<>old.cmp_afm  
THEN
```

```
    SET new.employee_username=old.employee_username , new.empl_id=old.empl_id, new.cmp_afm=old.cmp_afm;
```

```
END IF;
```

```
IF new.expe_years=old.expe_years AND new.bio=old.bio AND new.sistatikes=old.sistatikes AND  
new.certificates=old.certificates AND new.awards=old.awards THEN
```

```

        SET succsestrig5='FAILED';
ELSE
        SET succsestrig5='SUCCSED';
END IF;
SET datetrig5=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('employee',datetrig5,'UPDATE',succsestrig5);
END$
DELIMITER ;
SHOW TRIGGERS LIKE 'employee';

```

```

DROP TRIGGER IF EXISTS trig_log_empldel;
DELIMITER $
CREATE TRIGGER trig_log_empldel BEFORE DELETE ON employee FOR EACH ROW
BEGIN
DECLARE datetrig6 DATETIME;
DECLARE succsestrig6 ENUM ('SUCCSED','FAILED');
IF old.empl_id IS NOT NULL THEN
        SET succsestrig6='SUCCSED';
ELSE
        SET succsestrig6='FAILED';
END IF;
SET datetrig6=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('employee',datetrig6,'DELETE',succsestrig6);
END$
DELIMITER ;
SHOW TRIGGERS LIKE 'employee';

```

```

/*REQUEST EVALUATION*/
DROP TRIGGER IF EXISTS trig_log_requin;
DELIMITER $
CREATE TRIGGER trig_log_requin BEFORE INSERT ON request_evaluation FOR EACH ROW
BEGIN
DECLARE datetrig7 DATETIME;
DECLARE succsestrig7 ENUM ('SUCCSED','FAILED');

```

```

IF new.empl_usnm is not null and new.jobid is not null then
    SET succsestrig7='SUCCSED' ;
ELSE
    SET succsestrig7='FAILED' ;
END IF;
SET datetrig7=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('request_evaluation',datetrig7,'INSERT',succsestrig7);
END$
DELIMITER ;
SHOW TRIGGERS LIKE 'request_evaluation';

```

```

DROP TRIGGER IF EXISTS trig_log_requp;
DELIMITER $
CREATE TRIGGER trig_log_requp BEFORE UPDATE ON request_evaluation FOR EACH ROW
BEGIN
DECLARE datetrig8 DATETIME;
DECLARE succsestrig8 ENUM ('SUCCSED','FAILED');
IF new.empl_usnm<>old.empl_usnm THEN
    SET new.empl_usnm=old.empl_usnm;
END IF;
IF new.jobid=old.jobid THEN
    SET succsestrig8='FAILED' ;
ELSE
    SET succsestrig8='SUCCSED';
END IF;
SET datetrig8=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('request_evaluation',datetrig8,'UPDATE',succsestrig8);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS trig_log_reqdel;
DELIMITER $
CREATE TRIGGER trig_log_reqdel BEFORE DELETE ON request_evaluation FOR EACH ROW

```



```

BEGIN
DECLARE datetrig9 DATETIME;
DECLARE succsestrig9 ENUM ('SUCCSED','FAILED');
SET succsestrig9='SUCCSED';
SET datetrig9=NOW();
INSERT INTO log (table_edited,date_of_action,resault_of_action,result_succses) VALUES
('request_evaluation',datetrig9,'DELETE',succsestrig9);
END$
DELIMITER ;

```

- Στο δεύτερο ερώτημα για την δημιουργία trigger, για τον πίνακα company , σκεφτήκαμε πριν κάθε UPDATE στον πίνακα, με την χρήση συνθήκης IF για κάθε μία από τις μεταβλητές ξεχωριστά, να βάζουμε αν το παλιό(old)AFM, DOY, company_name δεν είναι ίδιο με το καινούργιο(new) που δίνει για να κάνει UPDATE ο χρήστης , θα δείχνει ότι γίνεται ενημέρωση, αλλά διατηρεί τις old μεταβλητές.

/*SECOND TRIGGER*/

```

DROP TRIGGER IF EXISTS trig_ch_cmp
DELIMITER $
CREATE TRIGGER trig_ch_cmp BEFORE UPDATE ON company FOR EACH ROW #mono manager mporei!!!!
BEGIN
IF NEW.company_name NOT LIKE OLD.company_name THEN
SET NEW.company_name=OLD.company_name;
END IF;
IF NEW.AFM NOT LIKE OLD.AFM THEN
SET NEW.AFM=OLD.AFM;
END IF;
IF NEW.DOY NOT LIKE OLD.DOY THEN
SET NEW.DOY=OLD.DOY;
END IF;
END$
DELIMITER ;

```

- Στο τελευταίο ερώτημα για τα Triggers, σκεφτήκαμε ότι για κάθε χρήστη δηλαδή για κάθε manager,evaluator ,employee και admin θα πρέπει να δημιουργηθούν ξεχωριστά triggers για κάθε έναν από αυτούς τους χρήστες. Με την χρήση IF συνθήκης και με το OLD & NEW στα username του κάθε χρήστη έχουμε ότι:
- Όπως σκεφτήκαμε και για το δεύτερο trigger έτσι κι εδώ όταν το παλιό employee_username δεν είναι ίδιο με το καινούργιο, δηλαδή αυτό που προσπαθεί να κάνει update , τότε θα διατηρεί το παλιό. Εφόσον μόνο ο admin έχει την δυνατότητα να αλλάξει τα δικά του στοιχεία.

```
/*THIRD TRIGGER*/
```

```
DROP TRIGGER IF EXISTS trig_not_auth_empl;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_not_auth_empl BEFORE UPDATE ON employee FOR EACH ROW
```

```
BEGIN
```

```
IF NEW.employee_username NOT LIKE OLD.employee_username THEN
```

```
    SET NEW.employee_username=OLD.employee_username;
```

```
END IF;
```

```
IF NEW.empl_id NOT LIKE OLD.empl_id THEN
```

```
    SET NEW.empl_id=OLD.empl_id;
```

```
END IF;
```

```
ENDS
```

```
DELIMITER ;
```

- Όπως σκεφτήκαμε και για το δεύτερο trigger έτσι κι εδώ όταν το παλιό manager_username δεν είναι ίδιο με το καινούργιο, δηλαδή αυτό που προσπαθεί να κάνει update , τότε θα διατηρεί το παλιό. Εφόσον μόνο ο admin έχει την δυνατότητα να αλλάξει τα δικά του στοιχεία.

```
DROP TRIGGER IF EXISTS trig_not_auth_man;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_not_auth_man BEFORE UPDATE ON manager FOR EACH ROW
```

```
BEGIN
```

```
IF NEW.manager_username NOT LIKE OLD.manager_username THEN
```

```
    SET NEW.manager_username=OLD.manager_username;
```

```
END IF;
```

```
IF NEW.manager_id NOT LIKE OLD.manager_id THEN
```

```
    SET NEW.manager_id=OLD.manager_id;
```

```
END IF;
```

```
ENDS
```

```
DELIMITER ;
```

- Όπως σκεφτήκαμε και για το δεύτερο trigger έτσι κι εδώ όταν το παλιό evaluator_username δεν είναι ίδιο με το καινούργιο, δηλαδή αυτό που προσπαθεί να κάνει update , τότε θα διατηρεί το παλιό. Εφόσον μόνο ο admin έχει την δυνατότητα να αλλάξει τα δικά του στοιχεία.

```
DROP TRIGGER IF EXISTS trig_not_auth_eva;
```

```
DELIMITER $
```

```
CREATE TRIGGER trig_not_auth_eva BEFORE UPDATE ON evaluator FOR EACH ROW
```

```

BEGIN
IF NEW.evaluator_username NOT LIKE OLD.evaluator_username THEN
    SET NEW.evaluator_username=OLD.evaluator_username;
END IF;
IF NEW.evaluator_id NOT LIKE OLD.evaluator_id THEN
    SET NEW.evaluator_id=OLD.evaluator_id;
END IF;
END$
DELIMITER ;

```

- Όπως σκεφτήκαμε και για το δεύτερο trigger έτσι κι εδώ όταν το παλιό admin_username δεν είναι ίδιο με το καινούργιο, δηλαδή αυτό που προσπαθεί να κάνει update , τότε θα διατηρεί το καινούργιο. Εφόσον μόνο ο admin έχει την δυνατότητα να αλλάξει τα δικά του στοιχεία.

```

DROP TRIGGER IF EXISTS trig_auth_admin;
DELIMITER $
CREATE TRIGGER trig_uth_admin BEFORE UPDATE ON administrator FOR EACH ROW
BEGIN
IF NEW.admin_username NOT LIKE OLD.admin_username THEN
    SET NEW.admin_username=OLD.admin_username;
END IF;
END$
DELIMITER ;

```

GUI

➤ Διευκρινίσεις :

- Καθώς ασχοληθήκαμε με την δημιουργία του GUI, είδαμε πως χρειάζεται να προσθέσουμε κάποια procedures και triggers στη βάση μας, ώστε να μας βοηθήσουν στην λειτουργία του προγράμματος μέσω της βάσης και όχι μέσω του κώδικα του Interface.
- Το πρώτο procedure είναι για τη λειτουργία του Sign in στο Interface. Ανάλογα με το username και password του κάθε user παραπέμπει στο ανάλογο interface της εφαρμογής, κρατώντας ένα counter ανάλογα εάν το username και το password που βάζουμε στο interface υπάρχει στη βάση και ανάλογα το table που βρίσκεται το username μας δίνει τιμή από το 1 έως το 4 σαν έξοδο. Έτσι μέσω του κώδικα της Java του interface καταλήγουμε στο ανάλογο παράθυρο. Το συγκεκριμένο procedure είναι ένα IN-OUT procedure, δηλαδή έχουμε δύο μεταβλητές που δίνουμε τιμές (siusername,sipassword) και επιστρέφει την μεταβλητή GUIPOINTER ανάλογα τον user που κάνει sign in .

```
/*PROCEDURE SIGN IN*/
```

```
DROP PROCEDURE IF EXISTS signin;  
DELIMITER $  
CREATE PROCEDURE signin(IN siusername VARCHAR(30), IN sipassword VARCHAR(20), OUT  
GUIPOINTER INT )  
BEGIN  
DECLARE signinusername VARCHAR(30);  
DECLARE signinpassword VARCHAR(20);  
DECLARE done INT DEFAULT 0;  
DECLARE counter INT DEFAULT 0;  
DECLARE cursorusername VARCHAR(30);  
DECLARE cursorpassword VARCHAR(20);  
DECLARE userexists INT DEFAULT 0;  
DECLARE manageruser VARCHAR(30);  
DECLARE employeeuser VARCHAR(30);  
DECLARE evaluatoruser VARCHAR(30);  
DECLARE adminuser VARCHAR(30);  
#DECLARE GUIPOINTER INT DEFAULT 0;  
DECLARE cursorManager CURSOR FOR SELECT manager_username FROM manager;  
DECLARE cursorEmployee CURSOR FOR SELECT employee_username FROM employee;  
DECLARE cursorEvaluator CURSOR FOR SELECT evaluator_username FROM evaluator;  
DECLARE cursorAdmin CURSOR FOR SELECT admin_username FROM administrator;  
DECLARE cursorusercheck CURSOR FOR SELECT username,`password` FROM `user`;  
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=1;  
SELECT siusername INTO signinusername;  
SELECT sipassword INTO signinpassword;  
#1ο MEROS Στο πρώτο LOOP ελέγχουμε εάν υπάρχει το username και το password στη βάση.  
OPEN cursorusercheck;
```

```
loopcheck : LOOP
```

```
FETCH cursorusercheck INTO cursorusername,cursorpassword;  
IF done THEN  
    LEAVE loopcheck;  
END IF;  
IF signinusername=cursorusername AND signinpassword=cursorpassword THEN
```

```

        SET counter=counter+1;
    END IF;
END LOOP;
CLOSE cursorusercheck;

IF counter=1 THEN
    SET userexists=1;
ELSE
    SET userexists=0;
END IF;

```

#2ο MEROS Στο δεύτερο μέρος με χρήση κέρσορα και loop αλλά και χρήση συνθήκης IF ελέγχουμε τι κατηγορία χρήστη είναι (manager,evaluator,employee και administrator)

```

IF userexists=1 THEN

    OPEN cursorManager;
    SET done=0;
    loopmanager : LOOP

    FETCH cursorManager INTO manageruser;

        IF done THEN
            LEAVE loopmanager;
        END IF;

        IF manageruser=signinusername THEN
            SET GUIPOINTER=1;
            #SELECT "USER IS A MANAGER";
        END IF;

    END LOOP;
    CLOSE cursorManager;

    OPEN cursorEvaluator;
    SET done=0;
    loopevaluator : LOOP

    FETCH cursorEvaluator INTO evaluatoruser;

        IF done THEN
            LEAVE loopevaluator;
        END IF;

        IF evaluatoruser=signinusername THEN
            SET GUIPOINTER=2;
            #SELECT "USER IS AN EVALUATOR";
        END IF;

    END LOOP;
    CLOSE cursorEvaluator;

    OPEN cursorEmployee;
    SET done=0;
    loopemployee : LOOP

    FETCH cursorEmployee INTO employeeuser;
    IF done THEN

```

```

        LEAVE loopemployee;
    END IF;

    IF      employeeuser=signinusername THEN
        SET GUIPOINTER=3;
        #SELECT "USER IS AN EMPLOYEE";
    END IF;

    END LOOP;
CLOSE cursorEmployee ;

OPEN cursorAdmin;
SET done=0;
loopadmmmin : LOOP

FETCH cursorAdmin INTO adminuser;
    IF done THEN
        LEAVE loopadmmmin;
    END IF;

    IF      adminuser=signinusername THEN
        SET GUIPOINTER=4;
        #SELECT "USER IS THE ADMINISTRATOR";
    END IF;

    END LOOP;
CLOSE cursorAdmin ;
SELECT GUIPOINTER;
ELSE
SET GUIPOINTER=5;
#SELECT "WRONG USERNAME OR PASSWORD.";
END IF;
END$
DELIMITER ;

```

- Δεύτερο procedure το οποίο υπολογίζει το average grade του evaluator βάση των τελικών βαθμών που έχει δώσει στις αξιολογήσεις του.

```

#SP POY YPOLOGIZEI TO AVG GRADE KAPOIOU EVALUATOR APO TO evaluation_result
DROP PROCEDURE IF EXISTS avggrade;
DELIMITER $
CREATE PROCEDURE avggrade (IN evaluatorsid int(4))
BEGIN
    DECLARE sum int default 0;
    DECLARE gradenum INT DEFAULT 0;
    DECLARE average FLOAT ;
    DECLARE done INT DEFAULT 0;
    DECLARE spevid INT(4);
    DECLARE sgrade INT(4);
    DECLARE inevid INT(4);
    DECLARE checkavg CURSOR FOR SELECT Evid,grade FROM evaluation_result;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=1;
    SELECT evaluatorsid INTO inevid;

    OPEN checkavg;

    loopavg : LOOP
    FETCH checkavg INTO spevid,sgrade;

```

```

IF done THEN
    LEAVE loopavg;
END IF;

IF inevid=spevid AND sgrade IS NOT NULL THEN
    SET sum=sum+sgrade;
    SET gradenum=gradenum+1;
END IF;
END LOOP;
SET average=sum / gradenum;
UPDATE evaluator_stats SET avgrade=average WHERE sevaluatorid=evaluatorsid;

CLOSE checkavg;
END$
DELIMITER ;

```

- Τρίτο procedure ανάλογα το Sign in username για να βρίσκουμε το AFM κάθε εταιρείας που έχει ο χρήστης για να έχουμε τα αποτελέσματα που θέλουμε για τις εργασίες του manager για την εταιρεία του.

```

#PROCEDURE DINEI TO AFM TIS ETAIRIAS POU DOULEUEI KAPOIOS USER
DROP PROCEDURE IF EXISTS AFM;
DELIMITER $
CREATE PROCEDURE AFM(IN afmusername VARCHAR(30), OUT AFMCOMPUSER INT )
BEGIN
SELECT manager_company FROM manager WHERE afmusername=manager_username INTO
AFMCOMPUSER;
END$
DELIMITER ;
CALL AFM("markzymn",@AFMCOMPUSER);
SELECT @AFMCOMPUSER;

```

- Procedure για τα labels για να μπαίνει το όνομα του χρήστη στο Interface στο Welcome message που έχουμε δημιουργήσει για να καλωσορίζει το χρήστη.

```

DROP PROCEDURE IF EXISTS TEST;
DELIMITER $
CREATE PROCEDURE TEST (IN usernametst VARCHAR(30) , OUT nametest VARCHAR(25))
BEGIN
SELECT `name` FROM `user` WHERE usernametst=username INTO nametest;
END $

```

- TRIGGERS για AFTER INSERT ΚΑΙ AFTER UPDATE για να ανανεώσει το μέσο όρο τελικών βαθμών κάποιου αξιολογητή όταν βάζει καινούργιο τελικό βαθμό καλώντας το procedure avggrade.

```

#update to avg grade sto evaluator_stats otan ginetai update bathmos sto result_evaluation
DROP TRIGGER IF EXISTS uavgradeevaluator;
DELIMITER $
CREATE TRIGGER uavgradeevaluator AFTER UPDATE ON evaluation_result FOR EACH ROW

```

```
BEGIN
```

```
CALL avggrade(NEW.Evid);
```

```
END$
```

```
DELIMITER ;
```

```
#update to avg grade sto evaluator_stats otan ginetai insert bathmos sto result_evaluation
```

```
DROP TRIGGER IF EXISTS iavgradeevaluator;
```

```
DELIMITER $
```

```
CREATE TRIGGER iavgradeevaluator AFTER INSERT ON evaluation_result FOR EACH ROW
```

```
BEGIN
```

```
CALL avggrade(NEW.Evid);
```

```
END$
```

```
DELIMITER ;
```

➤ Κώδικας Java για Interface για το GUI :

Αρχικά βλέπουμε το Sign in για το GUI , δηλαδή την αρχική οθόνη της εφαρμογής. Κάναμε σύνδεση τη Βάση μας μέσω του JDBC. Μέσω δύο text fields παίρνουμε το username & password από το χρήστη και καλούμε το procedure Sign in .

```
package AppPack;
```

```
import java.sql.Connection;
```

```
import java.sql.DriverManager;
```

```
import java.sql.SQLException;
```

```
import java.util.logging.Level;
```

```
import java.util.logging.Logger;
```

```
import java.sql.*;
```

```
public class SignInGUI extends javax.swing.JFrame {
```

```
    Connection con;
```

```
    public SignInGUI() {
```

```
        initComponents();
```

```
        createConnection();
```

```
    }
```



```

void createConnection() { Σύνδεση με DATABASE.
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/StaffEvaluation","root","Rastakir1");
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(DPPProject.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(DPPProject.class.getName()).log(Level.SEVERE, null, ex);
    }
}

}

@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jLabel1 = new javax.swing.JLabel();
    SignInUsername = new javax.swing.JTextField();
    jLabel2 = new javax.swing.JLabel();
    jLabel3 = new javax.swing.JLabel();
    SignInPassword = new javax.swing.JPasswordField();
    jToggleButton1 = new javax.swing.JToggleButton();
    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    jLabel1.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
    jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
    jLabel1.setText("Log-In");
    SignInUsername.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
    SignInUsername.setHorizontalAlignment(javax.swing.JTextField.CENTER);
    jLabel2.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
    jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
    jLabel2.setText("Username");
    jLabel2.setToolTipText("");
    jLabel3.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N

```

```

jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel3.setText("Password");

SignInPassword.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
SignInPassword.setHorizontalAlignment(javax.swing.JTextField.CENTER);
jToggleButton1.setText("OK");
jToggleButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jToggleButton1ActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(46, 46, 46)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.CENTER)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 150,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(SignInUsername, javax.swing.GroupLayout.DEFAULT_SIZE, 250,
Short.MAX_VALUE)
                .addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT_SIZE, 250,
Short.MAX_VALUE)
                .addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT_SIZE, 250,
Short.MAX_VALUE)
                .addComponent(SignInPassword, javax.swing.GroupLayout.DEFAULT_SIZE, 250,
Short.MAX_VALUE)
                .addComponent(jToggleButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(52, Short.MAX_VALUE))
        );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()

```

```

        .addGap(33, 33, 33)

        .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 45,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 38,
Short.MAX_VALUE)

        .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 35,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(18, 18, 18)

        .addComponent(SignInUsername, javax.swing.GroupLayout.PREFERRED_SIZE, 50,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(39, 39, 39)

        .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 35,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(18, 18, 18)

        .addComponent(SignInPassword, javax.swing.GroupLayout.PREFERRED_SIZE, 50,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(46, 46, 46)

        .addComponent(jToggleButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(53, 53, 53)
    );

    pack();
    setLocationRelativeTo(null);
} // </editor-fold>

```

```

private void jToggleButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    try {

        String username = SignInUsername.getText(); Παίρνουμε το εισαγόμενο password &
username στις αντίστοιχες μεταβλητές username & password

        String password = SignInPassword.getText();
    }
}

```

CallableStatement stmt1 = con.prepareCall("CALL TEST(?,?)"); Καλούμε το procedure TEST για να πάρουμε το όνομα του user που κάνει sign in για επόμενα interface.

```
stmt1.setString(1, username);  
stmt1.registerOutParameter(2, Types.VARCHAR);  
stmt1.executeQuery();  
String name = stmt1.getString(2);  
System.out.println(name);  
stmt1.close();
```

CallableStatement stmt = con.prepareCall("CALL signin(?,?,?)"); Καλούμε το procedure signin με input τις μεταβλητές που έχουμε πάρει και output το GUIPOINTER.

```
stmt.setString(1, username);  
stmt.setString(2, password);  
stmt.registerOutParameter(3, Types.INTEGER);  
  
stmt.executeQuery();  
int Guipointer = stmt.getInt(3);  
System.out.println(Guipointer);  
if (Guipointer == 1){  
    dispose();  
    new SignInManagerGUI(username, name).setVisible(true);  
}  
else if (Guipointer == 2){  
    dispose();  
    new SignInEvaluatorGUI().setVisible(true);  
}  
else if (Guipointer == 3){  
    dispose();  
    new SignInEmployeeGUI().setVisible(true);
```

```

    }
    else if (Guipointer == 4){
        dispose();
        new SignInAdminGUI().setVisible(true);
    }
    else if (Guipointer == 5){
        new SignInErrorGUI().setVisible(true);
    } Ανάλογα το output του procedure πάμε στο κατάλληλο interface.

```

```

stmt.close();

```

```

} catch (SQLException ex) {
    Logger.getLogger(SignInGUI.class.getName()).log(Level.SEVERE, null, ex);
}
}

```

```

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new SignInGUI().setVisible(true);
        }
    });
}

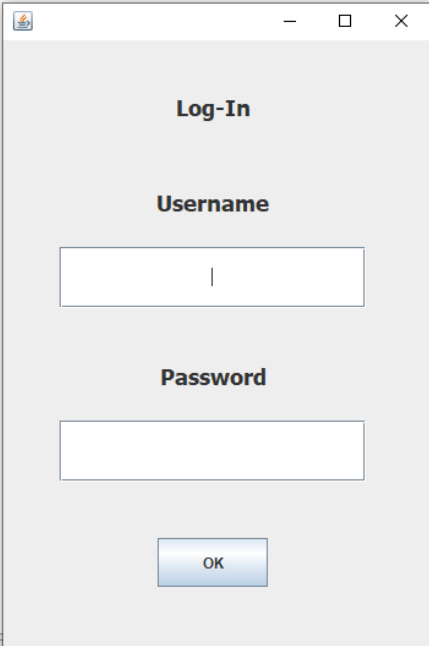
```

```

// Variables declaration - do not modify
private javax.swing.JPasswordField SignInPassword;
public javax.swing.JTextField SignInUsername;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JToggleButton jToggleButton1;
// End of variables declaration
}

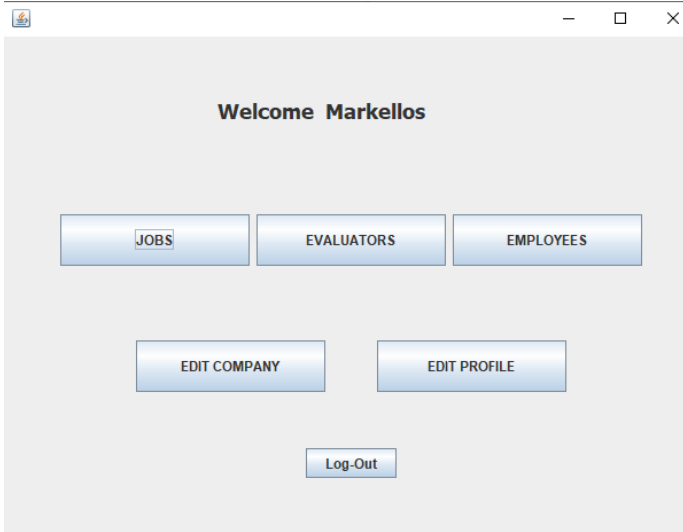
```

INTERFACE SIGN IN :



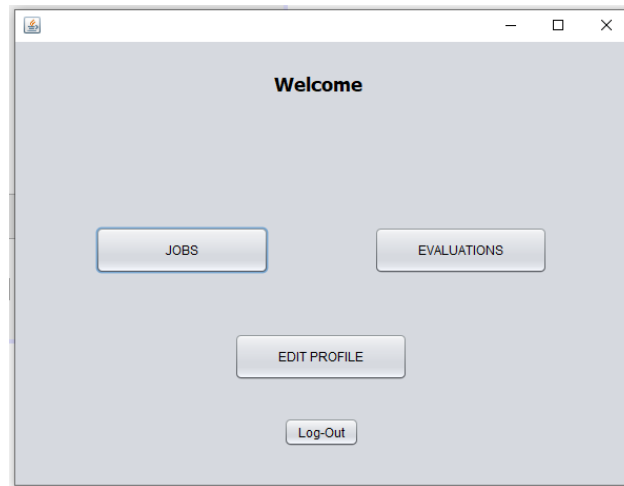
A screenshot of a 'Log-In' window. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main content area is light gray. At the top, the text 'Log-In' is centered. Below it, the label 'Username' is centered above a white text input field. Further down, the label 'Password' is centered above another white text input field. At the bottom center, there is a blue button with the text 'OK'.

Av GUIPOINTER =1 : (manager)

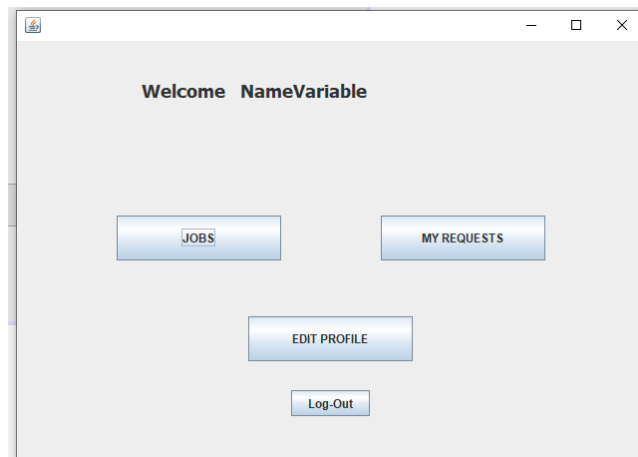


A screenshot of a manager dashboard window. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main content area is light gray. At the top, the text 'Welcome Markellos' is centered. Below this, there are three blue buttons arranged horizontally: 'JOBS', 'EVALUATORS', and 'EMPLOYEES'. Further down, there are two blue buttons arranged horizontally: 'EDIT COMPANY' and 'EDIT PROFILE'. At the bottom center, there is a blue button labeled 'Log-Out'.

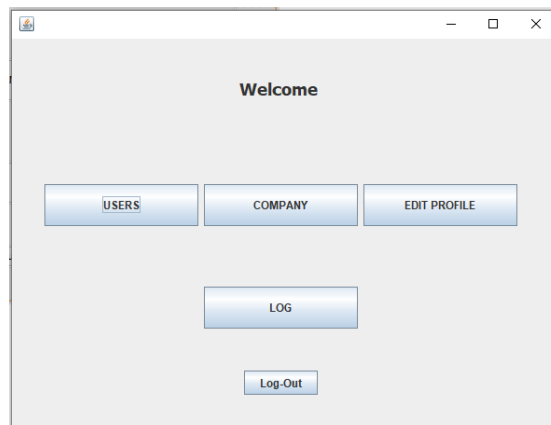
Av GUIPOINTER =2 : (evaluator)



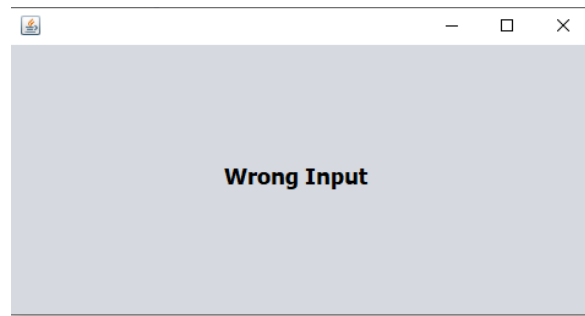
Av GUIPOINTER =3 : (employee)



Av GUIPOINTER =4 : (administrator)

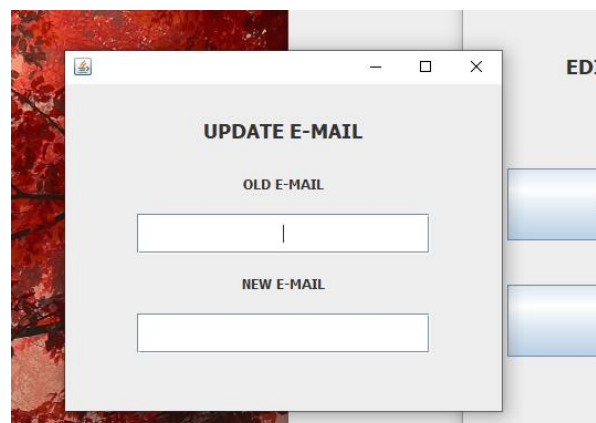
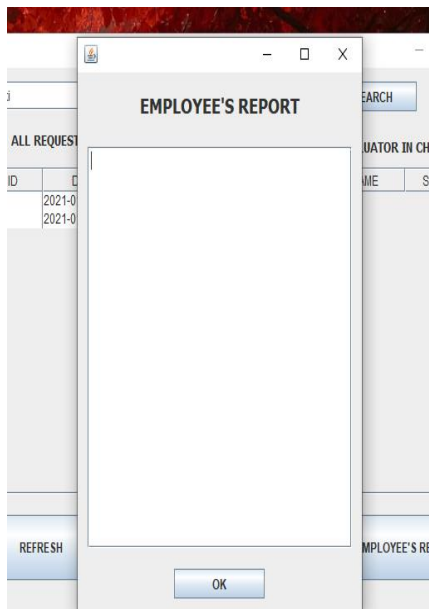
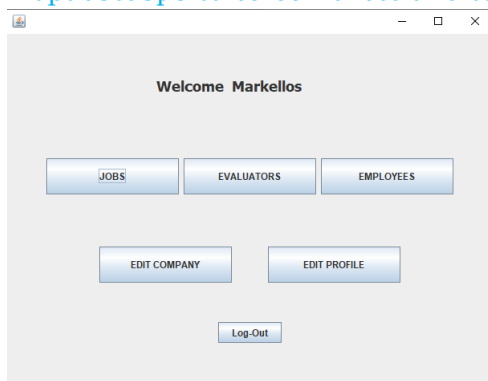


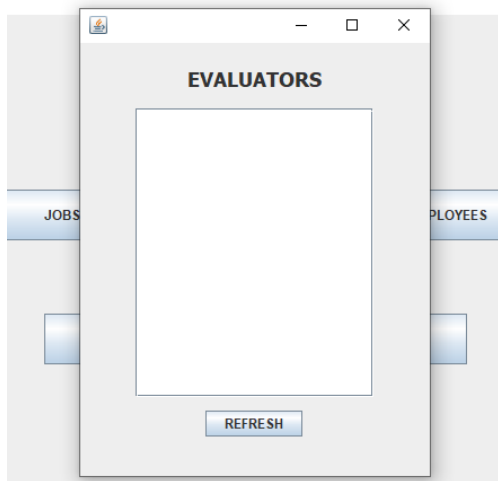
Av GUIPOINTER =5 (ERROR)



- ΤΟ ΚΟΥΜΠΙ LOG OUT ΑΠΟΛΟΥΣ ΤΟΥΣ USER ΓΥΡΙΖΕΙ ΣΤΟ SIGN IN (αρχική οθόνη).
- Ο κώδικας του manager JDBC είναι πολύ μεγάλος το έχουμε στα αρχεία του κώδικα.

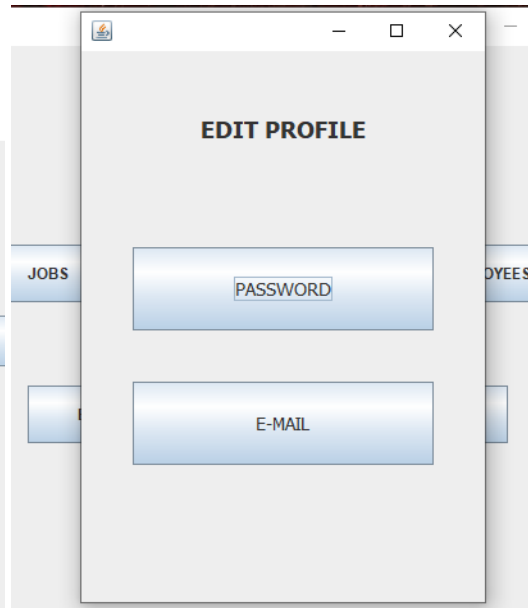
Παραθέτουμε τα screen-shots από το JDBC του manager για την λειτουργία του κώδικα:





EVALUATORS

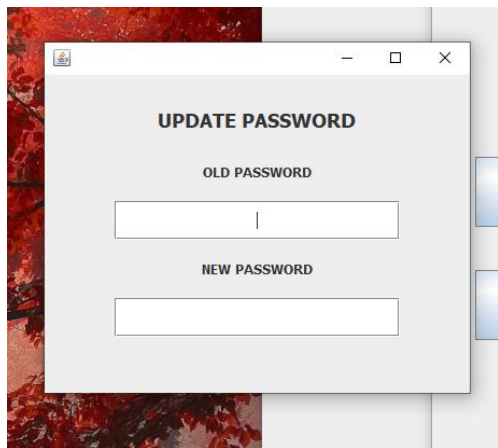
REFRESH



EDIT PROFILE

PASSWORD

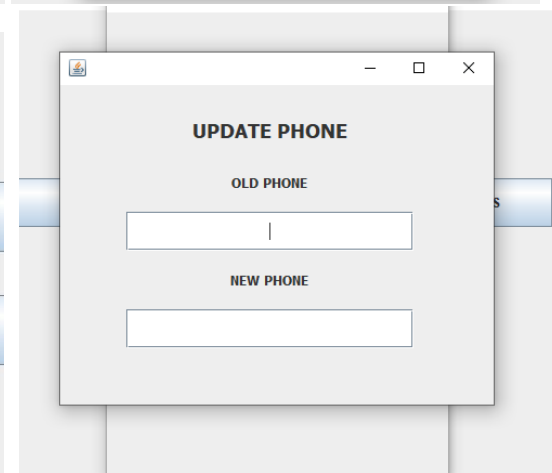
E-MAIL



UPDATE PASSWORD

OLD PASSWORD

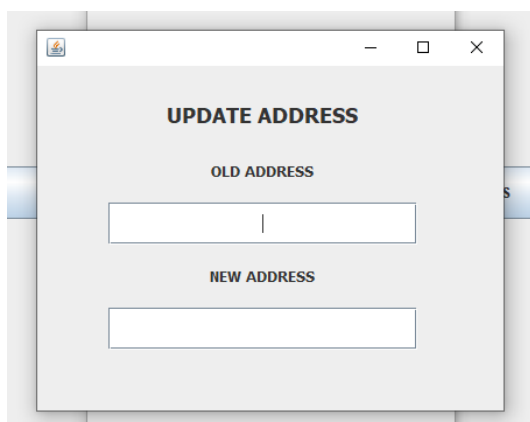
NEW PASSWORD



UPDATE PHONE

OLD PHONE

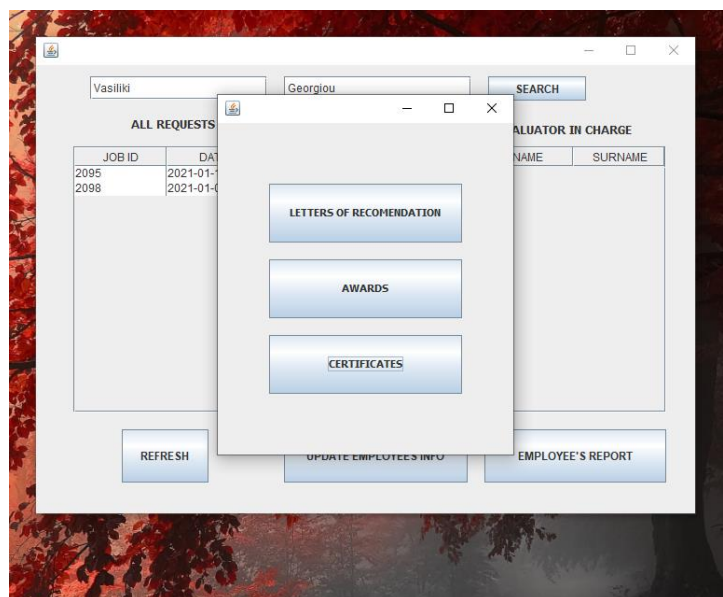
NEW PHONE



UPDATE ADDRESS

OLD ADDRESS

NEW ADDRESS



Vasiliki Georgiou SEARCH

ALL REQUESTS

JOB ID	DATE
2095	2021-01-7
2098	2021-01-4

REFRESH

LETTERS OF RECOMMENDATION

AWARDS

CERTIFICATES

EVALUATOR IN CHARGE

NAME	SURNAME
------	---------

UPDATE EMPLOYEES INFO

EMPLOYEE'S REPORT

Vasiliki

Georgiou

SEARCH

ALL REQUESTS

JOB ID	DATE
2095	2021-01-15
2098	2021-01-08

REFRESH

FINAL EVALUATIONS

POSITION	GRADE
----------	-------

UPDATE EMPLOYEES INFO

EVALUATOR IN CHARGE

NAME	SURNAME
------	---------

EMPLOYEE'S REPORT

FINAL EVALUATIONS

Name	Position	Evaluation Grade
basilikigeo	Assistant Manager	9
dimitriskous	Assistant Manager	10

REFRESH

PROMOTION POSITIONS

Position	Salary
Assistant Manager	1800.0
Secretary	1000.0
Assistant Advisor	1200.0
Advisor	1500.0
Advisor	1500.0

Assistant Manager

1800

EDIT

UPDATE

REFRESH

PROMOTION POSITIONS

Position	Salary
Assistant Manager	2100.0
Secretary	1000.0
Assistant Advisor	1200.0
Advisor	1500.0
Advisor	1500.0

POSITION

EDIT

UPDATE

REFRESH

PROMOTION POSITIONS

Position	Salary
Assistant Manager	2100.0
Secretary	1000.0
Assistant Advisor	1200.0
Advisor	1500.0
Advisor	1500.0

Assistant Manager

1800

EDIT

UPDATE

REFRESH