

week 3

Μόνιμη αποθήκευση δεδομένων

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Την εβδομάδα αυτή θα εξετάσουμε διάφορες τεχνικές για
μόνιμη αποθήκευση πληροφορίας στον υπολογιστή.

L3.1

Εισαγωγή, αρχεία csv

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Στην διάλεξη 3.1 θα κάνουμε μια γρήγορη επανάληψη στην αποθήκευση δεδομένων σε αρχεία χαρακτήρων και στη συνέχεια θα μιλήσουμε για αποθήκευση αντικειμένων μέσω της βιβλιοθήκης csv.

week 3

Μόνιμη αποθήκευση δεδομένων

Προγραμματίζω
με την python 

week 3

L3.1 Εισαγωγή, αρχεία csv

L3.2 Οι βιβλιοθήκες pickle και shelves – η
κωδικοποίηση JSON

L3.3 Σχεσιακές βάσεις δεδομένων

μάθημα L3.1

Εισαγωγή, αρχεία csv

Προγραμματίζω
με την python



- V3.1.1 Εισαγωγή, μόνιμη αποθήκευση
- V3.1.2 Η εφαρμογή contacts
- V3.1.3 Αποθήκευση σε αρχείο csv
- V3.1.4 Παράδειγμα: η contacts με csv

Προγραμματίζω
με την python



V3.1.1

Εισαγωγή – μόνιμη αποθήκευση

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Το πρόβλημα της μόνιμης αποθήκευσης

Προγραμματίζω
με την python



Η αναζήτηση τεχνολογιών για αποθήκευση δεδομένων σε δευτερεύουσα μνήμη για επαναχρησιμοποίηση

Απαιτήσεις:

- υψηλή ταχύτητα λειτουργιών δημιουργίας, ανάγνωσης, τροποποίησης διαγραφής δεδομένων (λειτουργίες Create, Read, Update and Delete CRUD)
- μικρό μέγεθος του ψηφιακού αποτυπώματος

Εναλλακτικοί τρόποι

Προγραμματίζω
με την python 

1. Αρχεία χαρακτήρων (text, csv)
2. Σειριοποίηση των δεδομένων σε αρχείο από **bytes** είτε με αποθήκευση μιας ενιαίας δομής (pickle) ή αποθήκευση πολλών δομών στις οποίες έχουμε πρόσβαση μέσω κλειδιών (shelve)
3. Αποθήκευση σε **πίνακες βάσεων δεδομένων** SQL ή μη-SQL

Σύνοψη της ενότητας

Προγραμματίζω
με την python 

Θα αναπτύξουμε την **εφαρμογή contacts**, λύνοντας το πρόβλημα της μόνιμης αποθήκευσης των δεδομένων της με διαφορετικούς τρόπους:

1. Σε αρχείο χαρακτήρων (βιβλιοθήκη `csv`)
2. Αποθήκευση της λίστας των αντικειμένων σε `pickle`
3. Αποθήκευση των αντικειμένων σε δεικτοδοτημένη ακολουθία δεδομένων με την `shelve`
4. Αποθήκευση σε σχεσιακή βάση δεδομένων `sqlite3`

Επίσης Θα εξετάσουμε την δομή **JSON** που επιτρέπει σειριοποίηση δομών δεδομένων προγράμματος μας και αποστολή τους σε τρίτους.

Προγραμματίζω
με την python



v3.1.2

Η εφαρμογή contacts

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts

Προγραμματίζω
με την python 

Θα κατασκευάσουμε μια απλή εφαρμογή για αποθήκευση των "επαφών" μας. Κάθε επαφή αποτελείται από δύο δεδομένα το όνομα και το τηλέφωνο. Το όνομα περιέχει πολλές λέξεις, υποθέτουμε ότι η τελευταία είναι το επίθετο της επαφής.



contacts εκδ.0

Προγραμματίζω
με την python 

Στην αρχική έκδοση η εφαρμογή μας επιτρέπει να δημιουργήσουμε επαφές, να διαγράψουμε επαφές, καθώς και να αναζητήσουμε επαφές. Για διευκόλυνση ελέγχου καλής λειτουργίας της εφαρμογής θα δημιουργήσουμε επίσης ένα εργαλείο 'μαζικής δημιουργίας εγγραφών'. Στην έκδοση v.0 δεν θα γίνεται μόνιμη αποθήκευση των επαφών.



contacts εκδ.0

```
class Contact():
    ''' κλάση επαφών με όνομα και τηλέφωνο
        με μεταβλητή κλάσης theContacts '''
    theContacts = {}
    def list_contacts(term = ''):
        for c in sorted(Contact.theContacts, key=lambda x : x.split()[-1]):
            if term:
                if term.lower() in c.lower():
                    print(Contact.theContacts[c])
            else: print(Contact.theContacts[c])

    def __init__(self, name, number=''): # μέθοδος δημιουργός επαφών
        print(name,number)
        self.name = name.strip()
        self.number = number.strip()
        Contact.theContacts[self.name] = self
    def __repr__(self): # μέθοδος εκτύπωσης επαφών
        return self.name + ': ' + self.number
```

Προγραμματίζω
με την python 



v3.1.3

Η εφαρμογή contacts: μέθοδος create_contacts

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts

Προγραμματίζω
με την python 

Για διευκόλυνση ελέγχου καλής λειτουργίας της εφαρμογής δημιουργούμε εργαλείο 'μαζικής δημιουργίας εγγραφών', τη μέθοδο `create_contacts()`.

Προγραμματίζω
με την python



V3.1.4

Αποθήκευση σε αρχείο csv

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Ανάκτηση από αρχείο κειμένου (επανάληψη)

Προγραμματίζω
με την python 

with open() ... for line in f

```
# ανάκτηση από αρχείο
students = {}
with open ('st_0.csv', 'r', encoding='utf-8') as f:
    for line in f:
        line = line.split(';')
        students[line[0]] = [line[1]]
        students[line[0]].extend([float(x) for x in line[2:]])
```

Αποθήκευση σε αρχείο κειμένου (επανάληψη)

Προγραμματίζω
με την python



with open() ... write()

```
# αποθήκευση σε αρχείο
with open('st_0.csv', 'w', encoding='utf-8') as f:
    for s in students:
        out = s
        for i in students[s]:
            out += ';' + str(i)
    f.write(out + '\n')
```

Αποθήκευση σε αρχείο csv comma separated values (αρχείο κειμένου οργανωμένο σε στήλες, διαβάζεται από excel)

Προγραμματίζω
με την python



'wt' (write text) και 'rt' είναι ισοδύναμα με 'w' και 'r'

```
print('...writing')
with open('vouna.csv', 'wt', encoding='utf-8') as f:
    writer = csv.writer(f, delimiter=';', quoting=csv.QUOTE_NONNUMERIC)
    for v in vouna:
        writer.writerow(v)
```

```
print('...reading')
with open('vouna.csv', 'rt', encoding='utf-8') as f:
    reader = csv.reader(f, delimiter=';', quoting=csv.QUOTE_NONNUMERIC)
    for row in reader:
        print(row)
```

keyword quoting
`csv.writer(... , quoting = τιμή)`

Προγραμματίζω
με την python 

`csv.QUOTE_ALL` βάλε όλες τις τιμές σε εισαγωγικά ανεξάρτητα τύπου

`csv.QUOTE_MINIMAL` βάλε σε εισαγωγικά μόνο ότι μπορεί να
μπερδέψει τον συντακτικό αναλυτή (**προκαθορισμένη τιμή**)

`csv.QUOTE_NONNUMERIC` βάλε σε εισαγωγικά όλα τα πεδία εκτός από
τους ακέραιους και πραγματικούς αριθμούς, οι αριθμοί μετατρέπονται
σε float.

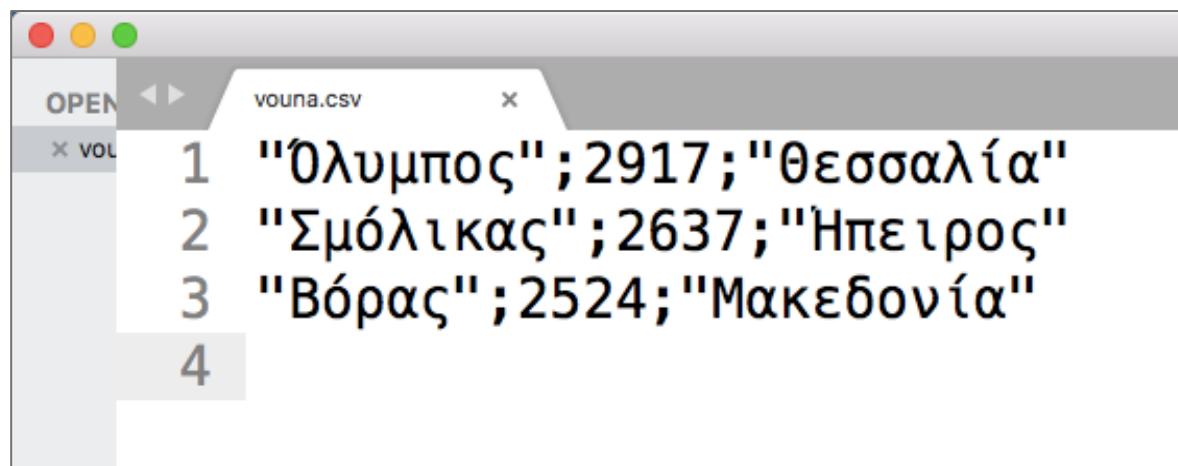
`csv.QUOTE_NONE` μην βάλεις εισαγωγικά. Αν βρεις χαρακτήρα
εισαγωγικό θεώρησε τον ως στοιχείο της συμβολοσειράς

από λίστα vouna σε αρχείο vouna.csv

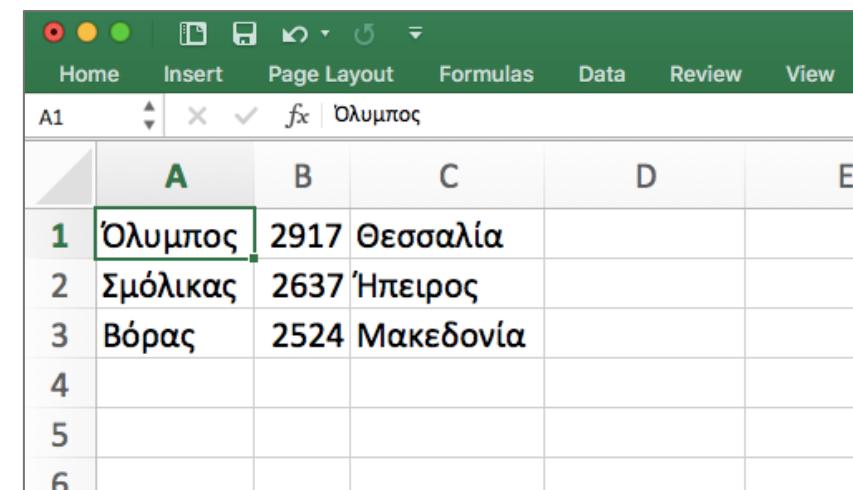
Προγραμματίζω
με την python 

```
vouna = [[ 'Ολυμπος' , 2917 , 'Θεσσαλία' ] ,  
          [ 'Σμόλικας' , 2637 , 'Ηπειρος' ] ,  
          [ 'Βόρας' , 2524 , 'Μακεδονία' ] ]
```

vouna.csv



A terminal window titled "vouna.csv" displays the following text:
1 "Ολυμπος";2917;"Θεσσαλία"
2 "Σμόλικας";2637;"Ηπειρος"
3 "Βόρας";2524;"Μακεδονία"
4



	A	B	C	D	E
1	Όλυμπος	2917	Θεσσαλία		
2	Σμόλικας	2637	Ηπειρος		
3	Βόρας	2524	Μακεδονία		
4					
5					
6					

Προγραμματίζω
με την python



V3.1.5 παράδειγμα: η contacts με csv

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts εκδ.1

Προγραμματίζω
με την python 

αποθήκευση

```
with open(self.filename, 'w', encoding='utf-8') as f:  
    writer = csv.writer(f, delimiter=';', quoting=csv.QUOTE_ALL)  
    for c, contact in Contact.theContacts.items():  
        writer.writerow([contact.name, contact.number])
```

ανάκτηση

```
with open(self.filename, 'r', encoding='utf-8') as f:  
    reader = csv.reader(f, delimiter=';', quoting=csv.QUOTE_ALL)  
    for row in reader:  
        Contact(row[0], row[1])
```

L3.2

Οι βιβλιοθήκες pickle και shelve – η κωδικοποίηση JSON

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Στην διάλεξη 3.2 θα μιλήσουμε για αποθήκευση αντικειμένων μέσω της βιβλιοθήκης pickle, της βιβλιοθήκης shelve, ενώ θα γίνει σύντομη αναφορά στη βιβλιοθήκη json

week 3

Μόνιμη αποθήκευση δεδομένων

Προγραμματίζω
με την python 

week 3

L3.1 Εισαγωγή, αρχεία csv

L3.2 Οι βιβλιοθήκες pickle και shelves

L3.3 Σχεσιακές βάσεις δεδομένων

μάθημα L3.2

Οι βιβλιοθήκες pickle και shelve – η κωδικοποίηση JSON

Προγραμματίζω
με την python



- V3.2.1 Η βιβλιοθήκη pickle
- V3.2.2 Παράδειγμα: η contacts με pickle
- V3.2.3 Η βιβλιοθήκη shelve
- V3.2.4 Παράδειγμα: η contacts με shelve
- V3.2.5 JSON

Προγραμματίζω
με την python



V3.2.1

Η βιβλιοθήκη pickle

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

pickle

Προγραμματίζω
με την python 

Μόνιμη αποθήκευση δομών δεδομένων
σε αρχείο ως ακολουθία bytes

η pickle κάνει σειριοποίηση των
δεδομένων πριν τα μετατρέψει σε bytes



(πίκλες, τουρσιά, παραδοσιακός
τρόπος συντήρησης κύρια
λαχανικών σε ξύδι και άλμη)

Προγραμματίζω
με την python



- + γρήγορος και αποδοτικός τρόπος αποθήκευσης,
άμεση αποθήκευση δεδομένων από σύνθετες δομές
(λεξικά, λίστες αντικειμένων, κλπ)
- όχι συμβατός με άλλες γλώσσες προγραμματισμού,
θέματα ασφάλειας

pickle : αποθήκευση λεξικού

Προγραμματίζω
με την python 

pickle.dump(δομή, αρχείο)

```
my_dict = {'1': 'καλημέρα', (1,2): 33.3+22.2j, 3: [ [10,20],[30,40]] }  
print(my_dict)  
with open('pickle0.db', 'wb') as f:  
    pickle.dump(my_dict, f)
```

pickle : ανάκτηση του λεξικού

Προγραμματίζω
με την python 

δομή = pickle.load(αρχείο)

```
if os.path.isfile('pickle0.db'):
    with open('pickle0.db', 'rb') as f:
        my_dict = pickle.load(f)
        for k,v in my_dict.items(): print(k, '\t--->', v)
else:
    print('το αρχείο δεν υπάρχει')
```

pickle αποθήκευση ενός container αντικειμένων

Προγραμματίζω
με την python 

αποθήκευση

```
with open('obj.db', 'wb') as f:  
    pickle.dump(Obj.theObjects, f)
```

ανάκτηση

```
with open('obj.db', 'rb') as f:  
    Obj.theObjects = pickle.load(f)
```

Προσοχή: στην ανάκτηση η κλάση Obj θα πρέπει να έχει
οριστεί! αλλιώς η pickle.load δίνει error

Προγραμματίζω
με την python



V3.2.2

Παράδειγμα: η contacts με pickle

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts εκδ.2

αποθήκευση

```
def store(self):
    with open(self.filename, 'wb') as f:
        pickle.dump(Contact.theContacts, f)
```

ανάκτηση

```
def retrieve(self):
    if os.path.isfile(self.filename):
        with open(self.filename, 'rb') as f:
            Contact.theContacts = pickle.load(f)
        return True
    else: return False
```

Προγραμματίζω
με την python 

Προγραμματίζω
με την python



V3.2.3

Η βιβλιοθήκη shelve

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



Η βιβλιοθήκη `shelve`

Προγραμματίζω
με την python



- Η βιβλιοθήκη `shelve` χρησιμεύει για αποθήκευση περισσότερων από ένα αντικείμενα στα οποία να έχουμε άμεση πρόσβαση, αν δεν επιθυμούμε να αποθηκεύσουμε τον container όπως στο προηγούμενο παράδειγμα.
- Με τον τρόπο αυτό μπορούμε να ανασύρουμε επί μέρους αντικείμενα, ανεξάρτητα της σειράς που τα αποθηκεύσαμε.



Η βιβλιοθήκη shelve

Προγραμματίζω
με την python



```
import shelve
```

```
d = shelve.open(filename) # άνοιγμα αρχείου filename.db
d[key] = data # αποθήκευση δεδομένων στο κλειδί
data = d[key] # αντιγράφει δεδομένα (KeyError αν key δεν υπάρχει)
del d[key] # διαγράφει δεδομένα (KeyError αν key δεν υπάρχει)
flag = key in d # True αν το key υπάρχει
klist = list(d.keys()) # λίστα κλειδιών (αργή!)
```

Βιβλιοθήκη shelve παράδειγμα

```
>>> from person import Person  
>>> nikos = Person('Νίκος', 'δάσκαλος', 700)  
>>> maria = Person('Μαρία', 'μουσικός', 400)  
  
>>> import shelve  
>>> dbase = shelve.open( 'contacts' ) ←  
>>> for obj in (nikos, maria):  
    ... dbase[obj.name] = obj  
>>> dbase.close()
```

Προγραμματίζω
με την python 

δημιουργείται αρχείο
contacts.db

shelve ανάγνωση δεδομένων

```
>>> dbase = shelve.open('contacts')
>>> list(dbase.keys())
['Νίκος', 'Μαρία']
>>> print(dbase['Μαρία'])
Μαρία 400
>>>
>>> print(dbase['Νίκος'].pay)
700
```



άνοιγμα αρχείου
contacts.db

Προγραμματίζω
με την python 

Προγραμματίζω
με την python



V3.2.4

Παράδειγμα: η contacts με shelve

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts εκδ.3a

αποθήκευση

ανάκτηση

Προγραμματίζω
με την python



```
def __init__(self):
    self.db = 'contacts'

def store(self):
    if os.path.isfile(self.db+'.db'):
        os.remove(self.db+'.db')
    with shelve.open(self.db) as db:
        for c_name, contact in Contact.theContacts.items():
            db[c_name] = contact

def retrieve(self):
    with shelve.open(self.db) as db:
        for k in db:
            Contact(db[k].name, db[k].number)
```



contacts εκδ.3b

Προγραμματίζω
με την python

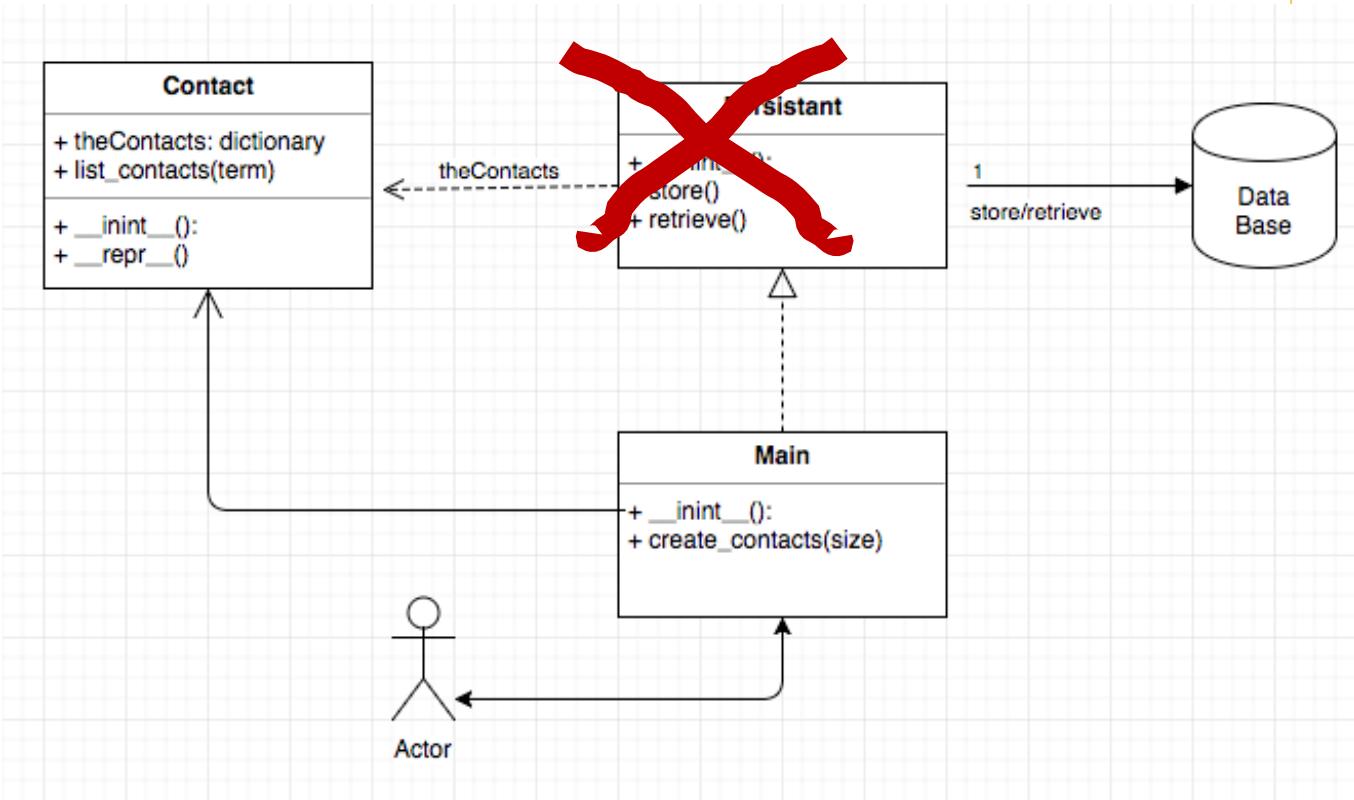


Αλλαγή παραδείγματος, αντί για ανάκτηση όλης της πληροφορίας από τη βάση δεδομένων στη RAM στην αρχή του προγράμματος και αποθήκευση της τελικά στη βάση δεδομένων στο τέλος, στην έκδοση 3b θα ανακτούμε κάθε φορά μόνο τις εγγραφές (ή εγγραφή) που ζητάει ο χρήστης και θα πραγματοποιούμε την ενημέρωση της βάσης μόνο για αυτή την πληροφορία (πράξεις insert και update) άμεσα.



contacts εκδ.3b

Προγραμματίζω
με την python



class Contact():

''' κλάση επαφών [όνομα , τηλέφωνο]

```
theContacts = {}
```

```
db = 'contacts'
```

```
def count_records(): # COUNT
```

```
def retrieve_contacts(term = ''): # RETRIEVE
```

```
def del_contact(id): # DELETE
```

```
def __init__(self, name, number='', new=False):
```

```
def insert(self): # INSERT
```

```
def set_number(self, number): # UPDATE
```

```
def __repr__(self): #
```

Προγραμματίζω
με την python



V3.2.5 JSON

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

JSON

Προγραμματίζω
με την python 

Η JSON (JavaScript Object Notation) είναι ένα πρότυπο ανταλλαγής δεδομένων ως ακολουθίες χαρακτήρων (serialization). Είναι ευανάγνωστο από τους ανθρώπους και εύκολο στην επεξεργασία για τις μηχανές. Έχει προκύψει από τη γλώσσα JavaScript, Standard ECMA-262 εκδ.3(1999). Το JSON είναι ένα πρότυπο ανεξάρτητο από γλώσσες προγραμματισμού. Είναι εύκολη η αντιστοίχισή του σε δομές της Python, όπως το λεξικό και η λίστα. Είναι σήμερα το πιο διαδεδομένο πρότυπο ανταλλαγής δεδομένων (πχ για διεπαφές δεδομένων στο διαδίκτυο, API).

Η βιβλιοθήκη json

Προγραμματίζω
με την python 

```
j_data = json.dumps(data) # μετατροπή data → json  
data = json.loads(j_data) # μετατροπή json → data
```

Η βιβλιοθήκη επιτρέπει την μετατροπή αντικειμένων του προγράμματος σε σειριοποιημένη αναπαράσταση τύπου JSON `json.dumps()` και το αντίστροφο `json.loads()`.

Στο στοιχείο αυτό μοιάζει με την pickle, η διαφορά είναι ότι η JSON είναι στάνταρ και επιτρέπει τη διαλειτουργικότητα με άλλες γλώσσες προγραμματισμού.

Η βιβλιοθήκη json παράδειγμα

Προγραμματίζω
με την python 

```
j_data = json.dumps(data) # μετατροπή data → json
```

```
import json
data = [{ 'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
print('DATA:', repr(data))
j_data = json.dumps(data)
print('JSON:', j_data)
```

```
DATA: [ {'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
JSON: [{"a": "A", "b": [10, 20.0], "c": 3.0}]
```

δεν υπάρχει απόλυτη αντιστοίχιση, επίσης δομές δεν υποστηρίζονται,
αν `data = 1+2j`

`TypeError: Object of type 'complex' is not JSON serializable`

Η βιβλιοθήκη json παράδειγμα

Προγραμματίζω
με την python 

```
data = json.loads(j_data) # μετατροπή json → data

import json
data = [{ 'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
print('DATA:', repr(data))

j_data = json.dumps(data) # από λίστα python σε json
print('JSON:', j_data)

from_json = json.loads(j_data) # από json σε δεδομένα python
print('DECODED:', from_json)

print('ORIGINAL:', type(data[0]['b']))
print('DECODED :', type(from_json[0]['b']))
```

Η βιβλιοθήκη json παράδειγμα

Προγραμματίζω
με την python 

```
j_data = json.dumps(data) # μετατροπή data → json
```

```
import json
data = [{ 'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
print('DATA:', repr(data))
j_data = json.dumps(data)
print('JSON:', j_data)
```

```
DATA: [ {'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
JSON: [{"a": "A", "b": [10, 20.0], "c": 3.0}]
```

Η βιβλιοθήκη json παράδειγμα

Προγραμματίζω
με την python 

```
j_data = json.dumps(data) # μετατροπή data → json
```

```
import json
data = [{ 'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
print('DATA:', repr(data))
j_data = json.dumps(data)
print('JSON:', j_data)
```

```
DATA: [ {'a': 'A', 'b': (10, 20.0), 'c': 3.0}]
JSON: [{"a": "A", "b": [10, 20.0], "c": 3.0}]
```

Η βιβλιοθήκη json μετατροπή λεξικού σε json

Προγραμματίζω
με την python 

η json δεν δέχεται ως κλειδί άλλους τύπους
δεδομένων πλην συμβολοσειρών

παράδειγμα με κλειδιά λεξικού μη αλφαριθμητικά

```
data = [{'a': 'A', 'b': (10, 20.0), ('d',): 'D tuple'}]
```

```
print('παράδειγμα με κλειδί λεξικό')
```

```
try:
```

```
    print(json.dumps(data))
```

```
except TypeError as err:
```

```
    print('ERROR:', err)
```

παράδειγμα με κλειδί λεξικό
ERROR: keys must be a string
παράλειψη μη αλφαριθμητικών κλειδιών
[{"a": "A", "b": [10, 20.0]}]

```
print('παράλειψη μη αλφαριθμητικών κλειδιών')
```

```
print(json.dumps(data, skipkeys=True))
```

Η βιβλιοθήκη json αποθήκευση και ανάκτηση σε/από αρχεία

`json.dump(data, file)` `json.load(file)`

```
data = [{'a': 'A', 'b': (10, 20.0), ('d',): 'D tuple'}]
# αποθήκευση της δομής json σε αρχείο
with open('file.json', 'w', encoding="utf8") as f:
    json.dump(data, f, skipkeys=True) # προσοχή παράλειψη μη-αλφα κλειδιών

# ανάκτηση από αρχείο
with open('file.json', 'r', encoding='utf-8') as f:
    data = json.load(f)
print(data)
```

Προγραμματίζω
με την python 

Άσκηση

Προγραμματίζω
με την python 

Τροποποιήστε τον κώδικα contacts. ν3α ώστε
αντί για τη βιβλιοθήκη pickle να
χρησιμοποιήσετε την json
Τι πλεονεκτήματα και μειονεκτήματα έχει
αυτή η λύση;

L3.3

Σχεσιακές βάσεις δεδομένων

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Στην διάλεξη 3.3 θα εξετάσουμε τρόπους αποθήκευσης των δεδομένων σε σχεσιακές βάσεις δεδομένων. Ως παράδειγμα θα εξετάσουμε τη βιβλιοθήκη sqlite3 που περιλαμβάνεται στη βασική python.

week 3

Μόνιμη αποθήκευση δεδομένων

Προγραμματίζω
με την python 

week 3

L3.1 Εισαγωγή, αρχεία csv

L3.2 Οι βιβλιοθήκες pickle και shelves

L3.3 Σχεσιακές βάσεις δεδομένων

μάθημα L3.3

Σχεσιακές βάσεις δεδομένων

Προγραμματίζω
με την python



- V3.3.1 Σύντομη εισαγωγή στο σχεσιακό μοντέλο SQL
- V3.3.2 Η βιβλιοθήκη sqlite3
- V3.3.3 Παράδειγμα: η contacts με sqlite3
- V3.3.4 Σύγκριση διαφορετικών προσεγγίσεων

Προγραμματίζω
με την python

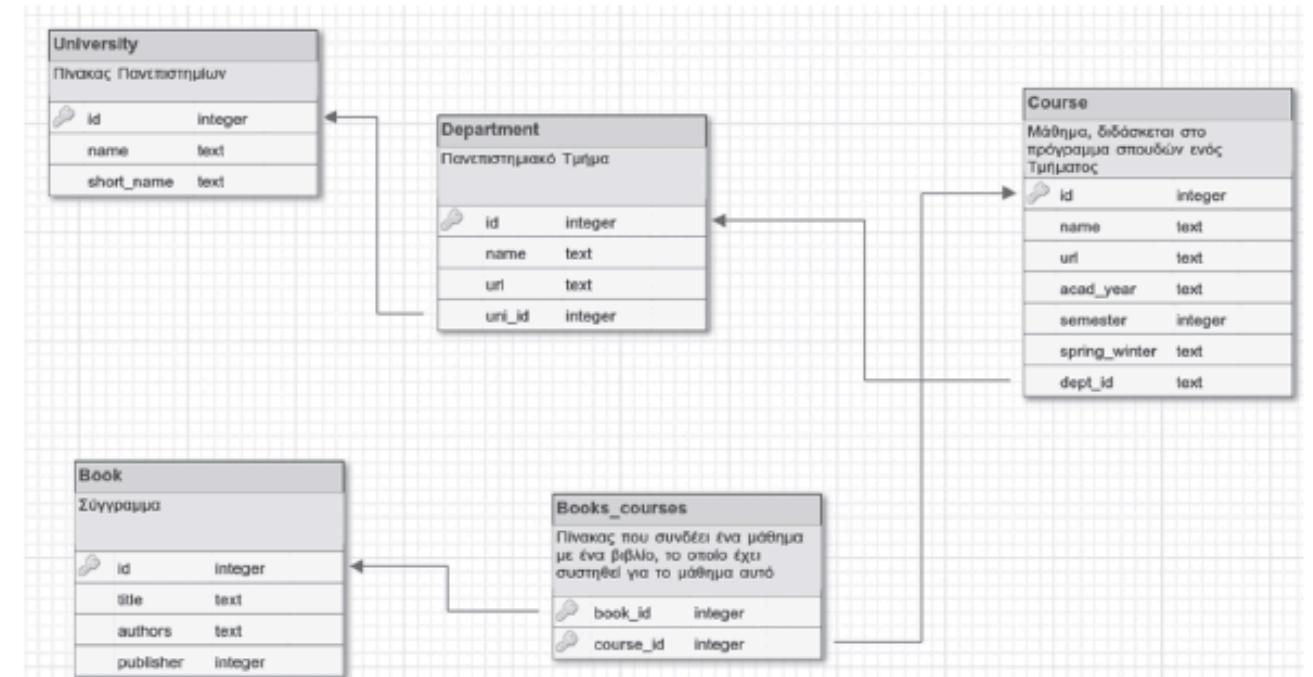
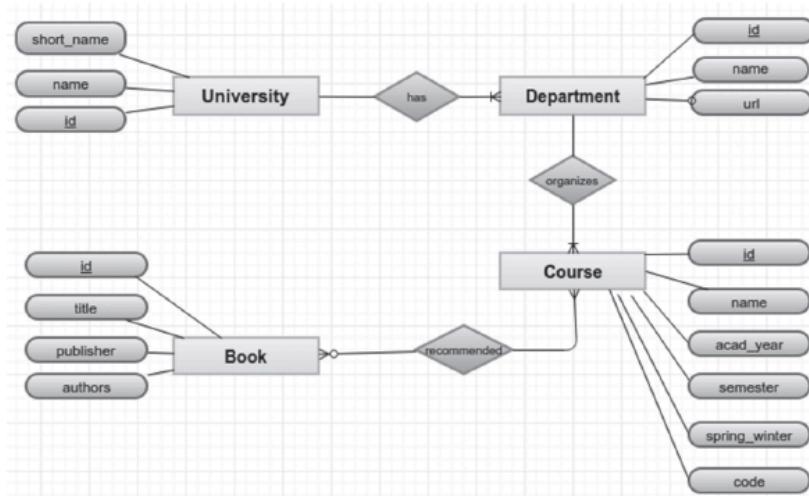


V3.3.1

Σύντομη εισαγωγή
στο σχεσιακό μοντέλο και την SQL

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

Σχεσιακές βάσεις δεδομένων: τα δεδομένα αποθηκεύονται σε συσχετιζόμενους πίνακες



Προγραμματίζω
με την python 

Structured Query Language (SQL) γλώσσα δημιουργίας και διαχείρισης δεδομένων σε σχεσιακή μορφή

Προγραμματίζω
με την python 

1. Δημιουργία πίνακα:

create table ονομα-πίνακα (γνωρίσμα, γνώρισμα, γνώρισμα);

2. Εισαγωγή δεδομένων στον πίνακα:

insert into όνομα-πινακα **values** (.., .., ..);

3. Διαγραφή δεδομένων από πίνακα:

delete from όνομα-πίνακα **where** κριτήριο διαγραφής;

4. Τροποποίηση δεδομένων πίνακα:

update όνομα-πίνακα **set** γνώρισμα = τιμή **where** κριτήριο επιλογής;

5. Ανάκτηση δεδομένων πίνακα ή πινάκων:

select γνωρίσματα **from** πίνακα ή πίνακες **where** κριτήρια ανάκτησης.

Σχεσιακές βάσεις δεδομένων

Προγραμματίζω
με την python



PostgreSQL



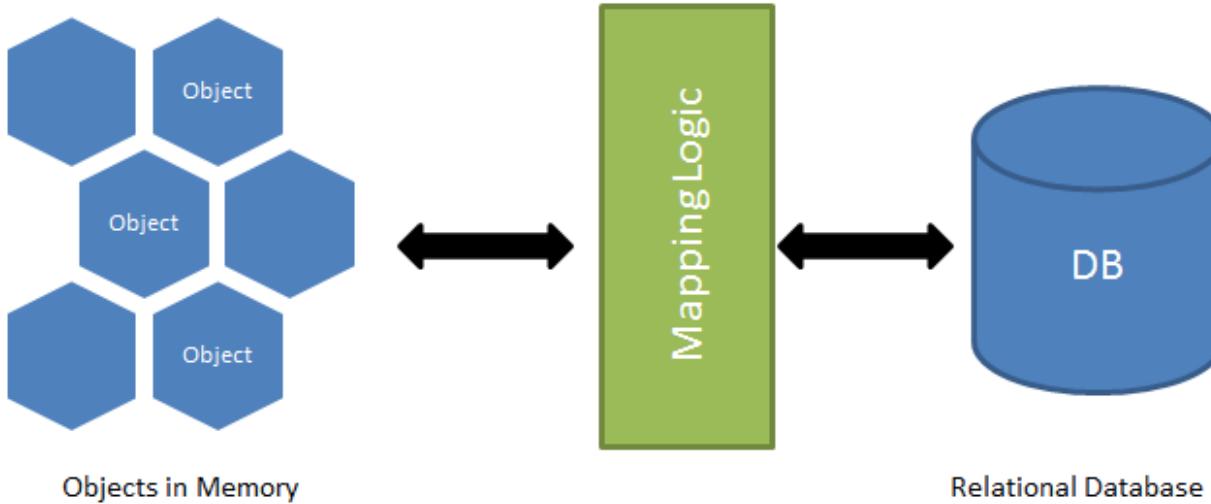
embedded data base

Ένα πρόβλημα: πώς θα αντιστοιχίσουμε τα αντικείμενα σε σχεσιακούς πίνακες

Προγραμματίζω
με την python



O/R Mapping



- SQLAlchemy
- Django ORM

web framework	None	Flask	Flask	Django
ORM	SQLAlchemy	SQLAlchemy	SQLAlchemy	Django ORM
database connector	(built into Python stdlib)	MySQL-python	psycopg	psycopg
relational database	SQLite	MySQL	PostgreSQL	PostgreSQL

Παράδειγμα εντολών sql στην python ανάκτηση από ένα πίνακα student

Προγραμματίζω
με την python 

```
def restore_data(self):
    try:
        conn = sqlite3.connect(self.file_name)
        curs = conn.cursor()
        curs.execute('select * from student;')
        for s in curs.fetchall():
            self.dd[s[0]] = list(s[1:])
    except: print('σφάλμα ανοιγματος αρχειου')
```

Προγραμματίζω
με την python



V3.3.2

Η βιβλιοθήκη sqlite3

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

```
import sqlite3 as lite
```

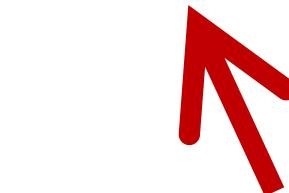
Προγραμματίζω
με την python 

- Απλή σχεσιακή βάση δεδομένων, που αποθηκεύεται σε απλό αρχείο. Δεν διαθέτει μηχανισμό για πολλαπλούς χρήστες και προσπελάσεις.
- Αναπτύχθηκε για ενσωματωμένες εφαρμογές, όπως Mozilla-Firefox, Symbian OS ή Android.
- Περιλαμβάνεται στις βασικές βιβλιοθήκες της python.
- Για να χρησιμοποιήσουμε μια βάση δεδομένων, πρέπει πρώτα να δημιουργήσουμε ένα αντικείμενο σύνδεσης `sqlite3.connect()`.
- Το αντικείμενο σύνδεσης θα αντιπροσωπεύει τη βάση δεδομένων. Το όρισμα της σύνδεσης λειτουργεί τόσο ως όνομα του αρχείου, όπου αποθηκεύονται τα δεδομένα, όσο και ως όνομα της βάσης δεδομένων.
- Αν υπάρχει ήδη αρχείο με αυτό το όνομα, θα ανοίξει. Πρέπει να είναι ένα αρχείο βάσης δεδομένων SQLite. Αν το αρχείο δεν υπάρχει, δημιουργείται.

sqlite3.connect()

παράδειγμα δημιουργίας πίνακα

```
try:  
    connection = sqlite3.connect("contacts.db")  
    with connection:  
        cursor = connection.cursor()  
        sql = 'create table contact(name text primary key, number text);'  
        cursor.execute(sql)  
        return True  
except sqlite3.Error as er :  
    print(er)  
    return False
```



name: όνομα στήλης (τύπου text)
primary key : μοναδικές τιμές

τύποι δεδομένων σε πίνακα της sqlite3

- **NULL.** τιμή null.
- **INTEGER.** προσημασμένος ακέραιος μεγέθους ως 8 bytes.
- **REAL.** πραγματικός αριθμός , αποθηκευμένος ως αριθμός 8-byte IEEE floating point.
- **TEXT.** συμβολοσειρά με χρήση της προκαθορισμένης κωδικοποίησης της βάσης δεδομένων (UTF-8, UTF-16BE or UTF-16LE).
- **BLOB.** (binary large object), ακολουθία από δυαδικά δεδομένα, αποθηκεύεται όπως έχει εισαχθεί στην εφαρμογή.

παράδειγμα διαγραφής στοιχείου από πίνακα

Προγραμματίζω
με την python 

```
try:  
    conn = lite.connect("contacts.db")  
    with conn:  
        curs = conn.cursor()  
        sql = "delete from contact where name = '{}';".format(id)  
        curs.execute(sql)  
except lite.Error as er:  
    print(er)
```



delete from Πίνακας where Συνθήκη ;

παράδειγμα εισαγωγής στοιχείου σε πίνακα

Προγραμματίζω
με την python 

```
try:  
    conn = lite.connect("contacts.db")  
    with conn:  
        curs = conn.cursor()  
        sql = "insert into contact values ('{}', '{}');"  
        curs.execute(sql.format(self.name, self.number))  
except lite.Error as er:  
    print(er)
```



insert into Πίνακας values (τιμές);

παράδειγμα τροποποίησης στοιχείου πίνακα

Προγραμματίζω
με την python 

```
try:  
    conn = lite.connect("contacts.db")  
    with conn:  
        curs = conn.cursor()  
        sql = 'update contact set number = "{}" where name =  
"{}";'.format(self.number, self.name)  
        curs.execute(sql)  
except lite.Error as er:  
    print(er)
```



update Πίνακα set ν=τιμή where συνθήκη;

παράδειγμα ανάκτησης στοιχείου/στοιχείων πίνακα

Προγραμματίζω
με την python 

```
try:  
    conn = lite.connect("contacts.db")  
    with conn:  
        curs = conn.cursor()  
        if term: # έχει δοθεί κλειδί αναζήτησης term  
            sql = "select * from contact where name like '%{}%'".format(term)  
        else:  
            sql = "select * from contact;"  
        curs.execute(sql)  
        records = curs.fetchall()  
        for rec in records:  
            Contact(rec[0], rec[1])  
except lite.Error as er: print(er)
```

χωρίς
συνθήκη

με συνθήκη

select κάτι from Πίνακα where συνθήκη;

Προγραμματίζω
με την python



V3.3.3

Παράδειγμα: η contacts με sqlite3

Νίκος Αβούρης, Πανεπιστήμιο Πατρών



contacts εκδ.4

Προγραμματίζω
με την python



- Η έκδοση 4 της εφαρμογής contacts ακολουθεί την δομή της έκδοσης 3b (shelve), αφού χειρίζεται εγγραφές και όχι όλη την πληροφορία.
- Θα πρέπει να προσέξουμε να δημιουργήσουμε την βάση δεδομένων πριν την χρησιμοποιήσουμε με εντολές `create table`

Προγραμματίζω
με την python



V3.3.4

Ανασκόπηση διαφορετικών προσεγγίσεων
μόνιμης αποθήκευσης δεδομένων

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

προσεγγίσεις που συζητήθηκαν

Προγραμματίζω
με την python 

1. Αρχείο χαρακτήρων (`open(f, 'w')`)
2. Δομημένο αρχείο χαρακτήρων (`csv`)
3. Σειριοποίηση πληροφορίας σε αρχείο bytes (`pickle`)
4. Οργάνωση πολλαπλών pickles με δείκτες (`shelve`)
5. Αρχείο με δομή JSON (`json`)
6. Σχεσιακή βάση δεδομένων (`sqlite3`)

Προγραμματίζω
με την python



v3.3.5

1^η εργασία αλληλο-αξιολόγησης

students

Νίκος Αβούρης, Πανεπιστήμιο Πατρών

περιγραφή προβλήματος

Προγραμματίζω
με την python 

1. Έστω μάθημα στο οποίο γράφονται φοιτητές. Καταχωρείται το όνομα, **επίθετο** και **αριθμός μητρώου** (ακέραιος) κάθε φοιτητή.
2. Κατά τη διάρκεια του εξαμήνου δίνονται 4 εργασίες.
3. Οι φοιτητές πρέπει να παραδώσουν τουλάχιστο 3 από τις 4 **εργασίες** και να πάρουν συνολικά τουλάχιστον 20 πόντους για να έχουν δικαίωμα συμμετοχής στην τελική εξέταση.
4. Αν δεν πάρουν προβιβάσιμο βαθμό (≥ 5) στην **τελική εξέταση**, έχουν δικαίωμα συμμετοχής στην **επαναληπτική εξέταση**.
5. Η **μη συμμετοχή** σε μια εξέταση ή **μη παράδοση κάποιας** εργασίας συμβολίζεται με τιμή -1.0.
6. Οι βαθμοί των εργασιών και των εξετάσεων είναι στην κλίμακα 0.0 μέχρι 10.0 με ακρίβεια 0.5.
7. Ο **Τελικός βαθμός**= $0.7 * \text{Εξέταση} + 0.3 * \text{μέση βαθμολογία ασκήσεων}$, αν ο βαθμός εξέτασης ≥ 5 , αλλιώς ο τελικός βαθμός είναι ο βαθμός της τελικής εξέτασης.

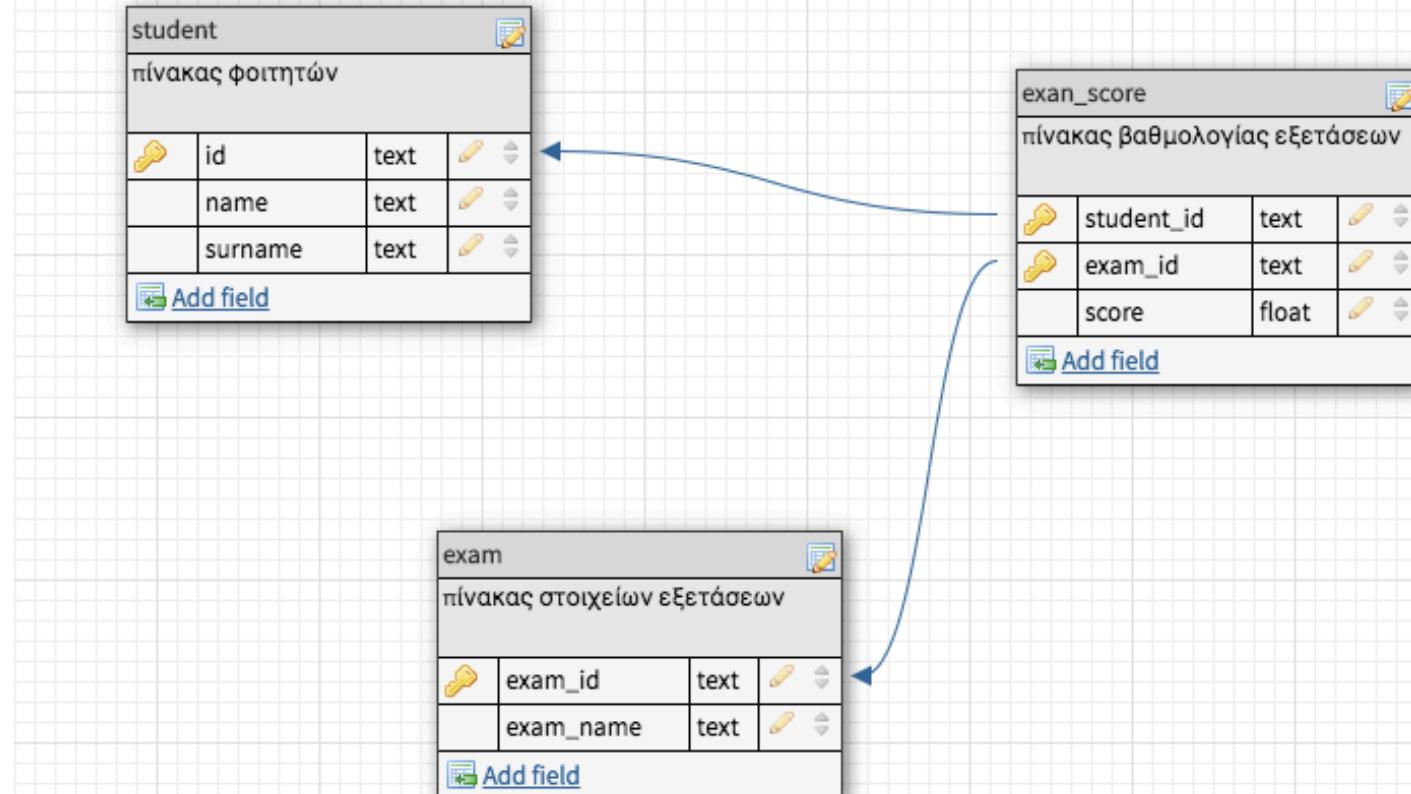
περιγραφή λύσης

Προγραμματίζω
με την python 

1. Ζητείται να δημιουργηθεί εφαρμογή python που δημιουργεί δεδομένα μιας τάξης (ονόματα φοιτητών και βαθμολογίες) σύμφωνα με τους κανόνες του μαθήματος
2. Αποθηκεύει και ανακτά τους φοιτητές και τις βαθμολογίες τους στη βάση δεδομένων students.sqlite3.
3. Παρουσιάζει τον αλφαριθμητικό κατάλογο των φοιτητών με τη λεπτομερή βαθμολογία τους στις εργασίες και τις εξετάσεις
4. Παρουσιάζει τον αλφαριθμητικό κατάλογο των φοιτητών με τις τελικές τους βαθμολογίες
5. Παρουσιάζει τη μέση βαθμολογία και ποσοστό επιτυχίας για κάθε μια από τις δύο εξετάσεις (τελική και επαναληπτική).

σχεδίαση της βάσης δεδομένων

Πίνακες:
student
exam
exam_score



Το σχήμα της βάσης δεδομένων δημιουργήθηκε με το εργαλείο dbdesigner.net

Προγραμματίζω
με την python



σχεδίαση της βάσης δεδομένων

Πίνακες:

student

exam

exam_score

The screenshot shows the DB Browser for SQLite interface. On the left, the 'Database Structure' tab is selected, displaying a tree view of tables and indices. The tables are: exam, exam_score, and student. The exam table has columns exam_id (TEXT), exam_name (TEXT). The exam_score table has columns student_id (TEXT), exam_id (TEXT), score (REAL). The student table has columns id (TEXT), name (TEXT), surname (TEXT). Indices listed are: sqlite_autoindex_exam_1, sqlite_autoindex_exam_score_1, sqlite_autoindex_student_1, and student_index. On the right, a large block of SQL code is shown, which creates these three tables and a primary key constraint on student_index.

```
1 BEGIN TRANSACTION;
2 CREATE TABLE "student" (
3     `id` TEXT,
4     `name` TEXT,
5     `surname` TEXT,
6     PRIMARY KEY(`id`)
7 );
8 CREATE TABLE "exam_score"
9     `student_id` TEXT,
10    `exam_id` TEXT,
11    `score` REAL,
12    PRIMARY KEY(`student_id`, `exam_id`)
13 );
14 CREATE TABLE "exam" (
15     `exam_id` TEXT,
16     `exam_name` TEXT,
17     PRIMARY KEY(`exam_id`)
18 );
19 CREATE INDEX `student_index` ON `student` (`id`);
20 COMMIT;
```

Δημιουργούμε τη βάση
δεδομένων στο εργαλείο DB
Browser for SQLite
το οποίο παράγει ένα αρχείο sql

Προγραμματίζω
με την python



πρόσθετες εντολές
BEGIN TRANSACTION; COMMIT; για
ολοκλήρωση της δοσοληφίας στη
βάση δεδομένων
εντολή CREATE INDEX xx ON πίνακας
για δημιουργία ευρετηρίου

δεδομένα

Προγραμματίζω
με την python 

δίδονται τα εξής αρχεία:

- main.py (κλάση Main, και κλάσεις Student, Create) ---ΣΗΜΕΙΩΣΗ: Στη διάλεξη γίνεται αναφορά σε δύο αρχεία main kai student, τελικά ενοποιήθηκαν σε ένα για διευκόλυνσή σας
- students.sqlite.sql (εντολές SQL δημιουργίας της βάσης δεδομένων student)

Επίσης δίνονται τα βοηθητικά αρχεία actors.txt και actresses.txt με ονόματα ηθοποιών για να χρησιμοποιηθούν ως ονόματα φοιτητών

ερώτημα 1. δημιουργία τάξης

Προγραμματίζω
με την python 

Έχει υλοποιηθεί ο περισσότερος κώδικας
ζητείται να υλοποιήσετε τη μέθοδο final_exams_check της
κλάσης Student που ελέγχει αν ο φοιτητής επιτρέπεται να
λάβει μέρος στην τελική εξέταση (επιστρέφει True/False)

μπορείτε να πειραματιστείτε με το αρχείο main.py

ερώτημα 2. αποθήκευση στη βάση δεδομένων

Προγραμματίζω
με την python 

Έχει υλοποιηθεί ο περισσότερος κώδικας στο αρχείο main.py

Ζητείται να υλοποιήσετε τη μέθοδο `create_database` της κλάσης Main

ερώτημα 3. αλφαβητικός κατάλογος βαθμολογιών

Προγραμματίζω
με την python 

Να υλοποιήσετε τη μέθοδο question_2 της κλάσης
Main καθώς και τη μέθοδο order_students_list της
κλάσης Student

ερώτημα 4. τελική βαθμολογία

Προγραμματίζω
με την python 

Να υλοποιήσετε τη μέθοδο question_3 της κλάσης Main καθώς και τη μέθοδο success_rate της κλάσης Student που υπολογίζει τα τελικά ποσοστά επιτυχίας στο μάθημα

ερώτημα 5. στατιστικά εξετάσεων

Προγραμματίζω
με την python 

Να υλοποιήσετε τη μέθοδο question_4 της κλάσης Main με χρήση αποκλειστικά της βάσης δεδομένων.

Προσοχή στο ερώτημα αυτό δεν θα πρέπει να χρησιμοποιηθεί η κλάση Student

σημείωση

Προγραμματίζω
με την python 

Μπορείτε να θέσετε τη σταθερά VERBOSE = True
ώστε να έχετε καλύτερη εικόνα της εκτέλεσης του προγράμματος και
να την χρησιμοποιήσετε για εκτύπωση μεταβλητών και break points

```
πχ if VERBOSE: print( ...) #εκτύπωση τιμών μεταβλητών
```

```
if VERBOSE: input() # break point
```