

# BAZE PODATAKA

*Progress isn't made by early risers.  
It's made by lazy men trying to find easier ways to do something.*

## Uvodni primer

Studenti prve godine informatike su došli na ideju da naprave program pomoću koga bi bilo moguće voditi evidenciju o knjigama u fakultetskoj biblioteci.

Zaključili su da bi trebalo čuvati **informacije o:**

- autorima** - neki spisak, bar, imena bi morao da bude dostupan
- knjigama** - ISBN, naziv, autori.

Od funkcionalnosti koje bi program trebao da zadovolji odlučili su se za sledeće:

- dodavanje, brisanje autora, kao i izmena imena
- brisanje i dodavanje knjige u evidenciju
- dodavanje podataka o tome koji je autor koju knjigu napisao, izmena i brisanje istih
- pretraga autora po knjigama - zadavanjem imena ili ISBN-a knjige dobija se spisak autora
- pretraga knjiga po naslovu sa i bez zadavanja autora
- pretraga autora po imenu sa spiskom knjiga koje je napisao

**Trebalo je odlučiti:**

- u kakve strukture smestiti podatke
- da li podatke držati u dinamičkim strukturama dok program radi pa ih na kraju rada izmenjene upisati u datoteke ili učitavati po potrebi

Nakon nekog vremena želje su počele da rastu pa se i kompleksnost problema menjala, npr.

- trebalo je ubaciti i godinu izdavanja
- broj zapisa je porastao na 200 000 različitih knjiga
- pojavili su se različiti autori sa istim imenima
- program su počeli da koriste i službenici kojima preciznost nije jača strana pa su više puta unosili podatke o istoj knjizi, a ime jednog istog autora zapisivali u različitim knjigama na različit način

## ZADATAK ZA VAS

Osmislite sami strukture, odlučite kako bi čuvali podatke u toku rada programa i u zavisnosti od toga napravite spisak f-ja sa skicom njihove signature i opisom šta bi radile.

Kakve bi izmene u vašem programu zahtevalo ispunjavanje dodatnih želja?

Na to dodajte još par talasa dodatnih želja i spisak problema koje su napravili raspoloženi službenici i pre ili kasnije stižete do želje da kod više nikad ne pogledate.

# KAKO BI TO IZGLEDALO U SLUČAJU DA KORISTIMO NEKI SOFTVER ZA UPRAVLJANJE BAZAMA PODATAKA

U bazi će podaci biti čuvani u tabelama, npr. ovakvim

Autor:

autorID	autor
1	T. Codd
2	P. Norvig
▶*	NULL

Knjiga:

ISBN	naslov
N0-01	Artificial Intellige...
N0-02	Relational Datab...
▶*	NULL

KA:

autorID	ISBN
1	N0-02
2	N0-01
▶*	NULL

Tabele su deo baze podataka **Biblioteka** o kojoj "brine" sistem za upravljanje bazama podataka - SUBP (npr. SQL Server, MySQL, Oracle i sl.). Isti sistem mora da vam omogući mehanizam upisa, brisanja i dodavanja novih podataka kao i dobijanja odgovora na pitanja koja se tiču podataka upisanih u bazu. "Razgovor" sa SUBP-om o tome šta želite da uradi se odvija na posebnom jeziku Structured Query Language - SQL, čije komande SUBP jeste u stanju da procesira. Dakle, umesto pisanja f-ja koje će manipulirati podacima po datotekama posao se svodi na zadavanje SQL komandi. Neki primeri su dati u tabeli:

dodavanje/	<code>insert autori values (3, 'T. Berners-Lee')</code>
brisanje autora/	<code>delete autori where autorID=3</code>
izmena imena autora	<code>update autori set autor='T.B.Lee' where autorID=3</code>
pretraga autora po knjigama - zadavanjem ISBN-a knjige dobija se spisak autora	<code>select * from autori where autorID in (select autorID                   from KA                   where ISBN='N0-01')</code>
dodavanje novog podatka u tabelu Knjige	<code>ALTER TABLE dbo.Knjige ADD                   godina numeric(4, 0) NULL</code>

Obezbeđivanje jedinstvenosti i brze pretrage po, npr. ISBN-u, je moguće obezbediti postavljanjem ograničenja na to polje, npr.

```
ALTER TABLE dbo.KA ADD CONSTRAINT
    IX_KA UNIQUE NONCLUSTERED ( autorID ) ON [PRIMARY]
```

Tada će SUBP voditi računa o jedinstvenosti. Pored ove postoji puno različitih vrsta ograničenja o kojima sistem može voditi računa, a o kojima bi inače brinule vaše f-je (na prethodnom primeru za upis knjiga - ako uopšte imate želju da zaštitite službenike od nenamernih grešaka, odnosno f-je pretrage bi morale da imaju vrlo brze algoritme koji traže i pogodne strukture u kojima bi se traženi podaci nalazili) .

Dakle, pojednostavljeno predstavljeno vaše je da:

- osmislite kako će podaci biti grupisani (po tabelama),
- popišete ograničenja,
- naučite da govorite SQL jezikom i ono što ste osmislili kažete SUBP-u da napravi, održava i
- da kada vam je potrebno postavite i po neko pitanje istom

Kako će podaci fizički biti zapisani, kako će biti brz obezbeđen odgovor na vaša pitanja, kako će vaša ograničenja biti obezbeđena i sl. nije vaš problem.