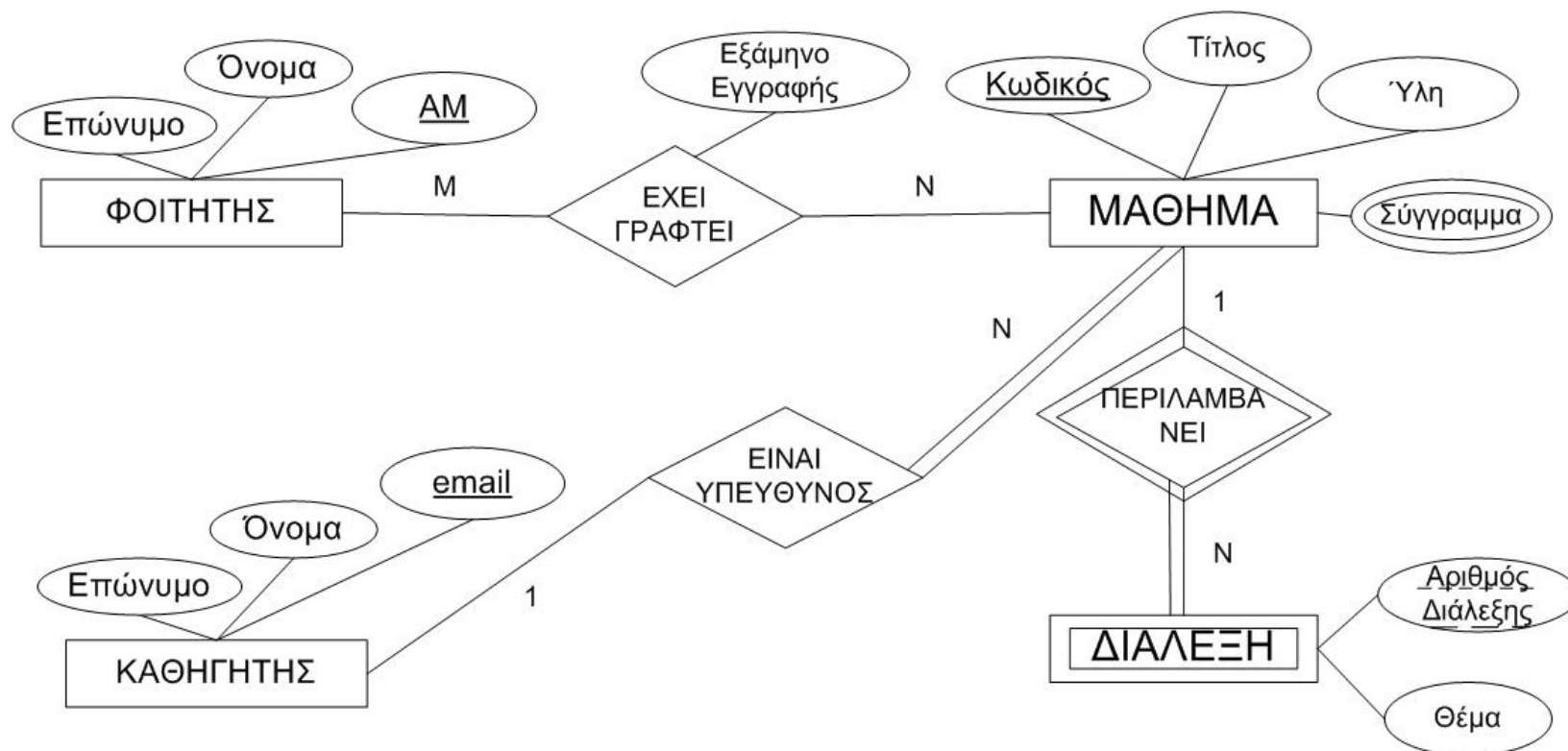


# Εργαστήριο Βάσεων Δεδομένων

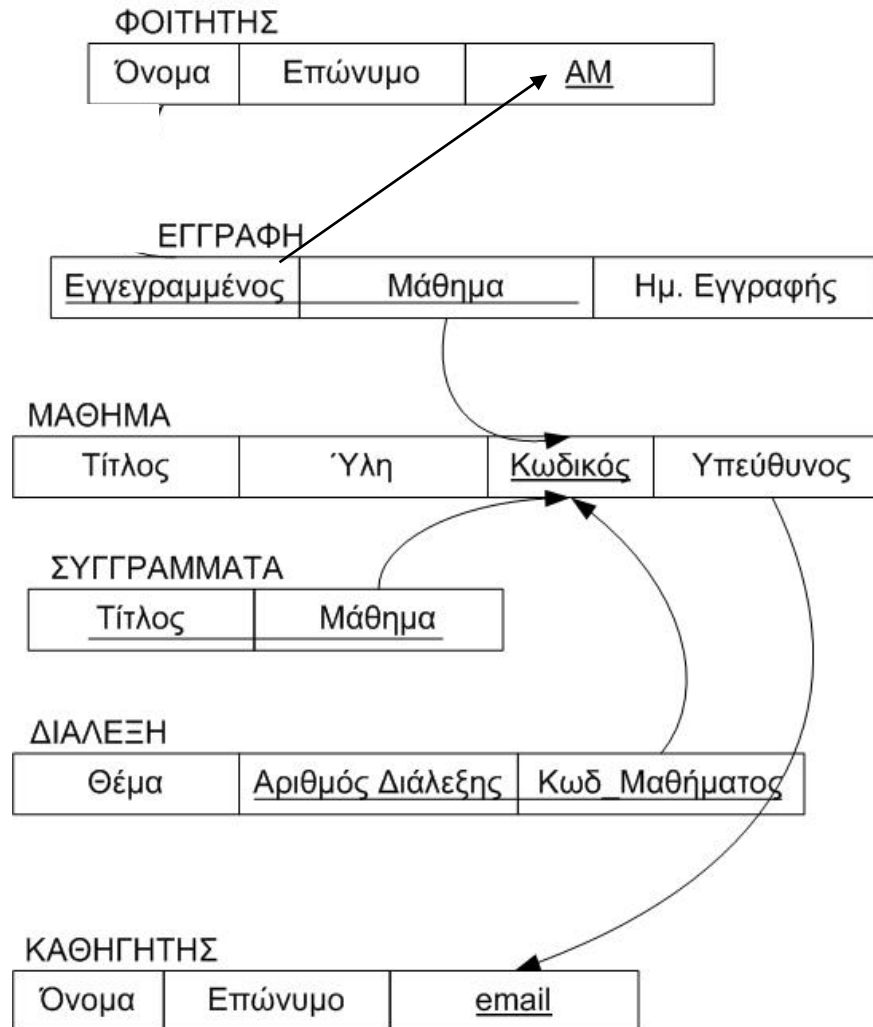
## Triggers



# Παράδειγμα - ER



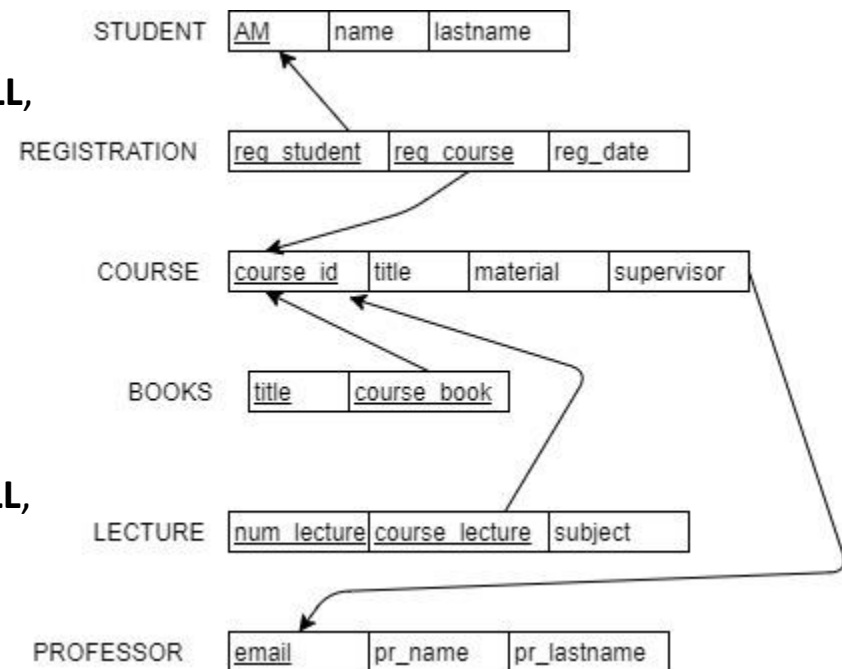
# Παράδειγμα-Σχεσιακό



# Παράδειγμα – Δημιουργία Πινάκων

```
CREATE TABLE student(  
  name VARCHAR(25) DEFAULT 'unknown' NOT NULL,  
  lastname VARCHAR(25) DEFAULT 'unknown' NOT NULL,  
  AM INT(5) NOT NULL AUTO_INCREMENT,  
  PRIMARY KEY(AM)  
);
```

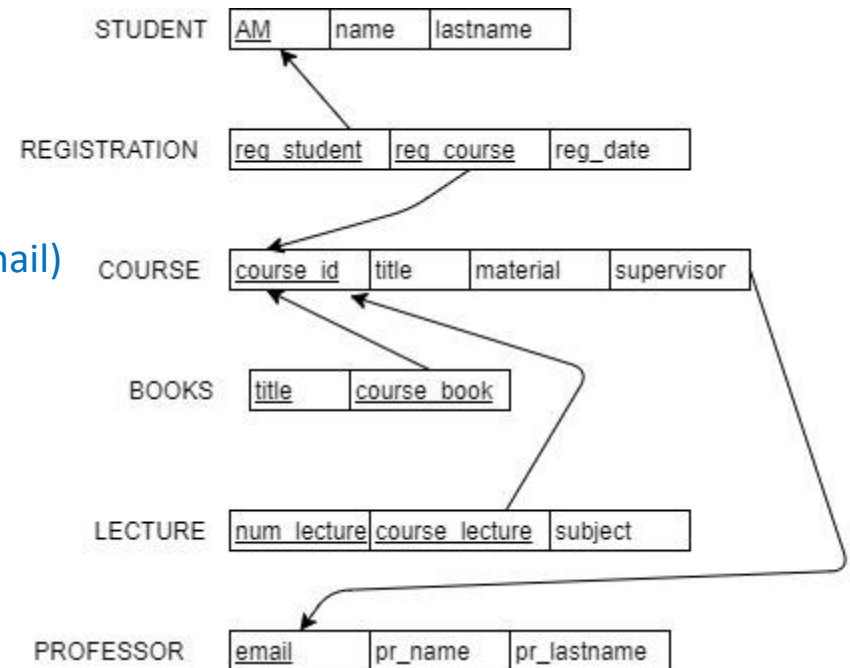
```
CREATE TABLE professor(  
  pr_name VARCHAR(25) DEFAULT 'unknown' NOT NULL,  
  pr_lastname VARCHAR(25) DEFAULT 'unknown' NOT NULL,  
  email VARCHAR(255) NOT NULL,  
  PRIMARY KEY(email)  
);
```



# Παράδειγμα – Δημιουργία Πινάκων

```
CREATE TABLE course (  
  title VARCHAR(255) DEFAULT 'unknown' NOT NULL,  
  material TEXT,  
  course_id INT(4) NOT NULL AUTO_INCREMENT,  
  supervisor VARCHAR(255) NOT NULL,  
  PRIMARY KEY(course_id),  
  UNIQUE(title),  
  CONSTRAINT SUPERVISED  
  FOREIGN KEY (supervisor) REFERENCES professor(email)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```

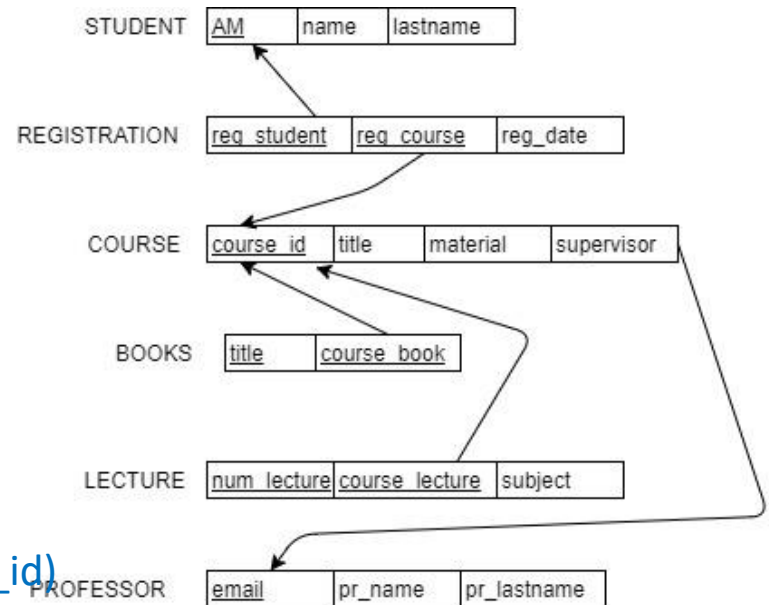
```
CREATE TABLE books (  
  title VARCHAR(128) DEFAULT 'Title' NOT NULL,  
  course_book INT(4) NOT NULL,  
  PRIMARY KEY(title,course_book),  
  CONSTRAINT CRSBOOK  
  FOREIGN KEY (course_book) REFERENCES course(course_id)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```



# Παράδειγμα – Δημιουργία Πινάκων

```
CREATE TABLE lecture (  
  subject VARCHAR(128),  
  num_lecture INT(2) NOT NULL,  
  course_lecture INT(4) NOT NULL,  
  PRIMARY KEY(num_lecture,course_lecture),  
  CONSTRAINT CRSLECTURE  
  FOREIGN KEY (course_lecture) REFERENCES course(course_id)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE registration (  
  reg_date DATE NOT NULL,  
  reg_student INT(5) NOT NULL,  
  reg_course INT(4) NOT NULL,  
  PRIMARY KEY(reg_student,reg_course),  
  CONSTRAINT CRSREGISTRATION  
  FOREIGN KEY (reg_course) REFERENCES course(course_id)  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  CONSTRAINT STDNTRREGISTRATION  
  FOREIGN KEY (reg_student) REFERENCES student(AM)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```



# Triggers: Βασικές έννοιες

- Ένας trigger είναι ένα κομμάτι κώδικα - μια ρουτίνα
- Συνδέεται με ένα συγκεκριμένο πίνακα και καλείται όταν συμβεί ένα γεγονός στον πίνακα
- Συχνές χρήσεις:
  - Έλεγχος ορθότητας των δεδομένων που εισάγονται στον πίνακα.
  - Υλοποίηση λογικών περιορισμών.
  - Υπολογισμός παραγόμενων τιμών.
  - Καταγραφή logs πρόσβασης και αλλαγών στον πίνακα.
- Η υλοποίηση με triggers κοστίζει αρκετά, καθώς καθυστερούν την προσπάθεια στον πίνακα.
  - Στη MySQL υποστηρίζονται από την 5.0.2 έκδοση.
  - Πριν την 5.1.6 έκδοση έπρεπε να έχεις SUPER δικαιώματα για να δηλώσεις trigger.
    - Η εξέταση πραγματοποιείται σε server που τρέχει η 5.5.45 έκδοση

# Βασικές εντολές

- Δημιουργία

`CREATE TRIGGER <όνομα trigger>`

- Διαγραφή

`DROP TRIGGER <όνομα trigger>`

- Ανάκτηση Κώδικα

`SHOW CREATE TRIGGER <όνομα trigger>`

- Ανάκτηση Λίστας Triggers

`SHOW TRIGGERS`

- Κλήση

Δε μπορούμε να καλέσουμε έναν trigger. Καλείται αυτόματα όταν συμβεί το γεγονός με το οποίο συνδέεται.



# Σχεδιασμός του trigger

- Για να δηλώσουμε έναν trigger πρέπει να καθορίσουμε
  - Σε ποιό πίνακα θα εφαρμοστεί:
    - Σε οποιοδήποτε πίνακα της βάσης στην οποία δημιουργείται.
  - Με τι είδος event θα συνδεθεί:
    - INSERT (όταν εισάγουμε μια εγγραφή στον πίνακα).
    - UPDATE (όταν τροποποιούμε μια εγγραφή στον πίνακα).
    - DELETE (όταν διαγράφουμε μια εγγραφή από τον πίνακα).
  - Σε ποια χρονική στιγμή (πότε) θα εκτελεστεί:
    - BEFORE Πριν ξεκινήσει το event.
    - AFTER Αφού ολοκληρωθεί το event.
  - Τη λειτουργία του trigger:
    - Στο σώμα του trigger μπορούμε να γράψουμε κανονικά κώδικα.

# Δημιουργία trigger

```
CREATE TRIGGER trigger_name trigger_time trigger_event  
ON table_name  
FOR EACH ROW trigger_body
```

- Όπου:
- *trigger\_name* Το όνομα του trigger
- *trigger\_time* Η χρονική στιγμή στην οποία θα κληθεί
- *trigger\_event* Το event με το οποίο συνδέεται
- *ON table\_name* Δήλωση του πίνακα στον οποίο ανήκει
- *FOR EACH ROW* Δήλωση ότι θα εκτελεστεί για κάθε γραμμή που περιλαμβάνεται στο event
- *trigger\_body* Ο κώδικας του trigger

# Δήλωση blocks κώδικα

- Στο σώμα του trigger είναι δυνατόν να περιλαμβάνονται πολλαπλές εντολές
  - Χρειάζεται η δυνατότητα ορισμού blocks εντολών  
`BEGIN ...εντολές block... END`
- Ο χαρακτήρας τερματισμού των εσωτερικών εντολών πρέπει να είναι διαφορετικός από το χαρακτήρα τερματισμού της `CREATE TRIGGER` εντολής
  - Αλλαγή του χαρακτήρα τερματισμού με την εντολή `DELIMITER`

# Ορισμός μεταβλητών στο περιβάλλον της MySQL

- Η mysql επιτρέπει τη δήλωση μεταβλητών
  - Η εμβέλειά τους είναι το τρέχον session
  - Το όνομά τους πρέπει να ξεκινάει με @ για να διαχωρίζονται από τις μεταβλητές του συστήματος
  - Η ανάθεση τιμής γίνεται με την εντολή SET

```
mysql> SET @x=4;
```

```
mysql> SET @y=7;
```

```
mysql> SET @z=@x-@y;
```

- Η εκτύπωση της τιμής της @z γίνεται με την εντολή SELECT

```
mysql> SELECT @z;
```

```
+-----+
```

```
| @z    |
```

```
+-----+
```

```
|    -3 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

# Δημιουργία Πίνακα Course

```
CREATE TABLE course (  
    title VARCHAR(255) DEFAULT 'unknown' NOT NULL,  
    material TEXT,  
    course_id INT(4) NOT NULL AUTO_INCREMENT,  
    supervisor VARCHAR(255) NOT NULL,  
    PRIMARY KEY(course_id),  
    UNIQUE(title),  
    CONSTRAINT SUPERVISED  
    FOREIGN KEY (supervisor) REFERENCES professor(email)  
    ON DELETE CASCADE ON UPDATE CASCADE);
```

# Δημιουργία trigger – Παράδειγμα

Όνομα keep\_count

**Λειτουργία** Κάθε φορά που εισάγεται ένα μάθημα ενημερώνεται μια μεταβλητή που κρατά το πλήθος των μαθημάτων


```
mysql>CREATE TRIGGER keep_count      Γιατί AFTER;  
->AFTER INSERT ON course  
->FOR EACH ROW  
->SET @courseCount=@courseCount+1;
```

```
mysql>SET @courseCount=(SELECT COUNT(*) FROM course);  
mysql>SELECT @courseCount;
```

```
+-----+  
| @courseCount |  
+-----+  
|           2 |  
+-----+  
1 row in set (0.00 sec)
```

```
mysql>INSERT INTO course (title,course_id,supervisor)  
->VALUES  
->('t1',NULL,'alex@upatras.gr'),  
->('t2',NULL,'alex@upatras.gr');
```

```
mysql>SELECT @courseCount;  
+-----+  
| @courseCount |  
+-----+  
|           4 |  
+-----+  
1 row in set (0.00 sec)
```



# Trigger Events

- Το event ορίζει το είδος του γεγονότος που ενεργοποιεί τον trigger.
- Είδη events:
  - **INSERT** Ενεργοποιείται κατά την εισαγωγή εγγραφών στον πίνακα. Δηλαδή από τα statements: INSERT, LOAD DATA και REPLACE
  - **UPDATE** Ενεργοποιείται κατά την τροποποίηση εγγραφών στον πίνακα. Δηλαδή από τα statements UPDATE
  - **DELETE** Ενεργοποιείται κατά τη διαγραφή εγγραφών στον πίνακα. Δηλαδή από τα statements: DELETE και REPLACE
    - Η DROP TABLE και TRUNCATE δεν καλεί τον trigger
- Η διαγραφή ή τροποποίηση εγγραφών λόγω ξένων κλειδιών δεν ενεργοποιεί triggers!

# Triggers ενός πίνακα

- Κάθε trigger σε ένα συγκεκριμένο πίνακα πρέπει να έχει μοναδικό συνδυασμό event και χρονικής στιγμής.
  - Μπορούμε να ορίσουμε πολλαπλούς triggers από την έκδοση 5.7 και μετά
- Άρα σε κάθε πίνακα μπορούμε να ορίσουμε το πολύ 6 triggers
  - BEFORE INSERT
  - AFTER INSERT
  - BEFORE UPDATE
  - AFTER UPDATE
  - BEFORE DELETE
  - AFTER DELETE

```
mysql> CREATE TRIGGER count_students
-> AFTER INSERT ON student
-> FOR EACH ROW
-> SET @stCount=@stCount+1;
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> CREATE TRIGGER count_students_alt
-> AFTER INSERT ON student
-> FOR EACH ROW
-> SET @studentsCount=@studentsCount+1;
ERROR 1235 (42000): This version of MySQL
doesn't yet support 'multiple triggers with
the same action time and event for one
table'
```



# Κώδικας μέσα στο σώμα του trigger

- Στο σώμα του trigger περιλαμβάνουμε τον κώδικα που εκτελείται όταν πυροδοτηθεί.
- Μπορούμε να συμπεριλάβουμε πολλαπλές ενέργειες χρησιμοποιώντας το block **BEGIN...END**.
  - Για τον ορισμό του σώματος του trigger επιβάλλεται η χρήση του delimiter.
  - Μπορούμε να ορίσουμε μεταβλητές
  - Μπορούμε να χρησιμοποιήσουμε εντολές ελέγχου ροής
  - Μπορούμε να ανακτήσουμε τα δεδομένα της βάσης με SELECT INTO και CURSORS
  - Μπορούμε να καλέσουμε stored procedures με την CALL
- Δεν μπορούμε:
  - Να εκτυπώσουμε αποτελέσματα μιας SELECT.
  - Να επεξεργαστούμε τον πίνακα στον οποίο εφαρμόζεται ο trigger.
- Οι περιορισμοί που ισχύουν για τους triggers ισχύουν και για τις stored procedures που καλούνται από triggers.

# Κώδικας σε triggers – Παραδείγμα 1

Όνομα init\_book

Λειτουργία Κάθε φορά που εισάγεται ένα μάθημα εισάγει μια εγγραφή στα βιβλία για το μάθημα αυτό με τον default τίτλο

```
mysql>DELIMITER $
mysql>CREATE TRIGGER init_book
->AFTER INSERT ON course
->FOR EACH ROW
->BEGIN
-> DECLARE cid INT(4);
-> SELECT MAX(course_id) INTO cid FROM course;
-> INSERT INTO books(title, course_book)
-> VALUES (DEFAULT, cid);
->END$
mysql>DELIMITER ;
```

# Δημιουργία Πίνακα Books

```
CREATE TABLE books (  
    title VARCHAR(128) DEFAULT 'Title' NOT NULL,  
    course_book INT(4) NOT NULL,  
    PRIMARY KEY(title,course_book),  
    CONSTRAINT CRSBOOK  
    FOREIGN KEY (course_book) REFERENCES course(course_id)  
    ON DELETE CASCADE ON UPDATE CASCADE);
```

# Κώδικας σε triggers – Παραδείγμα 1

Ελέγχουμε ότι ο trigger `init_book` λειτουργεί:

```
mysql>INSERT INTO  
course(title,course_id,supervisor)  
->VALUES  
->('t3',NULL,'alex@upatras.gr'),  
->('t4',NULL,'alex@upatras.gr');
```

```
mysql>SELECT books.title as Book,course.title as Course  
->FROM books  
->INNER JOIN course ON course_book=course_id;
```

Book	Course
Databases 1	Βάσεις Δεδομένων
Databases 1 2nd volume	Βάσεις Δεδομένων
Databases 2	Βάσεις Δεδομένων II
Title	t3
Title	t4

5 rows in set (0.00 sec)

# Κώδικας σε triggers - Περιορισμοί

- Από τον trigger δεν επιτρέπεται να εκτυπωθούν αποτελέσματα SELECT

```
mysql> DELIMITER $
mysql> CREATE TRIGGER init_book
-> AFTER INSERT ON course
-> FOR EACH ROW
-> BEGIN
-> DECLARE cid INT(4);
-> SELECT MAX(course_id) INTO cid FROM course;
-> INSERT INTO books(title, course_book)
-> VALUES (DEFAULT,cid);
-> SELECT title FROM books WHERE course_book=cid;
-> END$
ERROR 1415 (0A000): Not allowed to return a result set
from a trigger
```

Ο trigger δεν μπορεί να αλλάξει τον πίνακα στον οποίο εφαρμόζεται  
(ούτε αν καλέσει stored procedure που τον αλλάζει)

Όνομα overflow\_registrations –

```
mysql> DELIMITER $
mysql> CREATE TRIGGER overflow_registrations
-> BEFORE INSERT ON registration
-> FOR EACH ROW
-> BEGIN
-> DECLARE regnum INT;
-> DECLARE oldDate DATE;
-> SELECT COUNT(*) INTO regnum
-> FROM registration;
-> IF regnum >= 5 THEN
-> SELECT MIN(reg_date) INTO oldDate
-> FROM registration;
-> DELETE FROM registration
-> WHERE reg_date = oldDate;
-> END IF;
-> END $
mysql> DELIMITER ;
mysql> INSERT INTO registration(reg_date, reg_student, reg_course)
-> VALUES ('2012-12-03', 2193, 2);
```

```
CREATE TABLE registration (
    reg_date DATE NOT NULL,
    reg_student INT(5) NOT NULL,
    reg_course INT(4) NOT NULL,
    PRIMARY KEY (reg_student, reg_course),
    CONSTRAINT CRSREGISTRATION
    FOREIGN KEY (reg_course) REFERENCES course(course_id)
    ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT STDNTREGISTRATION
    FOREIGN KEY (reg_student) REFERENCES student(AM)
    ON DELETE CASCADE ON UPDATE CASCADE);
```

ERROR 1442 (HY000): Can't update table 'registration' in stored function/trigger because it is already used by statement which invoked this stored function/trigger.

# Προσπέλαση των τιμών της τρέχουσας εγγραφής

- Κάθε κλήση του trigger αφορά μια εγγραφή (row) του πίνακα, η οποία δημιουργείται, ενημερώνεται ή διαγράφεται.
- Η εγγραφή έχει δύο καταστάσεις σε σχέση με το trigger:
  - Η κατάστασή της πριν την πραγματοποίηση του event.
  - Η κατάστασή της αφού ολοκληρωθεί το event.
- Χρησιμοποιούμε τα ψευδώνυμα OLD και NEW για να αναφερθούμε στην αρχική και την τελική κατάσταση αντίστοιχα.
- Μέσω των αναφορών OLD και NEW μπορούμε να προσπελάσουμε τα στοιχεία της τρέχουσας εγγραφής (row):
  - **OLD.column\_name** Αναφέρεται στην τιμή της στήλης column\_name που είχε η εγγραφή πριν το event.
  - **NEW.column\_name** Αναφέρεται στην τιμή της στήλης column\_name που απέκτησε η εγγραφή μετά το event.

# OLD και NEW αναφορές

- Σε έναν trigger για **INSERT**
  - Δεν υπάρχει OLD εγγραφή, το OLD δεν χρησιμοποιείται.
  - Το NEW αναφέρεται είτε στην προσωρινή εγγραφή που θα εισαχθεί, είτε στην εγγραφή που μόλις έγινε.
- Σε έναν trigger για **UPDATE**
  - Το OLD αναφέρεται στις τιμές της εγγραφής πριν γίνει τροποποίηση.
  - Το NEW αναφέρεται είτε στις τιμές της εγγραφής που θα αποθηκευτούν, είτε στις τιμές που έχουν αποθηκευτεί ήδη.
- Σε έναν trigger για **DELETE**
  - Το OLD αναφέρεται στην εγγραφή που θα διαγραφεί.
  - Δεν υπάρχει NEW εγγραφή, το NEW δεν χρησιμοποιείται.
- Η αναφορά OLD επιτρέπει μόνο ανάγνωση των παλαιότερων τιμών.
- Η αναφορά NEW επιτρέπει και την αλλαγή των τιμών που είναι προς αποθήκευση.
  - Επιτρέπει την αλλαγή μόνο πριν την αποθήκευση, δηλαδή σε BEFORE χρόνο.
  - Στα AUTO\_INCREMENT πεδία η τιμή σε BEFORE χρόνο είναι μηδέν.



# Δημιουργία Πίνακα Registration

```
CREATE TABLE registration (  
    reg_date DATE NOT NULL,  
    reg_student INT(5) NOT NULL,  
    reg_course INT(4) NOT NULL,  
    PRIMARY KEY(reg_student,reg_course),  
    CONSTRAINT CRSREGISTRATION  
FOREIGN KEY (reg_course) REFERENCES  
course\(course\_id\)  
ON DELETE CASCADE ON UPDATE CASCADE,  
    CONSTRAINT STDNTREGISTRATION  
FOREIGN KEY (reg_student) REFERENCES student\(AM\)  
ON DELETE CASCADE ON UPDATE CASCADE);
```

# OLD και NEW σε BEFORE INSERT

```
mysql> CREATE TRIGGER checkRegDate  
-> BEFORE INSERT ON registration  
-> FOR EACH ROW  
-> BEGIN  
-> DECLARE currDate DATE;  
-> SET currDate=CURDATE();  
-> IF NEW.reg_date>currDate THEN SET NEW.reg_date=currDate;  
-> END IF;  
-> END$
```



Αν η ημερομηνία εγγραφής που θα εισαχθεί είναι μελλοντική, τότε την αντικαθιστά με τη σημερινή

Query OK, 0 rows affected (0.08 sec)

```
mysql> DELIMITER ;
```

*--Δοκιμάζουμε τον trigger:*

```
mysql> INSERT INTO registration (reg_date,reg_student,reg_course)  
->VALUES ('2021-01-03',2194,2);
```

Query OK, 1 row affected (0.02 sec)

```
mysql> select * from registration;
```

reg_date	reg_student	reg_course
2015-09-15	2129	2
2015-10-05	2129	3
2015-09-11	2193	2
2015-09-25	2193	3
2019-12-12	2194	2
2015-09-30	2194	3

6 rows in set (0.00 sec)

# OLD και NEW σε AFTER INSERT

```
mysql>DELIMITER $
mysql>CREATE TRIGGER returnId
->AFTER INSERT ON student
->FOR EACH ROW
->BEGIN
-> SET @newStudent=NEW.AM;
->END$
Query OK, 0 rows affected (0.08 sec)
```



Μετά την εισαγωγή ενός φοιτητή,  
επιστρέφει σε μια μεταβλητή στο  
περιβάλλον το AM του νέου φοιτητή

*--Δοκιμάζουμε τον trigger:*

```
mysql>DELIMITER ;
mysql> INSERT INTO student(AM,name,lastname) VALUES(NULL,'Γιώργος','Αναστασίου');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM student WHERE lastname='Αναστασίου';
+-----+-----+-----+
| name   | lastname | am   |
+-----+-----+-----+
| Γιώργος | Αναστασίου | 2194 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT @newStudent;
+-----+
| @newStudent |
+-----+
| 2194        |
+-----+
1 row in set (0.00 sec)
```

# OLD και NEW σε UPDATE

```
mysql>DELIMITER $
mysql>CREATE TRIGGER validateEmail
->BEFORE UPDATE ON professor
->FOR EACH ROW
->BEGIN
-> IF NEW.email NOT LIKE '%@%.%' THEN
-> SET NEW.email=OLD.email;
-> END IF;
->END$
Query OK, 0 rows affected (0.08 sec)
```



Αν επιχειρηθεί η τροποποίηση του email ενός καθηγητή, ελέγχει εάν είναι νόμιμη η μορφή της νέας διεύθυνσης και αν όχι διατηρεί την παλιά.

*--Δοκιμάζουμε τον trigger:*

```
mysql>DELIMITER $
```

```
mysql> update professor set email='nick@ceid' where email='alex@ceid.upatras.gr';
Query OK, 0 rows affected (0.064 sec)
Rows matched: 1  Changed: 0  Warnings: 0
```

```
mysql> update professor set email='nick@ceid.upatras.gr' where email='alex@ceid.upatras.gr';
Query OK, 1 row affected (0.061 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

# Ακύρωση του event από τον trigger


- Μια συχνή χρήση των triggers είναι ο έλεγχος των τιμών της εγγραφής προς επεξεργασία.
- Στην περίπτωση που ο έλεγχος αποτυγχάνει σε χρόνο BEFORE, οι συνήθεις επιλογές είναι δύο:
  - Διόρθωση της τιμής που δεν ικανοποιεί τους περιορισμούς σε επιτρεπτή τιμή.
  - Ακύρωση της λειτουργίας που πυροδότησε τον trigger.
- Για να ακυρωθεί το event που πυροδότησε τον trigger υπάρχουν δύο επιλογές:
  - Η πραγματοποίηση μέσα στον κώδικα κάποιας μη επιτρεπτής ενέργειας έτσι ώστε να προκληθεί σφάλμα και να ακυρωθεί η ενέργεια (εισαγωγή NULL σε κάποιο πεδίο που απαγορεύεται).
  - Η χρήση της εντολής **SIGNAL**. Είναι η εντολή που προκαλεί τη δημιουργία ενός error, το οποίο διακόπτει την εκτέλεση (υποστηρίζεται από την 5.5 έκδοση της MySQL)

# Δημιουργία Πίνακα Course

```
CREATE TABLE course (  
    title VARCHAR(255) DEFAULT 'unknown' NOT NULL,  
    material TEXT,  
    course_id INT(4) NOT NULL AUTO_INCREMENT,  
    supervisor VARCHAR(255) NOT NULL,  
    PRIMARY KEY(course_id),  
    UNIQUE(title),  
    CONSTRAINT SUPERVISED  
    FOREIGN KEY (supervisor) REFERENCES professor(email)  
    ON DELETE CASCADE ON UPDATE CASCADE);
```

# Ακύρωση του event – invalid operation

```
mysql>DELIMITER $
mysql>CREATE TRIGGER validateSupervisorCourses
->BEFORE INSERT ON course
->FOR EACH ROW
->BEGIN
-> DECLARE numOfCourses INT;
-> SELECT COUNT(*) INTO numOfCourses
-> FROM course
-> WHERE course.supervisor=NEW.supervisor;
-> IF numOfCourses>2 THEN
-> SET NEW.supervisor=NULL;
-> END IF;
->END$
Query OK, 0 rows affected (0.08 sec)
```



Πριν την εισαγωγή του μαθήματος ελέγχει πόσα μαθήματα έχει ήδη αναλάβει ο supervisor. Αν έχει αναλάβει τον μέγιστο αριθμό, τότε προκαλεί error

## Ελέγχουμε τον trigger:

```
mysql>DELIMITER ;
mysql> SELECT COUNT(*) FROM course WHERE supervisor='pap@ceid.upatras.gr';
+-----+
| COUNT(*) |
+-----+
| 1        |
+-----+
1 row in set (0.00 sec)
```

```
mysql> INSERT INTO course(title,supervisor) VALUES ('Lesson2','pap@ceid.upatras.gr');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO course(title,supervisor) VALUES ('Lesson3','pap@ceid.upatras.gr');
ERROR 1048 (23000): Column 'supervisor' cannot be null
```

# Ακύρωση του event μέσω SIGNAL

```
mysql>DELIMITER $
mysql>CREATE TRIGGER validateRegistration
->BEFORE INSERT ON registration
->FOR EACH ROW
->BEGIN
-> DECLARE currDate DATE;
-> DECLARE diff INT;
-> SET currDate=CURDATE();
-> SET diff=ABS(DATEDIFF(currDate,NEW.reg_date));
-> IF diff>=365 THEN
-> SIGNAL SQLSTATE VALUE '45000'
-> SET MESSAGE_TEXT = 'Invalid registration date! Must be within a year.';
-> END IF;
->END$
```



Πριν την εγγραφή ενός φοιτητή σε μάθημα, έλεγχος στην ημερομηνία εγγραφής. Αν είναι πάνω από ένα χρόνο σε απόσταση από σήμερα, αποτρέπεται η εγγραφή

Δοκιμάζουμε τον trigger:

```
mysql>DELIMITER ;
mysql> INSERT INTO registration(reg_date,reg_student,reg_course) VALUES ('2030-04-17',2194,2);
ERROR 1644 (45000): Invalid registration date! Must be within a year.

mysql> INSERT INTO registration(reg_date,reg_student,reg_course) VALUES ('2020-12-22',2194,2);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO registration(reg_date,reg_student,reg_course) VALUES ('2020-02-12',2193,2);
Query OK, 1 row affected (0.12 sec)
```



# Ακύρωση του event μέσω SIGNAL

- Στην **SIGNAL** ορίσαμε:
  - SQLSTATE VALUE - κωδικός του σήματος.
    - Το manual της MySQL ορίζει συγκεκριμένες τιμές και κανόνες για τον ορισμό τους.
    - Η τιμή που θα χρειαστείτε είναι '45000', που σημαίνει '**User defined error**'.
  - MESSAGE\_TEXT – το μήνυμα του σήματος.
- Συναρτήσεις που χρησιμοποιήθηκαν:
  - ABS(): Επιστρέφει το απόλυτο.
  - DATEDIFF(): Αφαιρεί δύο ημερομηνίες και επιστρέφει τη διαφορά τους σε μέρες.

# Δημιουργία Πίνακα Lecture

```
CREATE TABLE lecture (  
  subject VARCHAR(128),  
  num_lecture INT(2) NOT NULL,  
  course_lecture INT(4) NOT NULL,  
  PRIMARY KEY(num_lecture,course_lecture),  
  CONSTRAINT CRSLECTURE  
  FOREIGN KEY (course_lecture) REFERENCES course(course_id)  
  ON DELETE CASCADE ON UPDATE CASCADE);
```

# Παράδειγμα υλοποίησης AUTO\_INCREMENT

```
mysql>DELIMITER $
mysql>CREATE TRIGGER autoIncrementLecture
->BEFORE INSERT ON lecture
->FOR EACH ROW
->BEGIN
-> DECLARE maxNum INT(2);
-> SELECT MAX(num_lecture) INTO maxNum
-> FROM lecture
-> WHERE course_lecture=NEW.course_lecture;
-> SET NEW.num_lecture=maxNum+1;
->END$
mysql>DELIMITER ;
```



Υπολογίζει αυτόματα τον επόμενο  
αύξοντα αριθμό διάλεξης για  
ένα συγκεκριμένο μάθημα

Δοκιμάζουμε τον trigger:

```
mysql> SELECT * FROM lecture WHERE course_lecture=2;
+-----+-----+-----+
| subject          | num_lecture | course_lecture |
+-----+-----+-----+
| Εισαγωγή σε βάσεις | 1           | 2              |
| Ανάλυση απαιτήσεων | 2           | 2              |
| ER-Σχεσιακό       | 3           | 2              |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
mysql> INSERT INTO lecture(subject, num_lecture, course_lecture)
-> VALUES ('Εισαγωγή στη MySQL 1',0,2);
Query OK, 1 row affected (0.04 sec)
```

```
mysql> INSERT INTO lecture(subject, num_lecture, course_lecture)
-> VALUES ('Εισαγωγή στη MySQL 2',0,2);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM lecture WHERE course_lecture=2;
```

subject	num_lecture	course_lecture
Εισαγωγή σε βάσεις	1	2
Ανάλυση απαιτήσεων	2	2
ER-Σχεσιακό	3	2
Εισαγωγή στη MySQL 1	4	2
Εισαγωγή στη MySQL 2	5	2

```
5 rows in set (0.00 sec)
```

# Παράδειγμα υλοποίησης AUTO\_INCREMENT

Ερώτηση:

- Αν η SELECT δεν επιστρέψει τίποτα;
  - Πως πρώτη διάλεξη ενός μαθήματος θα πάρει δηλαδή την τιμή 1;

```
mysql>DELIMITER $
mysql>CREATE TRIGGER autoIncrementLecture
->BEFORE INSERT ON lecture
->FOR EACH ROW
->BEGIN
-> DECLARE maxNum INT(2);
-> SELECT MAX(num_lecture) INTO maxNum
-> FROM lecture
-> WHERE course_lecture=NEW.course_lecture;
-> SET NEW.num_lecture=maxNum+1;
->END$
mysql>DELIMITER ;
```

# Παράδειγμα υλοποίησης AUTO\_INCREMENT

Ερώτηση:

- Αν η SELECT δεν επιστρέψει τίποτα;
  - Πως πρώτη διάλεξη ενός μαθήματος θα πάρει δηλαδή την τιμή 1;

```
mysql>DELIMITER $
mysql>CREATE TRIGGER autoIncrementLecture
->BEFORE INSERT ON lecture
->FOR EACH ROW
->BEGIN
-> DECLARE maxNum INT(2);
-> SELECT MAX(num_lecture) INTO maxNum
-> FROM lecture
-> WHERE course_lecture=NEW.course_lecture;
-> IF maxNum is NULL THEN
->     SET maxNum=0;
-> END IF;
-> SET NEW.num_lecture=maxNum+1;
->END$
mysql>DELIMITER ;
```

# Προσεχώς

- Η επόμενη και τελευταία εργαστηριακή εξέταση θα γίνει 13/01/2022.
- Θα ανακοινωθεί ημερομηνία που θα γίνει εξ αποστάσεως
  - παρουσίαση του Πρότζεκτ,
  - παρουσίαση ορισμένων περσινών εργασιών, με σκοπό να δείτε παραδείγματα οργάνωσης και υλοποίησης της ομαδικής εργασίας.
- Υπάρχει στο eclass αναρτημένη διάλεξη που αφορά τη διασύνδεση MySQL με Java εφαρμογές με JDBC και απαιτείται για την υλοποίηση του πρότζεκτ.
- Το σετ εξάσκησης για τα triggers είναι στο σετ με τα stored procedures.

# Αναπλήρωση ασκήσεων

- Όσοι ΔΕΝ έχουν εξεταστεί σε μια εργαστηριακή άσκηση έχουν το δικαίωμα να αναπληρώσουν μόνο τη συγκριμένη άσκηση τον Ιανουάριο. Η αναπλήρωση δεν αποσκοπεί σε **βελτίωση βαθμολογίας ή διόρθωσης μηδενισμού** σε περίπτωση αντιγραφής.
- Σε ΜΙΑ μέρα και ώρα θα γίνουν παράλληλα οι αναπληρώσεις όλων των ασκήσεων. Η ημερομηνία αναπλήρωσης εργαστήριων θα ανακοινωθεί μετά την τελευταία εργαστηριακή άσκηση και θα είναι ανεξάρτητη των ωραρίων που γίνονται τα εργαστήρια.
- Θα βγει σχετική ανακοίνωση και θα κληθείτε να δηλώσετε στο eclass σε ποιά άσκηση θα συμμετάσχετε. Μόνο όσοι δηλώσουν εμπρόθεσμα θα προσέλθουν για αναπλήρωση.
- **ΟΣΟΙ δεν συμμετάσχουν στην αναπλήρωση του Ιανουαρίου** θα μπορούν να κάνουν την αναπλήρωση μιας άσκησης τον Σεπτέμβριο.



# Ευχαριστώ για την προσοχή σας!

- Απορίες;
- Διευκρινήσεις;