ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Διάλεξη 2η: ΕΙΣΑΓΩΓΗ - SQL

Structured Query Language (SQL)

- Τι είναι
- Εντολές Data Definition Language (DDL)
- Εντολές Data Manipulation Language (DML)
  Εντολές Data Control Language (DCL)
- Εντολές Transaction Control Language (TCL)
- Οι τελεστές JOIN

SQL: Τι είναι

- Γλώσσα για τη διαχείριση δεδομένων και υποβολή ερωτημάτων σε σχεσιακές βάσεις δεδομένων:
  - Δημιουργία/αλλαγή «σχήματος» (schema creation)
  - Εισαγωγή/αλλαγή δεδομένων
  - Αναζήτηση/επεξεργασία δεδομένων με διατύπωση ερωτημάτων
- Δηλωτική (declarative) κατά βάση γλώσσα, όμως και με στοιχεία προγραμματισμού (procedural)
- Πρότυπο ANSI (1986) και ISO (ISO/IEC 9075-1:2008)

SQL: DDL

- **Data Definition Language (DDL):** εντολές που χρησιμοποιούνται για τον ορισμό της δομής της ΒΔ (schema). Μερικά παραδείγματα:
  - ***CREATE*** - για τη δημιουργία αντικειμένων στη βάση
  - ***ALTER*** - για την αλλαγή της δομής
  - ***DROP*** - για τη διαγραφή αντικειμένων από τη βάση
  - ***TRUNCATE*** - διαγράφει όλες τις εγγραφές από έναν πίνακα και ελευθερώνει όλο το χώρο που είχε διατεθεί γι αυτές
  - ***COMMENT*** - προσθέτει σχόλια

SQL: DML

- **Data Manipulation Language (DML):** Εντολές για τη διαχείριση των δεδομένων των αντικειμένων της δομής. Μερικά παραδείγματα:
  - ***SELECT*** - ανακτά δεδομένα από τη βάση
  - ***INSERT*** - εισάγει δεδομένα σε κάποιον πίνακα
  - ***UPDATE*** - επικαιροποιεί υπάρχοντα δεδομένα σε πίνακα
  - ***DELETE*** - διαγράφει όλες τις εγγραφές ενός πίνακα, αλλά ο χώρος παραμένει δεσμευμένος

SQL: DCL

- **Data Control Language (DCL)**: Παραδείγματα:

  - ***GRANT*** - δίνει στους χρήστες δικαιώματα πρόσβασης στη ΒΔ
  - ***REVOKE*** – αποσύρει τα δικαιώματα πρόσβασης που δόθηκαν με εντολή GRANT

SQL: TCL

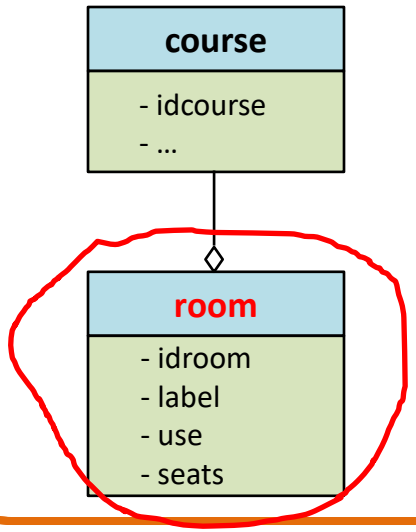
- **Transaction Control Language (TCL):** Εντολές που χρησιμοποιούνται για τη διαχείριση (μονιμοποίηση, αναίρεση) αλλαγών που έγιναν με εντολές DML. Επιτρέπει την ομαδοποίηση εντολών προς εκτέλεση:
  - ***COMMIT*** - αποθηκεύει ό, τι έχει γίνει
  - ***SAVEPOINT*** - σημειώνει ένα σημείο «συναλλαγής», όπου μπορεί αργότερα να επανέλθει η ροή εντολών με ROLLBACK
  - ***ROLLBACK*** - αναίρεση των αλλαγών που έγιναν από το τελευταίο COMMIT και έπειτα

Σχεδιασμός ΒΔ με αφετηρία ένα μοντέλο UML

- Υπάρχει ένα σύνολο πρακτικών οδηγιών
- Γενικά, ακολουθούμε τα εξής βήματα:
  1. Κάθε κλάση, που δεν συμμετέχει σε κάποια ιεραρχία με κληρονομικότητα, αντιστοιχίζεται σε έναν πίνακα
  2. Κάθε σχέση, πλην της κληρονομικότητας, αντιστοιχίζεται σε έναν πίνακα
  3. Μερικοί πίνακες μπορούν στη συνέχεια να συμπτυχθούν
  4. Οι ιεραρχίες κληρονομικότητας μετατρέπονται χωριστά

Αντιστοίχιση κλάσεων

- **Κάθε κλάση αντιστοιχίζεται σε έναν πίνακα**
  - Όνομα του πίνακα: το όνομα της κλάσης
  - Στήλες: οι ιδιότητες της κλάσης (attributes στο διάγραμμα UML)
  - Οι τύποι δεδομένων μπορεί να χρειαστεί να προσαρμοστούν στους διαθέσιμους τύπους της SQL [7]



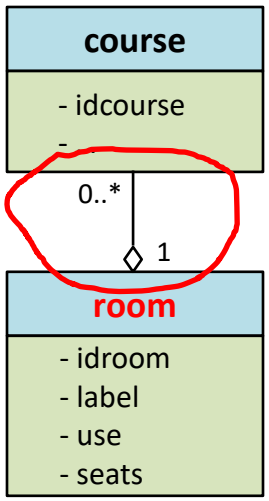

```
CREATE TABLE room (
  idroom  INTEGER NOT NULL PRIMARY KEY,
  label VARCHAR(50) DEFAULT NULL,
  use room_use,
  seats INTEGER
  );
```

δικός μας τύπος

Αντιστοίχιση σχέσεων

- **Κάθε σχέση μπορεί να αντιστοιχιστεί σε έναν πίνακα**
  - Όνομα του πίνακα: το όνομα της σχέσης, ή, αν υφίσταται, το όνομα της σχετιζόμενης κλάσης συσχέτισης (οποιοδήποτε νέο όνομα)
  - Στήλες: κύριες ιδιότητες των σχετιζόμενων κλάσεων, καθώς και της σχετιζόμενης κλάσης συσχέτισης (αν υπάρχει). Απαραίτητα, τα κύρια κλειδιά των πινάκων των σχετιζόμενων κλάσεων.

***Παράδειγμα**: Η σχέση 1:n μεταξύ room και course*

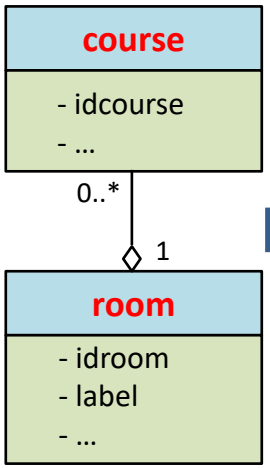



```
CREATE TABLE booking (
  roomid  INTEGER NOT NULL,
  courseid VARCHAR(50),
  PRIMARY KEY (roomid,courseid),
  FOREIGN KEY (roomid) REFERENCES room(idroom),
  FOREIGN KEY (courseid) REFERENCES course(idcourse)
  );
```

Συγχώνευση πινάκων

- **Οι πίνακες που προκύπτουν από σχέσεις 1:1 ή 1:n μπορούν να συγχωνευτούν με πίνακες των σχετιζόμενων κλάσεων**
  - Κλάσεις Α,Β, σε σχέση 1:n Το κλειδί του πίνακα Α μπορεί να προστεθεί, ως FOREIGN KEY, στον Β (την πλευρά πολλαπλότητας n) και ο πίνακας της σχέσης να καταργηθεί
  - Στήλες: κύριες ιδιότητες των σχετιζόμενων κλάσεων, καθώς και της σχετιζόμενης κλάσης συσχέτισης (αν υπάρχει)
  - Στη σχέση 1:1, το FOREIGN KEY μπορεί να προστεθεί στον έναν ή τον άλλον πίνακα

***Παράδειγμα**: Η σχέση 1:n μεταξύ room και course*



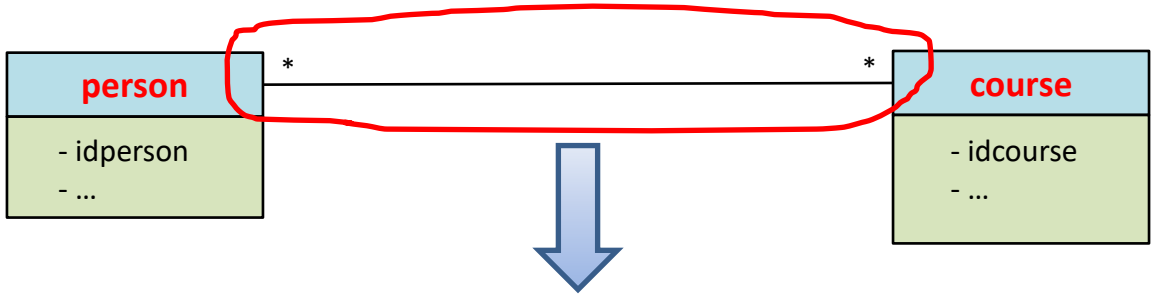
Κατάργηση του πίνακα booking. Κλειδί του πίνακα room για τον course

```
CREATE TABLE course (
  idcourse  VARCHAR(50),
  roomid  INTEGER NOT NULL,
  PRIMARY KEY (idcourse),
  FOREIGN KEY (roomid) REFERENCES room(idroom)
  );
```

Συγχώνευση πινάκων (συνέχεια)

- **Για τη σχέση n:n ο πίνακας θα πρέπει να διατηρηθεί!**

Παράδειγμα: *Η σχέση n:n μεταξύ person και course*



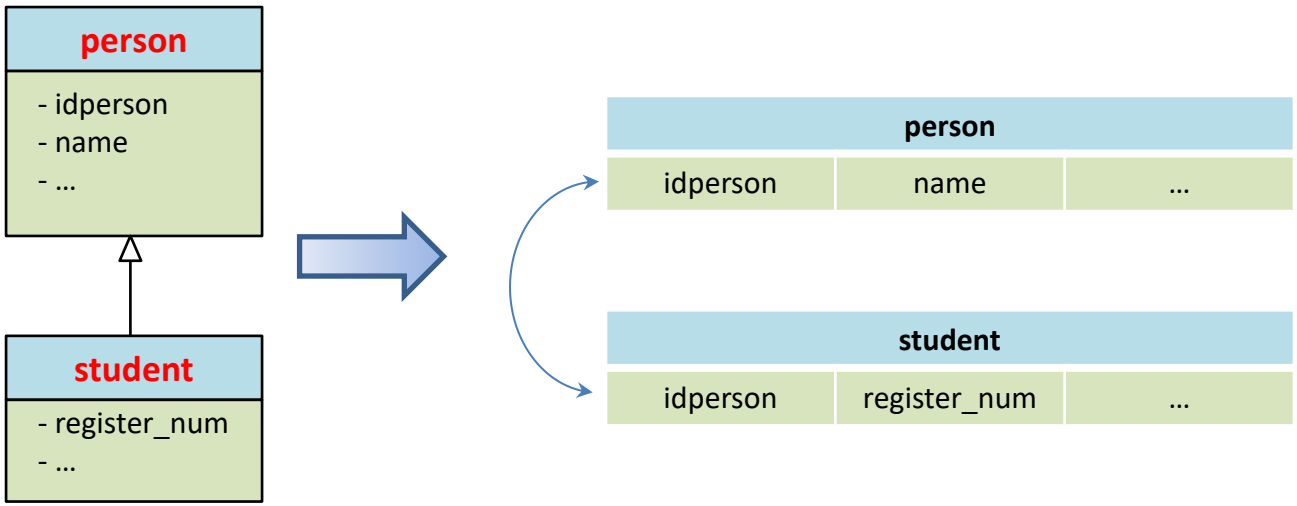

```
CREATE TABLE personcourse (
  personid  INTEGER NOT NULL,
  courseid  INTEGER NOT NULL,
  PRIMARY KEY (personid, courseid),
  FOREIGN KEY (personid) REFERENCES person(idperson),
  FOREIGN KEY (courseid) REFERENCES course(idcourse)
  );
```

Σχέσεις ιεραρχίας (κληρονομικότητα)

- **Σε ιεραρχίες υπερκλάσης-υποκλάσης, μόνο οι κλάσεις μετατρέπονται σε πίνακες, όχι οι σχέσεις**
- **Έχουμε τουλάχιστον δύο εναλλακτικές επιλογές:**
  - **Έναν πίνακα ανά κλάση της ιεραρχίας**
  - **Έναν και μόνο πίνακα για ολόκληρη την ιεραρχία**
- Κριτήρια για την επιλογή:
  - Απαιτούμενος χώρος
  - Ευκολία στη διατύπωση ερωτημάτων

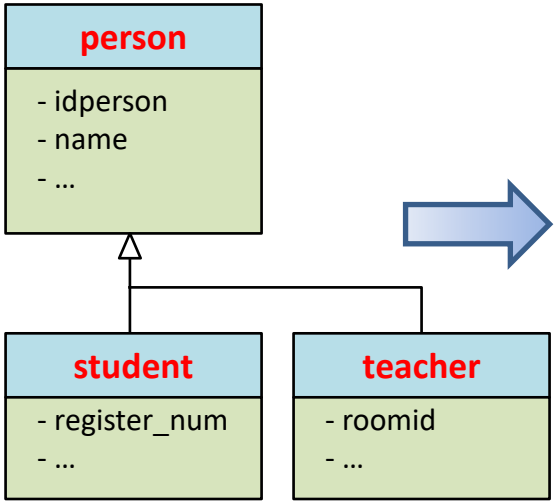
Σχέσεις ιεραρχίας (συνέχεια)

- **Επιλογή Ι: Ένας πίνακας ανά κλάση της ιεραρχίας,** με
  - Όνομα πίνακα: το όνομα της κλάσης
  - Στήλες:
    - τις ιδιότητες (attributes) της κλάσης στο διάγραμμα UML
    - Ιδιότητα-κλειδί της ρίζας της ιεραρχίας, που παραμένει κλειδί



Σχέσεις ιεραρχίας (συνέχεια)

- **Επιλογή ΙΙ: Ένας μόνον πίνακας για ολόκληρη την ιεραρχία.**
  - Όνομα πίνακα: μπορεί να είναι το όνομα της μητρικής κλάσης (ρίζας)
  - Στήλες: όλες οι ιδιότητες των κλάσεων που εμφανίζονται στην ιεραρχία, με νέα ονόματα αν χρειάζεται. Ιδιότητες που δεν υπάρχουν σε κάποια υποκλάση παίρνουν τιμή NULL[1].



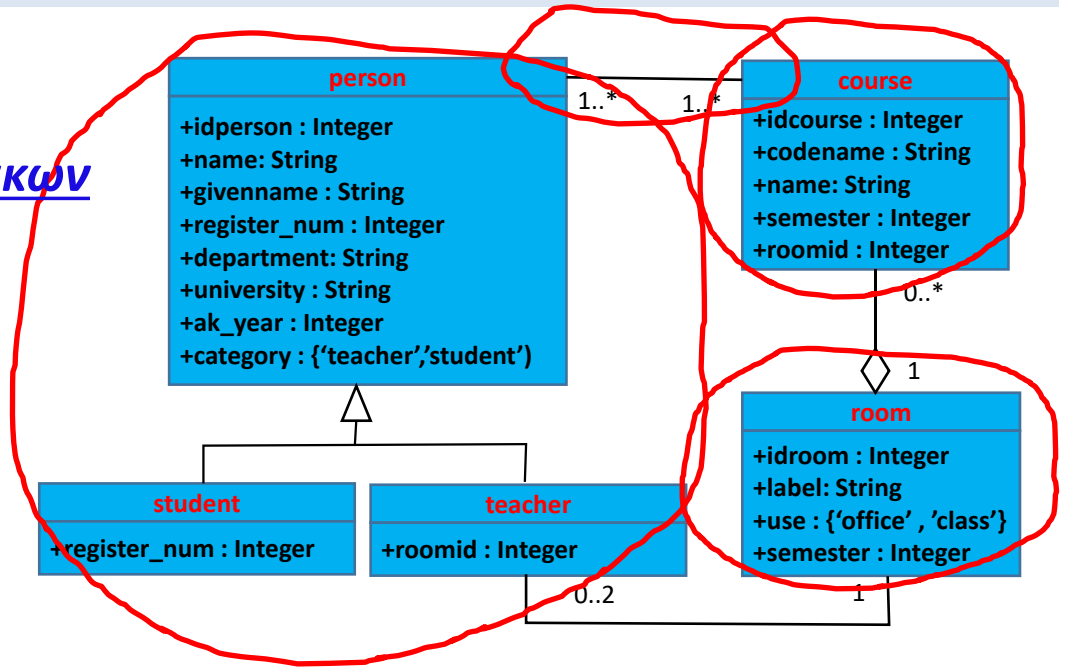
person				
idperson	**name**	**register_num**	**roomID**	**…**
52	ΜΗΤΡΟΥ	NULL	125	…
…	…	…	…	…
43	ΑΝΤΩΝΙΟΥ	2018005	NULL	…

[1] Γενικά, ο τύπος NULL χρησιμοποιείται στις περιπτώσεις: (α) άγνωστο, (β) μη διαθέσιμο, (γ) με εφαρμόσιμο

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Δημιουργία τεσσάρων πινάκων

- ✓ person
- ✓ course
- ✓ room
- ✓ personcourse



Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Πίνακας person

```
CREATE TYPE example2.person_category
    AS ENUM('student', 'teacher')
CREATE TABLE example2.person (
  idperson INTEGER NOT NULL PRIMARY KEY,
  name VARCHAR(50) DEFAULT NULL,
  givenname VARCHAR(50) DEFAULT NULL,
  register_num INTEGER DEFAULT NULL,
  department VARCHAR(50) DEFAULT NULL,
  university VARCHAR(50) DEFAULT NULL,
  ak_year INTEGER DEFAULT 2020,
  category person_category  DEFAULT 'student'
);
```

Έχοντας τον πίνακα student, ενημερώνουμε τον person ως εξής:

```
INSERT INTO example2.person (idperson, name, givenname, department, university, ak_year)
       SELECT idstudent, name, givenname, department, university, ak_year
       FROM example1.student;
```

idperson integer	name character varying(50)	givenname character varying(50)	register_num integer	department character varying(50)	university character varying(50)	ak_year integer	category example2.person_cate
202001	ΑΛΕΞΑΝΔΡΟΠΟΥΛΟΣ	ΧΡΗΣΤΟΣ		ΜΗΧ. ΧΩΡ., ΠΟΛΕΟΔ. ΚΑΙ ΠΕΡΙΦ. ΑΝΑΙ	ΠΑΝ. ΘΕΣΣΑΛΙΑΣ	2020	student
202002	ΑΡΑΠΕΛΛΗΣ	ΟΔΥΣΣΕΑΣ		ΤΜ. ΑΕΡΟΠΟΡΙΚΩΝ ΕΠΙΣΤ., ΚΑΤΕΥΘ. ΕΛ	ΣΧΟΛΗ ΙΚΑΡΩΝ	2020	student
202003	ΑΡΓΥΡΙΑΔΗ	ΠΟΛΥΞΕΝΗ		ΠΕΡΙΒΑΛΛΟΝΤΟΣ	ΠΑΝ. ΑΙΓΑΙΟΥ	2020	student
202004	ΒΑΚΑΛΟΠΟΥΛΟΥ	ΦΩΤΕΙΝΗ		ΑΤΜ	ΑΠΘ	2020	student
202005	ΒΟΥΛΓΑΡΑΚΗΣ	ΕΥΑΓΓΕΛΟΣ		ΑΤΜ	ΑΠΘ	2020	student
202006	ΓΕΡΟΥΣΗ	ΕΙΡΗΝΗ		ΑΤΜ	ΕΜΠ ΚΑΙ ΣΣΕ	2020	student
202007	ΚΑΡΑΓΙΑΝΝΗ	ΕΥΛΑΛΙΑ		ΑΤΜ	ΕΜΠ	2020	student
202008	ΚΑΤΣΑΛΑΚΗ	ΕΙΡΗΝΗ		ΠΛΗΡΟΦΟΡΙΚΗΣ	ΟΙΚΟΝ. ΠΑΝ. ΑΘΗΝ.	2020	student

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Πίνακες course *και* personcourse

```
CREATE TABLE example2.course (
  idcourse INTEGER NOT NULL PRIMARY KEY,
  code_name VARCHAR(15),
  name  VARCHAR(80),
  semester INTEGER NOT NULL,
  roomid INTEGER  NOT NULL
);
CREATE TABLE example2.personcourse (
  personid INTEGER NOT NULL,
  courseid INTEGER NOT NULL,
  PRIMARY KEY  (personid, courseid)
);
ALTER TABLE example2.personcourse
     ADD CONSTRAINT FK_personcourse_person
           FOREIGN KEY (personid)
           REFERENCES person (idperson)
           ON DELETE CASCADE;
ALTER TABLE example2.personcourse
     ADD CONSTRAINT FK_personcourse_course
          FOREIGN KEY (courseid)
          REFERENCES course (idcourse)
          ON DELETE CASCADE;
```

	idcourse [PK] integer	name character varying(80)	semester integer	roomid integer
1	1	ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ	2	1
2	2	ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ ΣΤΗ ΓΠ	1	2
*				

και τα δύο μαθήματα

μόνο το 2ο μάθημα

personid [PK] integer	courseid [PK] integer
202003	2
202004	1
202004	2
202006	2
202007	2
202008	2

καθηγητές

ΟΝΟΜΑ character varying(50)	ΕΠΩΝΥΜΟ character varying(50)	ΠΑΝΕΠ character var
ΒΑΣΙΛΕΙΟΣ	ΒΕΣΚΟΥΚΗΣ	ΕΜΠ
ΑΝΑΣΤΑΣΙΟΣ	ΖΑΦΕΙΡΟΠΟΥΛΟΣ	ΕΜΠ
ΙΩΑΝΝΗΣ	ΘΕΟΔΩΡΙΔΗΣ	ΠΑΠΕΙ
ΜΑΡΓΑΡΙΤΑ	ΚΟΚΛΑ	ΕΜΠ
ΝΙΚΟΛΑΟΣ	ΜΗΤΡΟΥ	ΕΜΠ

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Πίνακας room

```
CREATE TYPE example2.room_use AS ENUM ('office','class');

CREATE TABLE example2.room (
 idroom INTEGER NOT NULL PRIMARY KEY,
 label VARCHAR(50) DEFAULT NULL,
 use  example2.room_use,
-- use ENUM('office', 'class'), -- for oracle
 seats INTEGER
);
INSERT INTO example2.room VALUES (1,'ΣΑΤΜ-ΜΑ','class', 80);
…
```

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Ερωτήματα

1. Να βρεθούν οι σπουδαστές (όνομα & επώνυμο) που δήλωσαν το μάθημα 'ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ' και το επώνυμό τους ξεκινάει με γράμμα μέχρι και το Κ

```
SELECT givenname AS ONOMA, name AS ΕΠΩΝΥΜΟ FROM example2.person,
   (SELECT personid FROM example2.personcourse, example2.course
    WHERE courseid=idcourse AND course.name=
                  'ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ') s1
WHERE s1.personid=person.idperson
  AND person.category= 'student'
  AND name < 'Λ'
ORDER BY name;
```

ONOMA character varying(50)	ΕΠΩΝΥΜΟ character varying(50)
ΦΩΤΕΙΝΗ	ΒΑΚΑΛΟΠΟΥΛΟΥ
ΠΕΤΡΟΣ	ΚΥΡΙΑΚΟΥ

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Ερωτήματα (συνέχεια)

2. Να βρεθούν οι σπουδαστές οι οποίοι δήλωσαν 'ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ' και όχι 'ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ ΣΤΗ ΓΠ"

```
SELECT givenname AS ONOMA, name AS ΕΠΩΝΥΜΟ FROM example2.person,
       (SELECT personid FROM example2.personcourse, example2.course
        WHERE courseid=idcourse AND course.name='ΧΩΡΙΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ') s1
WHERE s1.personid=person.IDperson AND person.category= 'student'
      AND s1.personid NOT IN
         (SELECT personid FROM example2.personcourse, example2.course
          WHERE courseid=idcourse AND course.name='ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ ΣΤΗ ΓΠ')
ORDER BY name;
```

ONOMA character varying(50)	ΕΠΩΝΥΜΟ character varying(50)
ΠΕΤΡΟΣ	ΚΥΡΙΑΚΟΥ
ΜΙΧΑΗΛ	ΠΡΩΤΟΓΕΡΟΣ
ΕΙΡΗΝΗ	ΤΣΕΚΟΥΡΑ
ΒΑΣΙΛΕΙΟΣ	ΨΗΡΟΥΚΗΣ

Παράδειγμα 2: Βάση Δεδομένων ΔΠΜΣ (συνέχεια)

Ασκήση 1 (για παράδοση)

Θα αναρτηθεί στην ιστοσελίδα του μαθήματος

Βιβλιογραφία

[1] Elmasri, Navathe, *Fundamentals of Database Systems,* 6th edition, Addison-Wesley, 2011, (μπορεί να βρεθεί και σε ηλεκτρονική μορφή)

[1.a] “ *Θεμελιώδεις Αρχές Συστημάτων Βάσεων Δεδομένων, 7η* Έκδοση, ΔΙΑΥΛΟΣ Α.Ε. ΕΚΔΟΣΕΙΣ ΒΙΒΛΙΩΝ, 2016

[2] Clare Churcher, *Beginning Database Design – From Novice to Professional,* Apress 2007

[3] Hector Garcia-Molina, Jeffrey Ullman, *Database Systems: The Complete Book*, Pearson Education, 2009, 2013

[4] Jason Price*, Oracle Database 11g SQL*, Oracle Press, 2007

[5] Hans-Petter Halvorsen, Structured Query Language, 2017, https://www.halvorsen.blog/documents/tutorials/resources/Structured%20Query%20Language.pdf

[6] Unified Modeling Language® (UML®) Resource Page, http://www.uml.org/

[7] SQL Data types (στην Postgres) https://www.postgresql.org/docs/9.5/datatype.html

Βιβλιογραφία (συνέχεια)

- Elmasri, Navathe, *Fundamentals of Database Systems,* 6th edition, Addison-Wesley, 2011, (μπορεί να βρεθεί και σε ηλεκτρονική μορφή)