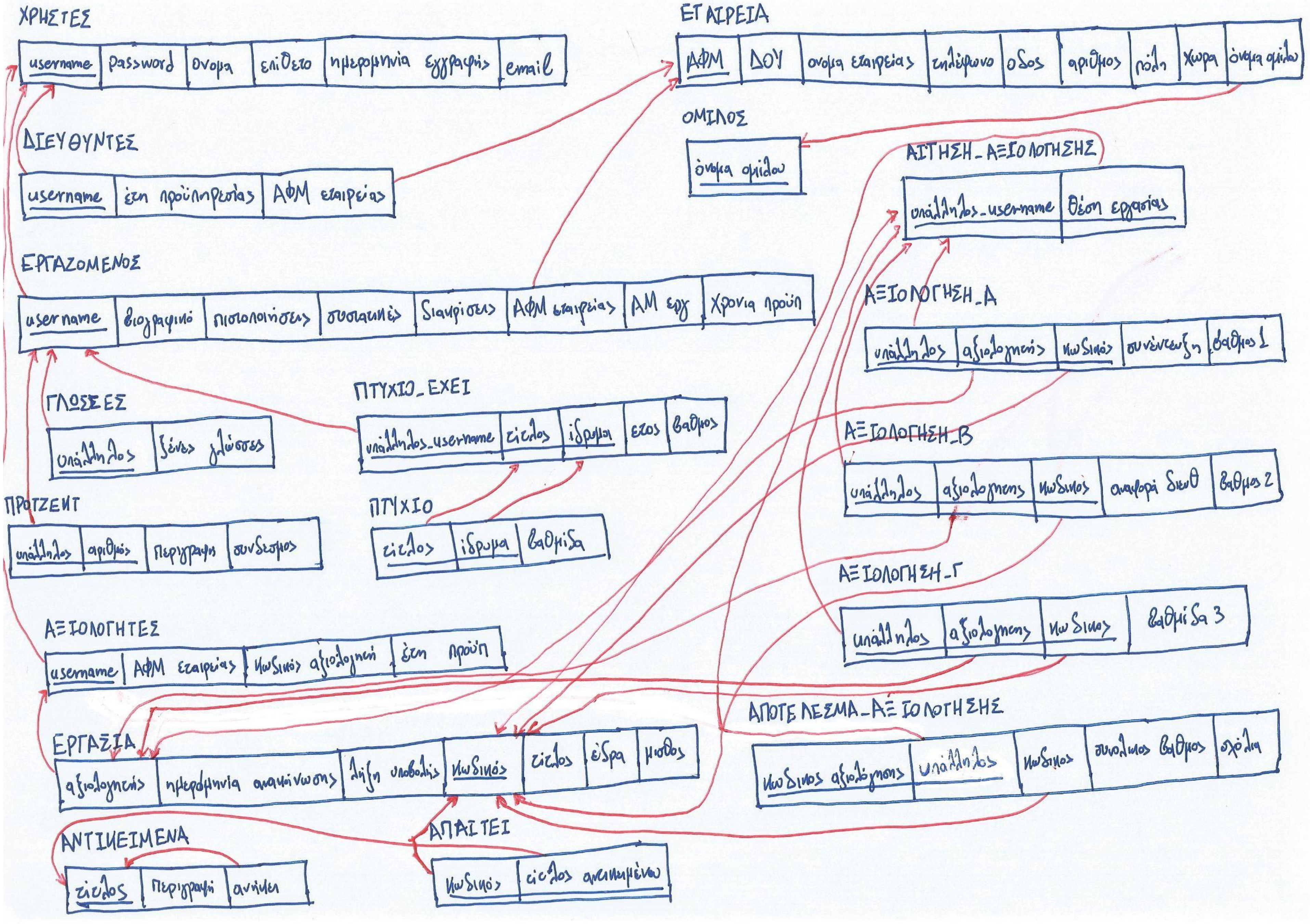
ΧΡΗΣΤΕΣ
username | password | ονομα | επιθετο | ημερομηνια εγγραφης | email
ΔΙΕΥΘΥΝΤΕΣ
username | έτη προϋπηρεσίας | ΑΦΜ εταιρείας
ΕΡΓΑΖΟΜΕΝΟΣ
username | βιογραφικό | πιστοποιήσεις | συστατικές | διακρίσεις | ΑΦΜ εταιρείας | ΑΜ εργ | Χρονια προϋπ
ΓΛΩΣΣΕΣ
υπάλληλος | ξένες γλώσσες
ΠΡΟΤΖΕΚΤ
υπάλληλος | αριθμός | περιγραφη | συνδεσμος
ΠΤΥΧΙΟ_ΕΧΕΙ
υπάλληλος_username | τίτλος | ίδρυμα | ετος | βαθμος
ΠΤΥΧΙΟ
τίτλος | ίδρυμα | βαθμίδα
ΑΞΙΟΛΟΓΗΤΕΣ
username | ΑΦΜ εταιρείας | κωδικός αξιολογητή | έτη προϋπ
ΕΡΓΑΣΙΑ
αξιολογητής | ημερομηνια ανακοινωσης | λήξη υποβολής | κωδικός | τίτλος | έδρα | μισθος
ΑΝΤΙΚΕΙΜΕΝΑ
τίτλος | περιγραφή | ανήκει
ΑΠΑΙΤΕΙ
κωδικός | τίτλος αντικειμένου
ΕΤΑΙΡΕΙΑ
ΑΦΜ | ΔΟΥ | ονομα εταιρείας | τηλέφωνο | οδος | αριθμος | πόλη | χωρα | όνομα ομίλου
ΟΜΙΛΟΣ
όνομα ομίλου
ΑΙΤΗΣΗ_ΑΞΙΟΛΟΓΗΣΗΣ
υπάλληλος_username | θέση εργασίας
ΑΞΙΟΛΟΓΗΣΗ_Α
υπάλληλος | αξιολογητής | κωδικός | συνέντευξη | βαθμος 1
ΑΞΙΟΛΟΓΗΣΗ_Β
υπάλληλος | αξιολογητης | κωδικός | αναφορά διευθ | βαθμος 2
ΑΞΙΟΛΟΓΗΣΗ_Γ
υπάλληλος | αξιολογητης | κωδικος | βαθμίδα 3
ΑΠΟΤΕΛΕΣΜΑ_ΑΞΙΟΛΟΓΗΣΗΣ
κωδικος αξιολόγησης | υπάλληλος | κωδικος | συνολικος βαθμος | σχόλια

```
DROP DATABASE project;

CREATE DATABASE project;

USE project;


CREATE TABLE user(

   username VARCHAR(12) NOT NULL,

   password VARCHAR(10) NOT NULL,

   name VARCHAR(25) NOT NULL,

   surname VARCHAR(35) NOT NULL,

   reg_date DATETIME DEFAULT NULL,

   email VARCHAR(30) DEFAULT NULL,

   PRIMARY KEY(username)

);


CREATE TABLE companies_group(

   name VARCHAR(15) NOT NULL,

   PRIMARY KEY(name)

);


CREATE TABLE company(

          afm CHAR(9) NOT NULL,

          doy VARCHAR(15) NOT NULL,

          name VARCHAR(15) NOT NULL,

          phone BIGINT(16) NOT NULL,

          street VARCHAR(15) NOT NULL,

   numbr TINYINT(4) NOT NULL,

          city VARCHAR(15) NOT NULL,

          country VARCHAR(15) NOT NULL,

          group_name VARCHAR(15) NOT NULL,

          PRIMARY KEY(afm,group_name),

          CONSTRAINT GROUP_NAM FOREIGN KEY(group_name) REFERENCES companies_group(name) ON DELETE
CASCADE ON UPDATE CASCADE
```

```
);


CREATE TABLE manager(
   man_username VARCHAR(12) NOT NULL,
   expyears TINYINT(4) NOT NULL DEFAULT 0,
   company_afm CHAR(9) NOT NULL,
   PRIMARY KEY(man_username),
   CONSTRAINT MAN_USER FOREIGN KEY(man_username) REFERENCES user(username) ON DELETE CASCADE ON
UPDATE CASCADE,
   CONSTRAINT C_AFM FOREIGN KEY(company_afm) REFERENCES company(afm) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE employee(
            e_username VARCHAR(12) NOT NULL,
            bio TEXT NOT NULL,
            certificates VARCHAR(50) NOT NULL,
            sistatikes VARCHAR(50) NOT NULL DEFAULT '',
            awards VARCHAR(50) NOT NULL DEFAULT '',
            c_afm CHAR(9) NULL,
            employee_am INT NOT NULL DEFAULT 0,
            years TINYINT(4) NOT NULL DEFAULT 0,
            PRIMARY KEY(e_username),
            CONSTRAINT EMP_USER FOREIGN KEY(e_username) REFERENCES user(username) ON DELETE CASCADE ON
UPDATE CASCADE,
            CONSTRAINT CO_AFM FOREIGN KEY(c_afm) REFERENCES company(afm) ON DELETE CASCADE ON UPDATE
CASCADE
);


CREATE TABLE languages(
   employe VARCHAR(12) NOT NULL,
   lang SET('EN','GER','GR') NOT NULL DEFAULT '',
   PRIMARY KEY(employe,lang),
   CONSTRAINT EM_USER FOREIGN KEY(employe) REFERENCES employee(e_username) ON DELETE CASCADE ON
UPDATE CASCADE
);


CREATE TABLE degree(
  titlos varchar(100) NOT NULL,
  idryma varchar(40) NOT NULL,
  bathmida ENUM ('LYKEIO','UNIV','MASTER','PHD') DEFAULT NULL,
  PRIMARY KEY (titlos,idryma)
);


CREATE TABLE has_degree(
  degr_title varchar(100) NOT NULL,
  degr_idryma varchar(40) NOT NULL,
  emp_username varchar(12) NOT NULL,
  etos year(4) DEFAULT NULL,
  grade float(3,1) DEFAULT NULL,
  PRIMARY KEY(emp_username,degr_title,degr_idryma),
  CONSTRAINT EMPL_USER FOREIGN KEY(emp_username) REFERENCES employee(e_username) ON DELETE CASCADE ON
UPDATE CASCADE,
  CONSTRAINT HAS_DGR FOREIGN KEY(degr_title,degr_idryma) REFERENCES degree(titlos,idryma) ON DELETE CASCADE
ON UPDATE CASCADE
);


CREATE TABLE project(
   empl_username VARCHAR(12) NOT NULL,
   num TINYINT(4) NOT NULL DEFAULT 0,
   descr TEXT NULL,
   url VARCHAR(60) NULL,
   PRIMARY KEY(empl_username,num),
   CONSTRAINT EMPUSER FOREIGN KEY(empl_username) REFERENCES employee(e_username) ON DELETE CASCADE ON
UPDATE CASCADE
);
```

```
CREATE TABLE evaluator(

    ev_username VARCHAR(12) NOT NULL,

    com_afm CHAR(9) NULL,

    ev_id INT NOT NULL DEFAULT 0,

    xp_years TINYINT(4) NOT NULL DEFAULT 0,

    PRIMARY KEY(ev_username),

    CONSTRAINT EV_USER FOREIGN KEY(ev_username) REFERENCES user(username) ON DELETE CASCADE ON UPDATE
CASCADE,

    CONSTRAINT COM_AFM FOREIGN KEY(com_afm) REFERENCES company(afm) ON DELETE CASCADE ON UPDATE
CASCADE

);


CREATE TABLE job(

    id INT(4) AUTO_INCREMENT,

    position VARCHAR(40) NOT NULL,

    edra VARCHAR(45) NOT NULL,

    salary FLOAT(6.1) NOT NULL,

    evaluator VARCHAR(12) NOT NULL,

    announce_date DATETIME DEFAULT NULL,

    submission_date DATE,

    PRIMARY KEY(id),

    CONSTRAINT EVA_USER FOREIGN KEY(evaluator) REFERENCES evaluator(ev_username) ON DELETE CASCADE ON
UPDATE CASCADE

);


CREATE TABLE antikeim(

    title VARCHAR(36) NOT NULL,

    descrb TINYTEXT NOT NULL,

    belongs_to VARCHAR(36),

    PRIMARY KEY(title),

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

CREATE TABLE needs(
    job_id INT(4) NOT NULL,
    title_antikeim VARCHAR(36) NOT NULL,
    PRIMARY KEY(job_id,title_antikeim),
    CONSTRAINT JOB_ID FOREIGN KEY(job_id) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT TTL_A FOREIGN KEY(title_antikeim) REFERENCES antikeim(title) ON DELETE CASCADE ON UPDATE
CASCADE
);

CREATE TABLE requestevaluation(
    employee_user VARCHAR(12) NOT NULL,
    jobid INT(4) NOT NULL,
    PRIMARY KEY(employee_user,jobid),
    CONSTRAINT EMPLOYEE_USER FOREIGN KEY(employee_user) REFERENCES employee(e_username) ON DELETE
CASCADE ON UPDATE CASCADE,
    CONSTRAINT JOBID FOREIGN KEY(jobid) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE evaluation_1(
    ypallhlos VARCHAR(12) NOT NULL,
    axiologhths VARCHAR(12) NOT NULL,
    kwdikos INT(4) NOT NULL,
    interview TEXT NULL,
    grade_1 ENUM('0','1','2','3','4') NULL,
    PRIMARY KEY(ypallhlos,axiologhths,kwdikos),
    CONSTRAINT YPAL1 FOREIGN KEY(ypallhlos) REFERENCES requestevaluation(employee_user) ON DELETE CASCADE ON
UPDATE CASCADE,
    CONSTRAINT AXIO1 FOREIGN KEY(axiologhths) REFERENCES job(evaluator) ON DELETE CASCADE ON UPDATE
CASCADE,
    CONSTRAINT KWD1 FOREIGN KEY(kwdikos) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE evaluation_2(

   ypallhlos_ev2 VARCHAR(12) NOT NULL,

   axiologhths_ev2 VARCHAR(12) NOT NULL,

   kwdikos_ev2 INT(4) NOT NULL,

   manager_report TEXT NULL,

   grade_2 ENUM('0','1','2','3','4') NULL,

   PRIMARY KEY(ypallhlos_ev2,axiologhths_ev2,kwdikos_ev2),

   CONSTRAINT YPAL2 FOREIGN KEY(ypallhlos_ev2) REFERENCES requestevaluation(employee_user) ON DELETE
CASCADE ON UPDATE CASCADE,

   CONSTRAINT AXIO2 FOREIGN KEY(axiologhths_ev2) REFERENCES job(evaluator) ON DELETE CASCADE ON UPDATE
CASCADE,

   CONSTRAINT KWD2 FOREIGN KEY(kwdikos_ev2) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE


);


CREATE TABLE evaluation_3(

   ypallhlos_ev3 VARCHAR(12) NOT NULL,

   axiologhths_ev3 VARCHAR(12) NOT NULL,

   kwdikos_ev3 INT(4) NOT NULL,

   grade_3 ENUM('0','1','2') NULL,

   PRIMARY KEY(ypallhlos_ev3,axiologhths_ev3,kwdikos_ev3),

   CONSTRAINT YPAL3 FOREIGN KEY(ypallhlos_ev3) REFERENCES requestevaluation(employee_user) ON DELETE
CASCADE ON UPDATE CASCADE,

   CONSTRAINT AXIO3 FOREIGN KEY(axiologhths_ev3) REFERENCES job(evaluator) ON DELETE CASCADE ON UPDATE
CASCADE,

   CONSTRAINT KWD3 FOREIGN KEY(kwdikos_ev3) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE

);


CREATE TABLE evaluationresult(

   evid INT(4) NULL AUTO_INCREMENT,

   em_username VARCHAR(12) NOT NULL,

   id_job INT(4) NOT NULL,
   total_grade INT(4) DEFAULT NULL,
   comments TEXT(255) DEFAULT NULL,
   PRIMARY KEY(evid,em_username),
   CONSTRAINT EMPL FOREIGN KEY(em_username) REFERENCES requestevaluation(employee_user) ON DELETE
CASCADE ON UPDATE CASCADE,
   CONSTRAINT JID FOREIGN KEY(id_job) REFERENCES job(id) ON DELETE CASCADE ON UPDATE CASCADE
);
```

--Αυτος ο trigger συμπληρωνει αυτοματα το id των υπαλληλων μιας εταιρειας παιρνοντας,καθε φορα που κανουμε insert εναν νεο υπαλληλο στην εταιραια,το αθροισμα των ηδη υπαρχοντων υπαλληλων και προσθετει +1 και το αποτελεσμα το αποθηκευει ως το νεο id του καινουριου υπαλληλου

```
DELIMITER $
DROP TRIGGER IF EXISTS employee_am$
CREATE TRIGGER employee_am
BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
DECLARE numofemployees INT;
SELECT COUNT(*) INTO numofemployees
FROM employee WHERE employee.c_afm=new.c_afm;
SET new.employee_am= numofemployees + 1;
END $
DELIMITER ;
```

--Ομοιως με τον employee_am αυτος ο trigger συμπληρωνει αυτοματα τον αριθμο των project ενος υπαλληλου

```
DELIMITER $
DROP TRIGGER IF EXISTS project_num$
CREATE TRIGGER project_num
BEFORE INSERT ON project
FOR EACH ROW
BEGIN
DECLARE numofprojects INT;
SELECT COUNT(*) INTO numofprojects
```

```
FROM project WHERE project.empl_username=new.empl_username;

SET new.num= numofprojects + 1;

END $

DELIMITER ;


--Ομοιως με τον employee_am αυτος ο trigger συμπληρωνει αυτοματα το id ενος αξιολογητη σε μια εταιρεια

DELIMITER $

DROP TRIGGER IF EXISTS eva_id$

CREATE TRIGGER eva_id

BEFORE INSERT ON evaluator

FOR EACH ROW

BEGIN

DECLARE numofevaluators INT;

SELECT COUNT(*) INTO numofevaluators

FROM evaluator WHERE evaluator.com_afm=new.com_afm;

SET new.ev_id= numofevaluators + 1;

END $

DELIMITER ;


--Αυτος ο trigger τοποθετει αρχικες τιμες στον πινακα evaluationresult μετα την εισαγωγη μιας αιτησης για αξιολογηση
απο εναν υπαλληλο

DELIMITER $

DROP TRIGGER IF EXISTS crt_evaluationresult$

CREATE TRIGGER crt_evaluationresult

AFTER INSERT ON requestevaluation

FOR EACH ROW

BEGIN

INSERT INTO evaluationresult(em_username,id_job)

VALUES (new.employee_user,new.jobid);

END$

DELIMITER ;
```

```
--Ομοιως με τον crt_evaluationresult αυτος ο trigger τοποθετει αρχικες τιμες στον evaluation_1
DELIMITER $
DROP TRIGGER IF EXISTS crt_evaluationresult1$
CREATE TRIGGER crt_evaluationresult1
AFTER INSERT ON requestevaluation
FOR EACH ROW
BEGIN
INSERT INTO evaluation_1(ypallhlos,axiologhths,kwdikos)
SELECT requestevaluation.employee_user,job.evaluator,job.id FROM requestevaluation
INNER JOIN job ON requestevaluation.jobid=job.id
WHERE new.jobid=job.id AND new.employee_user=requestevaluation.employee_user;
END$
DELIMITER ;

--Ομοιως με τον crt_evaluationresult αυτος ο trigger τοποθετει αρχικες τιμες στον evaluation_2
DELIMITER $
DROP TRIGGER IF EXISTS crt_evaluationresult2$
CREATE TRIGGER crt_evaluationresult2
AFTER INSERT ON requestevaluation
FOR EACH ROW
BEGIN
INSERT INTO evaluation_2(ypallhlos_ev2,axiologhths_ev2,kwdikos_ev2)
SELECT requestevaluation.employee_user,job.evaluator,job.id FROM requestevaluation
INNER JOIN job ON requestevaluation.jobid=job.id
WHERE new.jobid=job.id AND new.employee_user=requestevaluation.employee_user;
END$
DELIMITER ;

--Ομοιως με τον crt_evaluationresult αυτος ο trigger τοποθετει αρχικες τιμες στον evaluation_3
DELIMITER $
DROP TRIGGER IF EXISTS crt_evaluationresult3$
CREATE TRIGGER crt_evaluationresult3
```

```
AFTER INSERT ON requestevaluation

FOR EACH ROW

BEGIN

INSERT INTO evaluation_3(ypallhlos_ev3,axiologhths_ev3,kwdikos_ev3)

SELECT requestevaluation.employee_user,job.evaluator,job.id FROM requestevaluation

INNER JOIN job ON requestevaluation.jobid=job.id

WHERE new.jobid=job.id AND new.employee_user=requestevaluation.employee_user;

END$

DELIMITER ;


--Αυτος ο trigger συνδεει την συνεντευξη στην πρωτη αξιολογηση με τα σχολια στο αποτελεσμα της αξιολογησης,κατα την εισαγωγη μιας συνεντευξης γινεται ενημερωση στα σχολια στον πινακα evaluationresult

DELIMITER $

DROP TRIGGER IF EXISTS result_1$

CREATE TRIGGER result_1

AFTER INSERT ON evaluation_1

FOR EACH ROW

BEGIN

UPDATE evaluationresult

SET evaluationresult.comments=new.interview

WHERE evaluationresult.em_username=new.ypallhlos AND evaluationresult.id_job=new.kwdikos;

END$

DELIMITER ;


--Αυτος ο trigger ειναι παρομοιος με τον result_1,η μονη διαφορα ειναι οτι κατα την ενημερωση της στηλης interview στον evaluation_1 γινεται ενημερωση της στηλης comments στον evaluationresult

DELIMITER $

DROP TRIGGER IF EXISTS emp_interview$

CREATE TRIGGER emp_interview

AFTER UPDATE ON evaluation_1

FOR EACH ROW

BEGIN

UPDATE evaluationresult
```

```
SET evaluationresult.comments=new.interview
WHERE evaluationresult.em_username=old.ypallhlos AND evaluationresult.id_job=old.kwdikos;
END$
DELIMITER ;


--Αυτος ο trigger κατα την ενημερωση(δηλαδη εισαγωγη βαθμου και συνεντευξης αρα και ολοκληρωσης του) καλει την
store procedure του ερωτηματος 3.2 και ελεγχει αν και οι τρεις επιμερους αξιολογησεις ειναι ολοκληρωμενες(δηλαδη
εχουν βαθμο) και εφοσον αυτο ισχυει συμπληρωνει τον πινακα evaluationresult
DELIMITER $
DROP TRIGGER IF EXISTS check_evaluation1$
CREATE TRIGGER check_evaluation1
AFTER UPDATE ON evaluation_1
FOR EACH ROW
BEGIN
CALL check_evaluations(new.axiologhths,new.kwdikos);
END$
DELIMITER ;


--Ομοιως με τον check_evaluation1
DELIMITER $
DROP TRIGGER IF EXISTS check_evaluation2$
CREATE TRIGGER check_evaluation2
AFTER UPDATE ON evaluation_2
FOR EACH ROW
BEGIN
CALL check_evaluations(new.axiologhths_ev2,new.kwdikos_ev2);
END$
DELIMITER ;


--Ομοιως με τον check_evaluation1
DELIMITER $
DROP TRIGGER IF EXISTS check_evaluation3$
CREATE TRIGGER check_evaluation3
```

```
AFTER UPDATE ON evaluation_3

FOR EACH ROW

BEGIN

CALL check_evaluations(new.axiologhths_ev3,new.kwdikos_ev3);

END$

DELIMITER ;


--Εαν ενας υπαλληλος διαγραψει την αιτηση του για αξιολογηση αυτος ο trigger διαγραφει απο τον πινακα
evaluation_1 την γραμμη που εχει σχεση με την διεγραμμενη αιτηση για αξιολογηση

DELIMITER $

DROP TRIGGER IF EXISTS dlt_evaluationresult1$

CREATE TRIGGER dlt_evaluationresult1

AFTER DELETE ON requestevaluation

FOR EACH ROW

BEGIN

DELETE FROM evaluation_1

WHERE old.jobid=evaluation_1.kwdikos AND old.employee_user=evaluation_1.ypallhlos;

END$

DELIMITER ;


--Ομοιως με τον dlt_evaluationresult1 για τον πινακα evaluation_2

DELIMITER $

DROP TRIGGER IF EXISTS dlt_evaluationresult2$

CREATE TRIGGER dlt_evaluationresult2

AFTER DELETE ON requestevaluation

FOR EACH ROW

BEGIN

DELETE FROM evaluation_2

WHERE old.jobid=evaluation_2.kwdikos_ev2 AND old.employee_user=evaluation_2.ypallhlos_ev2;

END$

DELIMITER ;
```

```
--Ομοιως με τον dlt_evaluationresult1 για τον πινακα evaluation_3

DELIMITER $

DROP TRIGGER IF EXISTS dlt_evaluationresult3$

CREATE TRIGGER dlt_evaluationresult3

AFTER DELETE ON requestevaluation

FOR EACH ROW

BEGIN

DELETE FROM evaluation_3

WHERE old.jobid=evaluation_3.kwdikos_ev3 AND old.employee_user=evaluation_3.ypallhlos_ev3;

END$

DELIMITER ;


--Ομοιως με τον dlt_evaluationresult1 για τον πινακα evaluationresult

DELIMITER $

DROP TRIGGER IF EXISTS dlt_evaluationresult$

CREATE TRIGGER dlt_evaluationresult

AFTER DELETE ON requestevaluation

FOR EACH ROW

BEGIN

DELETE FROM evaluationresult

WHERE old.jobid=evaluationresult.id_job AND old.employee_user=evaluationresult.em_username;

END$

DELIMITER ;


--Αυτος ο trigger κατα την εισαγωγη νεου χρηστη στον πινακα user θετει στην ημερομηνια εγγραφης την τρεχουσα
ημερομηνια και ωρα

DELIMITER $

DROP TRIGGER IF EXISTS set_reg_date$

CREATE TRIGGER set_reg_date

BEFORE INSERT ON user

FOR EACH ROW

BEGIN
```

```
SET new.reg_date=CURRENT_TIMESTAMP;

END $

DELIMITER ;


--Αυτος ο trigger κατα την εισαγωγη μιας νεας θεσης εργασιας στον πινακα job θετει στην ημερομηνια ανακοινωσης
την τρεχουσα ημερομηνια και ωρα και ελεγχει εαν η ημερομηνια ληξης των αιτησεων ειναι μεταγενεστερη απο την
ημερομηνια ανακοινωσης

DELIMITER $

DROP TRIGGER IF EXISTS set_announce_date$

CREATE TRIGGER set_announce_date

BEFORE INSERT ON job

FOR EACH ROW

BEGIN

SET new.announce_date=CURRENT_TIMESTAMP;

IF DATEDIFF(new.submission_date,new.announce_date)<0 THEN

        SIGNAL SQLSTATE VALUE '45000' SET MESSAGE_TEXT = 'Wrong Submission Date';

END IF;

END$

DELIMITER ;


--Αυτος ο trigger ελεγχει αν τουλαχιστον μια απο τις επιμερους αξιολογησεις εχει ολοκληρωθει(δηλαδη εχει μπει
βαθμος) επομενως μπλοκαρει την προσπαθεια διαγραφης στον πινακα requestevaluation

DELIMITER $

DROP TRIGGER IF EXISTS dlt_requestevaluation$

CREATE TRIGGER dlt_requestevaluation

BEFORE DELETE ON requestevaluation

FOR EACH ROW

BEGIN

DECLARE bathmos_1 INT(4);

DECLARE bathmos_2 INT(4);

DECLARE bathmos_3 INT(4);

SELECT evaluation_1.grade_1 INTO bathmos_1 FROM evaluation_1

WHERE old.employee_user=evaluation_1.ypallhlos AND old.jobid=evaluation_1.kwdikos;
```

```
SELECT evaluation_2.grade_2 INTO bathmos_2 FROM evaluation_2
WHERE old.employee_user=evaluation_2.ypallhlos_ev2 AND old.jobid=evaluation_2.kwdikos_ev2;
SELECT evaluation_3.grade_3 INTO bathmos_3 FROM evaluation_3
WHERE old.employee_user=evaluation_3.ypallhlos_ev3 AND old.jobid=evaluation_3.kwdikos_ev3;
IF(bathmos_1 IS NOT NULL OR bathmos_2 IS NOT NULL OR bathmos_3 IS NOT NULL) THEN
	SIGNAL SQLSTATE VALUE '45000' SET MESSAGE_TEXT = 'Evaluation is active or finished';
END IF;
END$
DELIMITER ;

--Αυτος ο trigger ελεγχει εαν η ημερομηνια ληξης των αιτησεων ειναι μεταγενεστερη της τωρινης ημερομηνιας,εαν αυτο ισχυει μπλοκαρει την εισαγωγη νεας αιτησης αξιολογησης για την συγκεκριμενη θεση εργασιας
DELIMITER $
DROP TRIGGER IF EXISTS insrt_requestevaluation$
CREATE TRIGGER insrt_requestevaluation
BEFORE INSERT ON requestevaluation
FOR EACH ROW
BEGIN
DECLARE hmerominia DATE;
DECLARE telikh_hmerominia DATE;
SET hmerominia=CURRENT_TIMESTAMP;
SELECT job.submission_date INTO telikh_hmerominia FROM job
WHERE new.jobid=job.id;
IF DATEDIFF(telikh_hmerominia,hmerominia)<0 THEN
	SIGNAL SQLSTATE VALUE '45000' SET MESSAGE_TEXT = 'Submission date expired';
END IF;
END$
DELIMITER ;

--Αυτος ο trigger εμποδιζει την αλλαγη του αφμ,του δου,του ονοματος και του ονοματος του ομιλου μιας εταιρειας
DELIMITER $
DROP TRIGGER IF EXISTS update_company$
```

```
CREATE TRIGGER update_company

BEFORE UPDATE ON company

FOR EACH ROW

BEGIN

IF (old.afm!=new.afm OR old.doy!=new.doy OR old.name!=new.name OR old.group_name!=new.group_name ) THEN

        SET new.afm=old.afm;

        SET new.doy=old.doy;

        SET new.name=old.name;

        SET new.group_name=old.group_name;

END IF;

END$

DELIMITER ;


DELIMITER $

DROP PROCEDURE IF EXISTS AithshYpallhlou$

CREATE PROCEDURE AithshYpallhlou(IN onoma_ypallhlou VARCHAR(25),IN epi8eto_ypallhlou VARCHAR(35))

BEGIN

DECLARE onoma_xrhsth VARCHAR(12);

DECLARE idjob INT(4);

DECLARE idev INT(4);

DECLARE bathmos INT(4);

DECLARE sxolia VARCHAR(255);

DECLARE onoma_axiologhth VARCHAR(25);

DECLARE epi8eto_axiologhth VARCHAR(35);

DECLARE bathmos_1 INT(4);

DECLARE bathmos_2 INT(4);

DECLARE bathmos_3 INT(4);

DECLARE not_found INT;

/*χρησημοποιουμε cursor εδω γιατι ενας υπαλληλος μπορει να εχει κανει για παραπανω απο μια θεση εργασιας
αιτηση για αξιολογηση*/

DECLARE ecursor CURSOR FOR
```

```
SELECT
requestevaluation.jobid,evaluationresult.evid,evaluationresult.total_grade,evaluationresult.comments,user.name,user.s
urname FROM requestevaluation

INNER JOIN evaluationresult ON requestevaluation.jobid=evaluationresult.id_job

INNER JOIN job ON evaluationresult.id_job=job.id

INNER JOIN user ON job.evaluator=user.username

WHERE onoma_xrhsth=requestevaluation.employee_user;

/*η χρηση του cursor εδω περα ειναι για να βρουμε το username του υπαλληλου απο το ονομα και το επιθετο που
καλουμε με την procedure μας*/

DECLARE ucursor CURSOR FOR

SELECT user.username FROM user

INNER JOIN employee ON employee.e_username=user.username

WHERE user.name=onoma_ypallhlou AND user.surname=epi8eto_ypallhlou;

/*χρησημοποιουμε cursor εδω διοτι μπορει ενας υπαλληλος να εχει περισσοτερες απο μια αξιολογησεις*/

DECLARE bcursor CURSOR FOR

SELECT evaluation_1.grade_1,evaluation_2.grade_2, evaluation_3.grade_3 FROM evaluation_1

INNER JOIN evaluation_2 ON evaluation_1.ypallhlos=evaluation_2.ypallhlos_ev2 AND
evaluation_1.kwdikos=evaluation_2.kwdikos_ev2

INNER JOIN evaluation_3 ON evaluation_2.ypallhlos_ev2=evaluation_3.ypallhlos_ev3 AND
evaluation_2.kwdikos_ev2=evaluation_3.kwdikos_ev3

WHERE evaluation_1.ypallhlos=onoma_xrhsth AND evaluation_1.kwdikos=idjob;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET not_found=1;

OPEN ucursor;

SET not_found=0;

/*με την εντολη fetch βαζουμε την τιμη που εχει ο cursor στο onoma_xrhsth*/

FETCH ucursor INTO onoma_xrhsth;

/*αν δεν υπαρχει τιμη συμαινει πως το ονομα η το επιθετο που βαλαμε οταν καλεσαμε την procedure ειναι
λανθασμενο η δεν αντιστοιχει σε υπαλληλο*/

IF (not_found=1) THEN

   SELECT 'This user is not an employee';

END IF;

/*ν υπαρχσει τιμη στον cursor τοτε μπαινουμε στην εντολη while*/

WHILE (not_found=0) DO

   OPEN ecursor;
```

```
SET not_found=0;
/*με την εντολη fetch βαζουμε τιμες στον ecursor*/
FETCH ecursor INTO idjob,idev,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
/*αν δεν υπαρχουν τιμες συμαινει οτι συγκεκριμενος υπαλληλος δεν εχει κανει αιτηση για να αξιολογηθει*/
IF (not_found=1) THEN
   SELECT 'Employee has no requested an evaluation';
END IF;
/*αν υπαρχουν τιμες τοτε μπαινουμε στην εντολη while*/
WHILE (not_found=0) DO
   /*εαν η αξιολογηση εχει ολοκληρωθει ο τελικος βαθμος δεν πρεπει να ειναι null*/
   IF (bathmos IS NOT NULL) THEN
      SELECT idev,idjob,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
   ELSE
      SELECT 'Incomplete Total Evaluation',idev as kwdikos_axiologhshs;
      OPEN bcursor;
      SET not_found=0;
      /*σε αυτο το fetch βαζουμε τις τιμες των βαθμων απο τις 3 επιμερους αξιολογησεις*/
      FETCH bcursor INTO bathmos_1,bathmos_2,bathmos_3;
      IF (not_found=0) THEN
            /*εδω εξεταζουμε ποια απο τις 3 αξιολογησεις δεν ειναι ολοκληρωμενη*/
            IF (bathmos_1 IS NULL) THEN
               SELECT 'Incomplete Evaluation 1';
            ELSE
               SELECT * FROM evaluation_1 WHERE evaluation_1.kwdikos=idjob;
            END IF;
            IF (bathmos_2 IS NULL) THEN
               SELECT 'Incomplete Evaluation 2';
            ELSE
               SELECT * FROM evaluation_2 WHERE evaluation_2.kwdikos_ev2=idjob;
            END IF;
            IF (bathmos_3 IS NULL) THEN
               SELECT 'Incomplete Evaluation 3';
```

```
                 ELSE
                    SELECT * FROM evaluation_3 WHERE evaluation_3.kwdikos_ev3=idjob;
                 END IF;
             END IF;
             CLOSE bcursor;
         END IF;
     FETCH ecursor INTO idjob,idev,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
     END WHILE;
     CLOSE ecursor;
     FETCH ucursor INTO onoma_xrhsth;
END WHILE;
CLOSE ucursor;
END$
DELIMITER ;


DELIMITER $
DROP PROCEDURE IF EXISTS check_evaluations$
CREATE PROCEDURE check_evaluations(IN axiologhths VARCHAR(12),IN thesh INT(4))
BEGIN
DECLARE evaluator_username VARCHAR(12);
DECLARE employee_username VARCHAR(12);
DECLARE idjob INT(4);
DECLARE bathmos INT(4);
DECLARE bathmos_1 INT(4);
DECLARE bathmos_2 INT(4);
DECLARE bathmos_3 INT(4);
DECLARE not_found INT;
/*χρησημοποιουμε cursor εδω γιατι ενας αξιολογητης μπορει να εχει περισσοτερες απο μια ενεργες αξιολογησεις για
μια θεση εργασιας*/
DECLARE ecursor CURSOR FOR
SELECT
evaluation_1.axiologhths,evaluation_1.kwdikos,evaluation_1.ypallhlos,evaluation_1.grade_1,evaluation_2.grade_2,eval
uation_3.grade_3 FROM evaluation_1
```

```
INNER JOIN evaluation_2 ON evaluation_1.kwdikos=evaluation_2.kwdikos_ev2 AND
evaluation_1.ypallhlos=evaluation_2.ypallhlos_ev2

INNER JOIN evaluation_3 ON evaluation_2.kwdikos_ev2=evaluation_3.kwdikos_ev3 AND
evaluation_2.ypallhlos_ev2=evaluation_3.ypallhlos_ev3

WHERE evaluation_1.axiologhths=axiologhths AND evaluation_1.kwdikos=thesh;

DECLARE gcursor CURSOR FOR

SELECT evaluationresult.total_grade FROM evaluationresult

INNER JOIN job ON evaluationresult.id_job=job.id

WHERE evaluationresult.id_job=thesh AND job.evaluator=axiologhths AND
evaluationresult.em_username=employee_username;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET not_found=1;

OPEN ecursor;

SET not_found=0;

/*στην fetch εδω βαζουμε τις τιμες του cursor μας*/

FETCH ecursor INTO evaluator_username,idjob,employee_username,bathmos_1,bathmos_2,bathmos_3;

/*Οσο υπαρχουν τιμες θα μπαινουμε στην εντολη while*/

WHILE (not_found=0) DO

   /*εαν ολοι οι βαθμοι απο τις επιμερους αξιολογησεις ειναι ολοκληρωμενοι δηλαδη δεν ειναι null μπαινουμε στην
εντολη if*/

   IF (bathmos_1 IS NOT NULL AND bathmos_2 IS NOT NULL AND bathmos_3 IS NOT NULL) THEN

      OPEN gcursor;

      SET not_found=0;

      FETCH gcursor INTO bathmos;

      IF (not_found=0) THEN

         /*εαν ο συνολικος βαθμος ειναι null τοτε προσθετουμε τους 3 επιμερους βαθμους και το αποτελεσμα της
αθροισης το τοποθετουμε στο συνολικο βαθμο στον πινακα evaluationresult*/

         IF (bathmos IS NULL) THEN

            SET bathmos=bathmos_1+bathmos_2+bathmos_3;

            UPDATE evaluationresult

            SET total_grade=bathmos

            WHERE id_job=thesh AND em_username=employee_username;

         END IF;

      END IF;

   CLOSE gcursor;
```

```
    END IF;

    FETCH ecursor INTO evaluator_username,idjob,employee_username,bathmos_1,bathmos_2,bathmos_3;

END WHILE;

CLOSE ecursor;

END$

DELIMITER ;


DELIMITER $

DROP PROCEDURE IF EXISTS complete_evaluations$

CREATE PROCEDURE complete_evaluations(IN thesh INT(4))

BEGIN

DECLARE numofrequestevaluations INT(4);

DECLARE numofcompleteevaluations INT(4);

DECLARE numofnullgrade INT(4);

DECLARE bathmos INT(4);

DECLARE not_found INT;

/*με αυτον τον cursor παιρνουμε τον συνολικο βαθμο απο καθε αξιολογηση για μια συγκεκριμενη θεση εργασιας */

DECLARE gcursor CURSOR FOR

SELECT evaluationresult.total_grade FROM evaluationresult

WHERE evaluationresult.id_job=thesh;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET not_found=1;

/*εδω παιρνουμε το συνλικο αριθμο των αιτησεων που εχουν γινει για μια θεση εργασιας*/

SELECT COUNT(*) INTO numofrequestevaluations FROM requestevaluation WHERE requestevaluation.jobid=thesh;

/*εδω παιρνουμε το συνολικο αριθμο των αιτησεων που εχουν ολοκληρωθει για μια θεση εργασιας δηλαδη ο τελικος
βαθμος στον πινακα evaluationresult δεν ειναι null*/

SELECT COUNT(*) INTO numofcompleteevaluations FROM evaluationresult WHERE evaluationresult.id_job=thesh AND
evaluationresult.total_grade IS NOT NULL;

IF (numofrequestevaluations=0) THEN

    SELECT 'No request evaluations for this id';

ELSE

    /*αν το αθροισμα των συνολικων αιτησεων ειναι ισο με το αθροισμα των ολοκληρωμενων αιτησεων τοτε
εμφανιζεται ποσες ειναι οι ολοκληρωμενες αιτησεις οπως επισης και ολοι οι υποψηφιοι με τις τελικες τους
αξιολογησεις κατα φθινουσα σειρα */
```

```
   IF (numofcompleteevaluations=numofrequestevaluations) THEN

      SELECT numofcompleteevaluations AS 'Oristikopoihmenoi Pinakes';

      SELECT * FROM evaluationresult WHERE id_job=thesh ORDER BY total_grade DESC;

   ELSE

      /*αν το αθροισμα των συνολικων αιτησεων δεν ειναι ισο με το αθροισμα των ολοκληρωμενων αιτησεων με αυτον
τον cursor ελεγχουμε ποσες αξιολογησεις εκρεμουν και εμφανιζουμε τις ολοκληρωμενες αιτησεις με ολους τους
υποψηφιους με τις τελικες τους αξιολογησεις κατα φθινουσα σειρα */

      OPEN gcursor;

      SET not_found=0;

      FETCH gcursor INTO bathmos;

      SET numofnullgrade=0;

      WHILE (not_found=0) DO

            IF (bathmos IS NOT NULL) THEN

               SELECT * FROM evaluationresult WHERE id_job=thesh AND total_grade=bathmos ORDER BY total_grade
DESC;

            ELSE

                  SET numofnullgrade=numofnullgrade + 1;

            END IF;

      FETCH gcursor INTO bathmos;

      END WHILE;

      CLOSE gcursor;

      SELECT 'Axiologhsh se Exelixh',numofnullgrade as Ekremoun;

   END IF;

END IF;

END $

DELIMITER ;


/*Παρομοια με την procedure AithshYpallhlou η μονη διαφορα ειναι οτι κοιταμε τους υπαλληλους μια εταιρειας και
αυτο το πετυχαινουμε με την εισαγωγη του αφμ της εταιρειας οταν καλουμε την procedure μας */

DELIMITER $

DROP PROCEDURE IF EXISTS AithshYpallhlou1$

CREATE PROCEDURE AithshYpallhlou1(IN onoma_ypallhlou VARCHAR(25),IN epi8eto_ypallhlou VARCHAR(35),IN
afm_etaireias CHAR(9))

BEGIN
```

```
DECLARE onoma_xrhsth VARCHAR(12);

DECLARE idjob INT(4);

DECLARE idev INT(4);

DECLARE bathmos INT(4);

DECLARE sxolia VARCHAR(255);

DECLARE onoma_axiologhth VARCHAR(25);

DECLARE epi8eto_axiologhth VARCHAR(35);

DECLARE bathmos_1 INT(4);

DECLARE bathmos_2 INT(4);

DECLARE bathmos_3 INT(4);

DECLARE not_found INT;

DECLARE ecursor CURSOR FOR

SELECT
requestevaluation.jobid,evaluationresult.evid,evaluationresult.total_grade,evaluationresult.comments,user.name,user.s
urname FROM requestevaluation

INNER JOIN evaluationresult ON requestevaluation.jobid=evaluationresult.id_job

INNER JOIN job ON evaluationresult.id_job=job.id

INNER JOIN user ON job.evaluator=user.username

WHERE onoma_xrhsth=requestevaluation.employee_user;

DECLARE ucursor CURSOR FOR

SELECT user.username FROM user

INNER JOIN employee ON employee.e_username=user.username

INNER JOIN company ON employee.c_afm=company.afm

INNER JOIN manager ON company.afm=manager.company_afm

WHERE user.name=onoma_ypallhlou AND user.surname=epi8eto_ypallhlou AND company.afm=afm_etaireias;

DECLARE bcursor CURSOR FOR

SELECT evaluation_1.grade_1,evaluation_2.grade_2, evaluation_3.grade_3 FROM evaluation_1

INNER JOIN evaluation_2 ON evaluation_1.ypallhlos=evaluation_2.ypallhlos_ev2 AND
evaluation_1.kwdikos=evaluation_2.kwdikos_ev2

INNER JOIN evaluation_3 ON evaluation_2.ypallhlos_ev2=evaluation_3.ypallhlos_ev3 AND
evaluation_2.kwdikos_ev2=evaluation_3.kwdikos_ev3

WHERE evaluation_1.ypallhlos=onoma_xrhsth AND evaluation_1.kwdikos=idjob;

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

```
OPEN ucursor;
SET not_found=0;
FETCH ucursor INTO onoma_xrhsth;
IF (not_found=1) THEN
  SELECT 'This user is not an employee';
END IF;
WHILE (not_found=0) DO
  OPEN ecursor;
  SET not_found=0;
  FETCH ecursor INTO idjob,idev,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
  IF (not_found=1) THEN
    SELECT 'Employee has no requested an evaluation';
  END IF;
  WHILE (not_found=0) DO
    IF (bathmos IS NOT NULL) THEN
      SELECT idev,idjob,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
    ELSE
      SELECT 'Incomplete Total Evaluation',idev as kwdikos_axiologhshs;
      OPEN bcursor;
      SET not_found=0;
      FETCH bcursor INTO bathmos_1,bathmos_2,bathmos_3;
      IF (not_found=0) THEN
          IF (bathmos_1 IS NULL) THEN
            SELECT 'Incomplete Evaluation 1';
          ELSE
            SELECT * FROM evaluation_1 WHERE evaluation_1.kwdikos=idjob;
          END IF;
          IF (bathmos_2 IS NULL) THEN
            SELECT 'Incomplete Evaluation 2';
          ELSE
            SELECT * FROM evaluation_2 WHERE evaluation_2.kwdikos_ev2=idjob;
          END IF;
```

```
                IF (bathmos_3 IS NULL) THEN
                   SELECT 'Incomplete Evaluation 3';
                ELSE
                   SELECT * FROM evaluation_3 WHERE evaluation_3.kwdikos_ev3=idjob;
                END IF;
          END IF;
           CLOSE bcursor;
       END IF;
    FETCH ecursor INTO idjob,idev,bathmos,sxolia,onoma_axiologhth,epi8eto_axiologhth;
    END WHILE;
    CLOSE ecursor;
    FETCH ucursor INTO onoma_xrhsth;
END WHILE;
CLOSE ucursor;
END$
DELIMITER ;

/*με αυτην την procedure εμφανιζουμε ολες τις θεσεις εργασιας της εταιρειας του υπαλληλου που μπορει να ζητησει
να αξιολογηθει*/
DELIMITER $
DROP PROCEDURE IF EXISTS select_jobs$
CREATE PROCEDURE select_jobs(IN onoma_xrhsth VARCHAR(12))
BEGIN
SELECT job.id,job.position,job.edra,job.salary,job.evaluator,job.announce_date,job.submission_date FROM job
INNER JOIN evaluator ON job.evaluator=evaluator.ev_username
INNER JOIN employee ON evaluator.com_afm=employee.c_afm
WHERE employee.e_username=onoma_xrhsth;
END $
DELIMITER ;

/*με αυτην την procedure εμφανιζουμε τις πληροφοριες ενος υπαλληλου*/
DELIMITER $
```

```
DROP PROCEDURE IF EXISTS employee_informations$

CREATE PROCEDURE employee_informations(IN onoma_xrhsth VARCHAR(12))

BEGIN

SELECT employee.bio AS Biografiko,employee.certificates AS Certificates,employee.sistatikes AS
Sistatikes,employee.awards AS Awards,employee.c_afm AS AFM_Etaireias,

employee.employee_am AS AM,employee.years AS Xronia_Empeirias,languages.lang AS Glwsses,has_degree.degr_title
AS Titlos,has_degree.degr_idryma AS Idryma,

degree.bathmida AS Bathmida,has_degree.etos AS Xronologia,has_degree.grade AS Bathmos FROM employee

INNER JOIN languages ON employee.e_username=languages.employe

INNER JOIN has_degree ON employee.e_username=has_degree.emp_username

INNER JOIN degree ON has_degree.degr_title=degree.titlos AND has_degree.degr_idryma=degree.idryma

WHERE employee.e_username=onoma_xrhsth;

END$

DELIMITER ;


INSERT INTO user VALUES

('kostasd1','12345','Kostas','Dimitriou','2021-01-01 00:00:00','kostasd1@gmail.com'),

('panosgee','12345','Panos','Gialamas','2021-01-01 00:00:00','panosgee@gmail.com'),

('ipay67','12345','Giannis','Georgiou','2021-01-01 00:00:00','ipay67@gmail.com'),

('cakemaker','12345','Dimitra','Theodosiou','2021-01-01 00:00:00','cakemaker@gmail.com'),

('love1940','12345','Vaya','Kazatzidi','2021-01-01 00:00:00','love1940@gmail.com'),

('lamialover','12345','Aleka','Amanatidou','2021-01-01 00:00:00','lamialover@gmail.com'),

('doubled5','12345','Dimitris','Dimakopoulos','2021-01-01 00:00:00','doubled5@gmail.com'),

('dance367','12345','Mariloy','Eleftheriou','2021-01-01 00:00:00','dance367@gmail.com'),

('justice194','12345','Agamemnonas','Kexagias','2021-01-01 00:00:00','justice194@gmail.com'),

('cook178','12345','Ektoras','Mporini','2021-01-01 00:00:00','cook178@gmail.com'),

('kleftis91','12345','Paris','Iliadis','2021-01-01 00:00:00','kleftis91@gmail.com'),

('fterna32','12345','Axileas','Zografou','2021-01-01 00:00:00','fterna32@gmail.com'),

('dman','12345','Patroklos','Blaxopoulos','2021-01-01 00:00:00','dman@gmail.com'),

('super3','12345','Ifigenia','Bikelidou','2021-01-01 00:00:00','super3@gmail.com'),

('alone21','12345','Pinelopi','Babaka','2021-01-01 00:00:00','alone21@gmail.com'),

('lostboat','12345','Odiseas','Alitis','2021-01-01 00:00:00','lostboat@gmail.com'),

('detective','12345','Chloe','Decker','2021-01-01 00:00:00','detective@gmail.com'),
('jason24','12345','Jason','Detrulo','2021-01-01 00:00:00','jason24@gmail.com'),
('smallkid4','12345','Tracy','Decker','2021-01-01 00:00:00','smallkid4@gmail.com'),
('shorthair','12345','Dean','Winchester','2021-01-01 00:00:00','shorthair@gmail.com'),
('longhair','12345','Sam','Winchester','2021-01-01 00:00:00','longhair@gmail.com'),
('bd83','12345','Bobby','Dillan','2021-01-01 00:00:00','bd83@gmail.com'),
('audioj','12345','Jack','Willian','2021-01-01 00:00:00','audioj@gmail.com'),
('darkqueen','12345','Amara','Elarabi','2021-01-01 00:00:00','darkqueen@gmail.com'),
('angel123','12345','Castiel','Novak','2021-01-01 00:00:00','angel123@gmail.com'),
('cing99','12345','Crowley','Dickinson','2021-01-01 00:00:00','cing99@gmail.com'),
('codylegend','12345','Cody','Simpson','2021-01-01 00:00:00','codylegend@gmail.com'),
('sakisas','12345','Sakis','Georgoulas','2021-01-01 00:00:00','sakisas@gmail.com');

INSERT INTO companies_group VALUES
('Misk Productions');



INSERT INTO company VALUES
('197747737','Naousas','Mania Records','6982030794','Makedonias','4','Naousa','Ellada','Misk Productions'),
('943268720','Lamias','Fokos Records','6937425887','Kinou','79','Lamia','Ellada','Misk Productions'),
('126969074','Patras','Dimos Records','6940000123','Pinokio','7','Patra','Ellada','Misk Productions'),
('191797571','Filipiadas','BDCF Industry','6987276312','Hrwwn','6','Filipiada','Ellada','Misk Productions');



INSERT INTO manager VALUES
('longhair','3','197747737'),
('ipay67','15','943268720'),
('smallkid4','1','126969074'),
('bd83','6','191797571');

INSERT INTO employee VALUES
('kostasd1','19 xronwn','3+ years experience using digital audio hardware and software','','','197747737','','3'),
('panosgee','32 xronwn','Diploma Level 3 Music Production','','MAD Awards','943268720','','7'),
('cakemaker','44 xronwn','Bachelors degree in science of music','','Grammy','126969074','','20'),
('love1940','25 xronwn','SAE Certificate in Electronic Music Production','','Academy Award','191797571','','5'),
('lamialover','54 xronwn','Bachelors degree in science of music','','','197747737','','24'),
('doubled5','56 xronwn','Bachelors degree in science of music','','','943268720','','26'),
('dance367','22 xronwn','3+ years experience using digital audio hardware and software','','Academy
Award','126969074','','4'),
('cook178','24 xronwn','SAE Certificate in Electronic Music Production','','International Electronic Music
Award','191797571','','8'),
('kleftis91','36 xronwn','Diploma Level 3 Music Production','','Academy Award','197747737','','12'),
('fterna32','31 xronwn','Diploma Level 3 Music Production','','','943268720','','5'),
('dman','27 xronwn','SAE Certificate in Electronic Music Production','','Grammy','126969074','','12'),
('super3','23 xronwn','SAE Certificate in Electronic Music Production','','Mad Awards','191797571','','5'),
('shorthair','30 xronwn','Diploma Level 3 Music Production','','','197747737','','4'),
('cing99','49 xronwn','Bachelors degree in science of music','','International Classical Music Award','943268720','','29'),
('codylegend','34 xronwn','Diploma Level 3 Music Production','','','126969074','','6'),
('sakisas','20 xronwn','3+ years experience using digital audio hardware and software','','','191797571','','3');


INSERT INTO languages VALUES
('kostasd1','GR'),
('panosgee','GR,EN'),
('cakemaker','GR,EN'),
('love1940','GR,EN,GER'),
('lamialover','GR'),
('doubled5','GR'),
('dance367','GR'),
('cook178','GR,GER'),
('kleftis91','EN,GR'),
('fterna32','GR'),
('dman','GR,GER'),
('super3','GR,GER'),
('shorthair','EN'),
('cing99','EN,GER'),
('codylegend','EN'),
('sakisas','GR');

INSERT INTO degree VALUES
('Faculty of Music & Audiovisual Arts,Department of Music Studies','Ionian University','Univ'),
('Department of Music Science and Arts','University of Macedonia','MASTER'),
('College of Music','Berklee College','PHD'),
('Musical School of Athens','Musical School','LYKEIO');

INSERT INTO has_degree VALUES
('Musical School of Athens','Musical School','kostasd1','2020','17,3'),
('Department of Music Science and Arts','University of Macedonia','panosgee','2014','8,1'),
('College of Music','Berklee College','cakemaker','2007','9,4'),
('Faculty of Music & Audiovisual Arts,Department of Music Studies','Ionian University','love1940','2020','7'),
('College of Music','Berklee College','lamialover','2009','7,6'),
('College of Music','Berklee College','doubled5','2011','7,3'),
('Musical School of Athens','Musical School','dance367','2017','19'),
('Faculty of Music & Audiovisual Arts,Department of Music Studies','Ionian University','cook178','2020','6,9'),
('Department of Music Science and Arts','University of Macedonia','kleftis91','2012','6,4'),
('Department of Music Science and Arts','University of Macedonia','fterna32','2017','7,5'),
('Faculty of Music & Audiovisual Arts,Department of Music Studies','Ionian University','dman','2016','9,6'),
('Faculty of Music & Audiovisual Arts,Department of Music Studies','Ionian University','super3','2021','6,7'),
('Department of Music Science and Arts','University of Macedonia','shorthair','2017','6,8'),
('College of Music','Berklee College','cing99','2008','8,2'),
('Department of Music Science and Arts','University of Macedonia','codylegend','2016','5,9'),
('Musical School of Athens','Musical School','sakisas','2019','12,4');

INSERT INTO evaluator VALUES
('justice194','197747737','','2'),
('alone21','197747737','','3'),
('lostboat','943268720','','1'),
('detective','943268720','','2'),
('jason24','126969074','','5'),

('angel123','126969074','','3'),

('audioj','191797571','','2'),

('darkqueen','191797571','','1');



INSERT INTO job VALUES

('','BeatMaker','Naousa','1200.0','justice194','2021-01-01 00:00:00','2021-03-01'),

('','Producer','Naousa','1600.0','alone21','2021-01-01 00:00:00','2021-03-01'),

('','BeatMaker','Lamia','1300.0','lostboat','2021-01-01 00:00:00','2021-03-01'),

('','Producer','Lamia','1500.0','detective','2021-01-01 00:00:00','2021-03-01'),

('','BeatMaker','Patra','1250.0','jason24','2021-01-01 00:00:00','2021-03-01'),

('','Producer','Patra','1500.0','angel123','2021-01-01 00:00:00','2021-03-01'),

('','BeatMaker','Filipiada','1300.0','audioj','2021-01-01 00:00:00','2021-03-01'),

('','Producer','Filipiada','1650.0','darkqueen','2021-01-01 00:00:00','2021-03-01');


INSERT INTO antikeim VALUES

('Loops','Create loops',NULL),

('Midi,Speakers,DAW','Makes your life easier','Loops');


INSERT INTO needs VALUES

('2','Midi,Speakers,DAW'),

('7','Loops');
```