3

MODELI I STRUKTURE PODATAKA

U ovom poglavlju razmotrićemo dva pojma koji su od velikog značaja za deo informatike koji se bavi informacionim sistemima i bazama podataka. Prvi od njih, model podataka, je u samim osnovama te oblasti, dok je drugi, strukture podataka, značajan pošto se odnosi na organizaciju i način zapisivanja podataka.

3.1 Pojam modela podataka

Reč "model" se uvek javlja zajedno sa nečim, u smislu "model nečega" To "nešto" nazivamo originalom. Model podataka je samo specijalni slučaj modela, pa je uputno da prvo razmotrimo opšti pojam originala i modela i postupke koji se primenjuju pri sastavljanju modela, odnosno modeliranju.

3.1.1 Original, model i neki opšti pojmovi

Pod modelom se podrazumeva predstava nekog originala. Original je najčešće deo našeg okruženja i kao takav realan, ali može biti i abstraktan.

Svaki original, a posebno ako je deo našeg okruženja, možemo smatrati za sistem koji čine izabrani objekti i odnosi između njih. Sistem opisujemo preko odgovarajućih skupova objekata i skupova odnosa između njih, ali pri tome uvek mora postojati mogućnost raspoznavanja individualnih objekata i odnosa u tim skupovima.

Na osnovu ovih zapažanja, u mogućnosti smo da u vezi predstavljanja nekog sistema definišemo nekoliko opštih pojmova:

- entitet: opšti pojam za nešto što postoji i što se može jednoznačno odrediti (prepoznati) u skupu srodnih entiteta; ovako definisan, entitet obuhvata kao specijalne slučajeve pojmove objekta i veze kao posebnog vida odnosa;
- svojstvo: imenovana osobina entiteta, podrazumeva vrstu (ime) svojstva i vrednost svojstva (na primer, "boja" i "plavo"); entitete opisujemo pomoću jednog ili (najčešće) više izabranih svojstava;
- identifikator: svojstvo ili skup svojstava koji svojom vrednošću jednoznačno određuju svaki pojedinačni objekat ili odnos u skupu (ne mogu postojati dva elementa skupa sa istim vrednostima identifikatora);
- klasa: imenovani skup entiteta koji se odlikuju istim vrstama svojstava (na primer, klasa "knjiga" u koju ulaze sve pojedinačne knjige koje se odlikuju svojstvima "šifra" (tzv. inventarni broj) i "naslov"; svojstva koja odlikuju neku klasu nazivamo klasifikacionim svojstvima;
- instanca: element u klasi, entitet, pojedinačni objekat ili odnos.

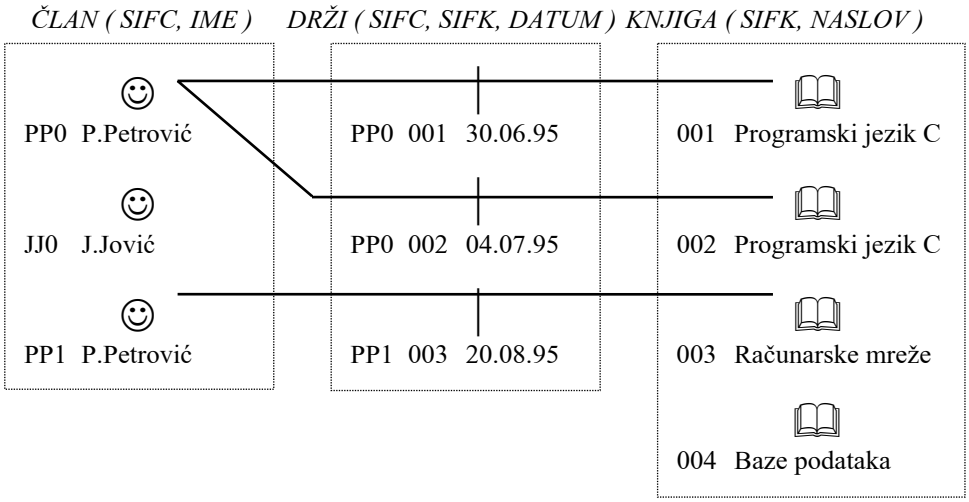
Pomoću prethodno navedenih pojmova, možemo da preciziramo šta sve karakteriše neki sistem, odnosno original:

- skup svih klasa objekata;
- skup svih klasa odnosa između objekata;
- klasifikaciona svojstva za svaku klasu objekata (najmanje jedno svojstvo, radi raspoznavanja pojedinačnih objekata);
- klasifikaciona svojstva za svaku klasu odnosa (najmanje onoliko identifikatora objekata koliko učestvuje u odnosu, radi raspoznavanja pojedinačnih odnosa);
- instance za svaku klasu objekata (pojedinačni objekti, koji se odlikuju određenim vrednostima klasifikacionih svojstava);
- instance za svaku klasu odnosa (pojedinačni odnosi, koji se odlikuju određenim vrednostima klasifikacionih svojstava).

Pri tome, posmatrani sistem je najčešće dinamički, odnosno promenljiv sa vremenom. Promene mogu biti sledeće:

- nastajanje ili nestajanje instance objekta u nekoj klasi objekata;
- nastajanje ili nestajanje instance odnosa u nekoj klasi odnosa;
- promena vrednosti jednog ili više svojstava instance objekta u nekoj klasi objekata;
- promena vrednosti jednog ili više svojstava instance odnosa u nekoj klasi odnosa.

Kao ilustracija prethodnog poslužiće nam krajnje pojednostavljeni prikaz stanja sistema BIBLIOTEKA sa jednim odnosom tipa veze, dat na slici 3-1.



Slika 3-1: Jedno stanje sistema BIBLIOTEKA.

Na ovoj ilustraciji su navedeni nazivi klasa i u zagradama klasifikaciona svojstva, a uz svaku instancu naznačene su vrednosti klasifikacionih svojstava. Napomenimo još i sledeće:

- u klasi *CLAN* smo zbog mogućnosti da članovi imaju isto ime, što je i slučaj, uveli pored svojstva *IME* i dodatno identifikaciono svojstvo "šifra člana" *SIFC*;
- sličnu situaciju imamo i u klasi *KNJIGA* gde smo zbog ponavljanja vrednosti svojstva *NASLOV* uveli identifikator "šifra knjige" (inventarni broj) *SIFK*;
- u klasi veze *DRZI* minimum svojstava bi bili identifikatori učesnika u vezi *SIFC* i *SIFK*, ali smo pored toga uveli i svojstvo *DATUM* koje govori o tome od kada član drži knjigu kod sebe.

Navedimo nekoliko karakterističnih primera promena u odnosu na stanje sistema BIBLIOTEKA prikazano na slici 3-1:

- nabavka nove knjige naslova 'PASCAL programiranje": u klasi *KNJIGA* nastaje nova instanca sa parom vrednosti klasifikacionih svojstava 005, PASCAL programiranje;
- član šifre PP1 vraća knjigu šifre 003 koju je držao kod sebe: iz klase *DRŽI* nestaje instanca sa skupom vrednosti klasifikacionih svojstava PP1, 003, 20.08.95;
- promena prezimena člana šifre JJ0: u klasi *ČLAN* instanca sa parom vrednosti klasifikacionih obeležja JJ0, J.Jović menja drugu vrednost u novu.

Već smo naglasili da je svaki model predstava nekog originala. Pri tome, postoji jedna vrlo bitna osobina modela: sredstvo predstavljanja. Navedimo nekoliko primera za to:

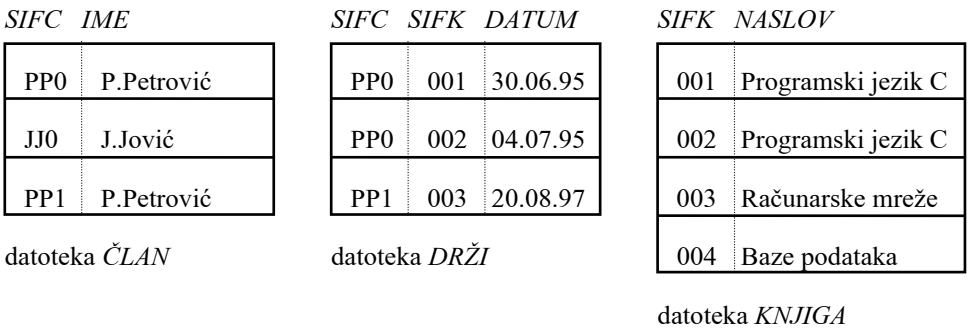
- plastična maketa aviona: ovo je grub model koji predstavlja samo oblik i spoljni izgled originala, a način predstavljanja je pomoću tela odgovarajućeg oblika ali umanjenog u odnosu na original;
- planovi za izradu aviona: ovo je detaljan model originala, pošto sadrže predstavu većine njegovih svojstava, a način predstavljanja je pomoću teksta i slika;
- neka teorija u fizici: ovo je manje ili više detaljan model jednog odabranog segmenta prirode, a način predstavljanja je preko teksta i matematičkih formula;
- kartotečki informacioni sistem biblioteke: ovo je detaljan model stvarne biblioteke, a način predstavljanja je preko podataka zapisanih na karticama;
- računarski informacioni sistem biblioteke: ovo je model stvarne biblioteke sa dovoljnim nivoom detalja za efikasan rad biblioteke, a način predstavljanja je preko podataka zapisanih na magnetnim disk jedinicama.

Iz navedenih primera uočavamo da podaci mogu biti sredstvo predstavljanja, odnosno da je informacioni sistem model nekog sistema.

3.1.2 Informacioni sistem kao model

U opštem smislu, podatak je predstava o "nečemu" izražena nekim jezikom (dogovorenim načinom opštenja). Taj jezik ne mora nužno biti govorni jezik. Primera radi, podatak o nekoj visini predstavićemo korišćenjem cifara decimalnog brojnog sistema ako ga zapišemo na papiru, ali će isti taj podatak u računaru biti predstavljen u binarnoj formi, na najednostavnijem mogućem jeziku koji raspolaže sa samo dva simbola - 0 i 1. Na sličan način, preko grupa simbola 0 ili 1 možemo predstaviti pojedina slova i reči govornog jezika.

Informacioni sistem kao skup podataka o nekom sistemu predstavlja model tog sistema. Ilustrujmo to prikazom informacionog sistema naše biblioteke koja je kao sistem prikazana na slici 3-1.



Slika 3-2: Podaci informacionog sistema BIBLIOTEKA.

Iz ove ilustracije možemo jasno sagledati kako informacioni sistem predstavlja model nekog realnog sistema:

- svakoj klasi objekta ili veze odgovara jedna datoteka slogovnog tipa;
- svakom klasifikacionom svojstvu neke klase objekta ili veze odgovara jedan podatak u sastavu sloga odgovarajuće datoteke;
- svakoj instanci u nekoj klasi objekata ili veza odgovara jedan slog u odgovarajućoj datoteci;
- svakoj vrednosti klasifikacionog svojstva neke instance u nekoj klasi objekata ili veza odgovara jedna vrednost odgovarajućeg podatka u odgovarajućem slogu odgovarajuće datoteke.

Sadržaj svih datoteka informacionog sistema sa slike 3-2 odgovara stanju sistema BIBLIOTEKA prikazanom na slici 3-1. Kako se sa vremenom menja stanje sistema BIBLIOTEKA, tako i informacioni sistem BIBLIOTEKA mora da prati te promene da bi bio valjan model originala. Da bi to pojasnili, navedimo kako bi se ranije navedene promene za sistem BIBLIOTEKA odrazile na naš informacioni sistem:

- nabavka nove knjige naslova “PASCAL programiranje": u datoteci *KNJIGA* se dodaje novi slog sa vrednostima podataka 005, PASCAL programiranje;
- član šifre PP1 vraća knjigu šifre 003 koju je držao kod sebe: iz datoteke *DRŽI* se uklanja slog sa vrednostima podataka PP1, 003, 20.08.95;
- promena prezimena člana šifre JJ0: u datoteci *ČLAN* u slogu sa parom vrednosti podataka JJ0, J.Jović menja se vrednost drugog podatka.

Pri tome, sve izmene u podacima neophodne da bi se pratilo stanje sistema koji se modelira obezbeđuje druga komponenta koju smo naveli u definiciji informacionog sistema, a to je skup postupaka za održavanje podataka.

3.1.3 Baza podataka kao model

Baza podataka je specijalni, i to najsavršeniji vid komponente "Podaci" informacionog sistema, pa samim tim dosta toga što smo rekli o informacionom sistemu kao modelu nekog sistema može da se primeni i na bazu podataka. Međutim, baza podataka je u svojstvu modela tu u značajnoj prednosti. Konkretno:

- zbog integrisanosti podataka baza podataka vernije predstavlja originalni sistem; ono što je u stvarnosti jedna celina predstavljeno je kao celina i u bazi podataka;
- zbog organizacije podataka prema potrebama korisnika baza podataka je u mogućnosti na integrisane podatke kao model "preslika" na onoliko parcijalnih modela koliko to treba pojedinim korisnicima; jednom istom sistemu može odgovarati više modela sa manjim ili većim nivoom detalja.

3.1.4 Postupci modeliranja

U poglavlju 1, kada smo naveli primer biblioteke kao sistema, u obzir smo uzeli samo neke od postojećih skupove objekata i veza između njih. Konkretno, opredelili smo se za tretman sistema BIBLIOTEKA preko sledećih skupova klasa:

Objekti = { {naslov} , {knjiga} , {član} , {autor} }

Veze = { {drži} , {sadrži} , {je_autor} }

Međutim, kada smo u ovom poglavlju pojam originala sveli na pojam sistema, opredelili smo se za jednostavniji tretman biiblioteke, preko samo dve klase objekata i jedne klase veze:

Objekti = { {knjiga} , {član} }

Veze = { {drži} }

Uz to, naglasimo da smo za svaku pojedinu klasu vršili izbor klasifikacionih svojstava. Primera radi, umesto da za klasu član uzmemo detaljnu predstavu

ČLAN (SIFC, IME, POL, STAROST)

opredelili smo se za manji broj svojstava, odnosno za predstavu

ČLAN (SIFC, IME)

Postupak koji smo upravo naveli, zanemarivanje nebitnih i uzimanje u obzir samo bitnih klasa i svojstava, naziva se postupkom abstrakcije i on čini suštinu postupka razvoja svakog modela. Međutim, pre toga smo morali da uočimo klase, odnosno to da pojedinačna pojavljivanja entiteta imaju nešto zajedničko i da ih treba posmatrati kao elemente određenih klasa. Taj postupak se naziva klasifikacija. Obrnuto, ako imamo već definisane klase a treba da uočimo da su neki pojedinačni entiteti elementi tih klasa, u pitanju je postupak instancizacije.

Na osnovu prethodnog u mogućnosti smo da formulišemo osnovne postupke formiranja modela i njihov redosled:

- klasifikacija objekata: uočava se koji pojedinačni objekti čine koje klase objekata;
- klasifikacija odnosa: uočava se koji pojedinačni odnosi čine koje klase odnosa;
- abstrakcija objekata: zanemaruju se nebitne klase objekata i za dalji postupak zadržavaju se samo one od značaja;
- abstrakcija odnosa: od preostalih klasa odnosa (otpali su svi oni koji uključuju klase objekata koje smo zanemarili u prethodnom koraku) zanemaruju se nebitni i za dalji postupak zadržavaju samo oni od značaja;
- abstrakcija klasifikacionih svojstava objekata: za svaku klasu objekata zanemaruju se nebitna i uzimaju samo značajna klasifikaciona svojstva;
- abstrakcija klasifikacionih svojstava odnosa: za svaku klasu odnosa zanemaruju se nebitna i uzimaju (ako ih ima) samo značajna dodatna klasifikaciona obeležja (identifikatori za klase objekata koji su u odnosu automatski postaju klasifikaciona obeležja odnosa).

Ovi postupci su jasno zastupljeni kada se vrši modeliranje pomoću podataka, dok su u drugim slučajevima manje izraženi.

3.1.5 Model podataka

Informacioni sistem, odnosno podaci u sklopu njega, predstavlja model nekog sistema. Pri tome, korespondencija između takvog modela i originala je dvojaka:

- podaci svojim ustrojstvom odgovaraju ustrojstvu sistema; konkretno, datoteke i podaci unutar njihovih slogova moraju odgovarati klasama i svojstvima sistema, odnosno onom što se često naziva strukturom sistema;
- podaci svojim sadržajem odgovaraju stanju sistema; konkretno, slogovi u datotekama moraju odgovarati instancama u klasama, a vrednosti podataka u slogovima vrednostima svojstava tih instanci.

Da bi ostvarili navedenu korespondenciju, prvo moramo da kreiramo prazne datoteke sa odgovarajućim strukturama slogova, a zatim da u te datoteke unesemo podatke koji odgovaraju nekom početnom stanju našeg sistema. Od tog trenutka, zahvaljujući postupcima održavanja, takav informacioni sistem će svojim sadržajem pratiti stanje sistema koga predstavlja.

U vezi upravo opisanog početka svakog informacionog sistema, kao prirodno se nameće sledeće pitanje: *kako* znamo koje datoteke i kakve strukture treba da kreiramo? Očigledno je da moramo imati neku predstavu o ustrojstvu podataka u okviru budućeg informacionog sistema. Ta predstava o podacima, formulisana nekim pogodnim sredstvom izražavanja, jeste ono što nazivamo modelom podataka.

Model podataka je predstava ustrojstva podataka za neki informacioni sistem, odnosno bazu podataka. Ta predstava mora da sadrži odgovore na sledeća pitanja:

- koje datoteke ulaze u sastav informacionog sistema;
- koji podaci ulaze u sastav slogova svake od tih datoteka;
- kojih su tipova podataka svi ti podaci.

Navedeni odgovori nisu dovoljni sami za sebe, pošto ništa ne govore o eventualnim ograničenjima nad podacima i o načinu njihovog održavanja i korišćenja. Da bi bio sveobuhvatan, svaki model podataka mora da čine tri komponente:

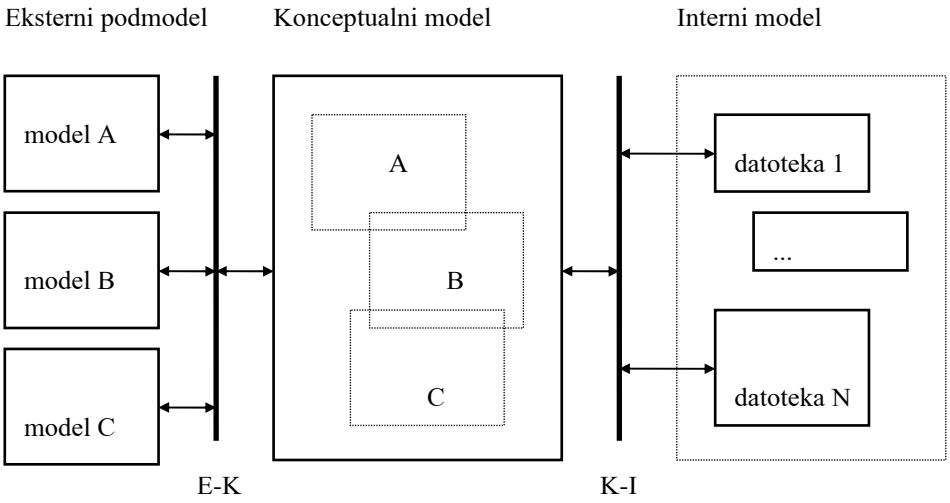
- strukturna komponenta: deo modela podataka koji smo već naveli, služi za predstavljanje ustrojstva podataka;
- integritetska komponenta: deo modela podataka koji služi za predstavljanje ograničenja nad podacima koja proizilaze iz ograničenja koja važe u originalnom sistemu (na primeru biblioteke, ograničenja su: knjiga ne može biti kod nepostojećeg člana, knjiga može biti ili kod ni jednog ili kod jednog člana, itd.);
- manipulativna komponenta: deo modela podataka koji služi za predstavljanje elementarnih postupaka održavanja i korišćenja podataka preko kojih se mogu opisati sve konkretne situacije u sistemu-okruženju.

Od navedenih komponenti kao posebno značajnu treba izdvojiti strukturnu. To je sasvim razumljivo ako se ima na umu da ustrojstvo podataka neposredno utiče na ograničenja i manipulacije nad podacima. Uz to, na osnovu našeg uvodnog primera škole u poglavlju 1 možemo naslutiti složenu prirodu ustrojstva podataka, koje treba da obuhvati ustrojstvo sa gledišta krajnjih korisnika, sveukupno ustrojstvo i ustrojstvo na odgovarajućim medijumima (uglavnom magnentnim diskovima).

Pokazalo se da je razdvajanje navedenih delova strukturne komponente modela podataka osnovni način za ostvarivanje maksimalne nezavisnosti programa i podataka, pa je vrlo brzo (1975) formulisan i standard o tome. Taj standard, poznat pod oznakom ANSI/X3/SPARC, obuhvata tri strukturna podmodela, i to:

- eksterni podmodel: čini ga skup korisničkih modela, odnosno predstava ustrojstva podataka sa gledišta pojedinih korisnika; ovom odgovara pojam podšeme koji smo objasnili u poglavlju 2;
- konceptualni podmodel: čini ga predstava sveukupnog ustrojstva podataka, i kao takav sadrži sve eksterne podmodele integrisane u celinu; tome odgovara ranije uvedeni pojam šeme
- interni podmodel: čini ga detaljna predstava načina zapisa sveukupnih podataka na odgovarajućim medijumima (uglavnom magnetnim diskovima).

Kao ilustracija za prethodno, neka nam posluži slika 3-3.



Slika 3-3: Strukturni podmodeli podataka.

Ova ilustracija na posredan način uključuje i sistem upravljanja bazom podataka (SUBP). Naime:

- granicu E-K, odnosno funkciju preslikavanja u oba smera na relaciji eksterni podmodel-konceptualni model, obezbeđuje SUBP; u cilju maksimalne nezavisnosti programa i podataka, ta granica treba da je što čvršća i kao takva jedini posrednik između navedena dva podmodela;
- granicu K-I, odnosno preslikavanje u oba smera na relaciji konceptualni podmodel-interni podmodel, takođe obezbeđuje SUBP; ta granica takođe treba da je što čvršća u cilju maksimalne nezavisnosti programa i podataka.

3.1.6 Vrste modela podataka

Podela na vrste modela podataka je odraz podele na vrste baza podataka, pri čemu je suštinska razlika između pojedinih vrsta modela u strukturnoj komponenti. Tako, imamo sledeće vrste modela podataka (izuzimajući objektni):

- hijerahijski model podataka: jedini način predstavljanja odnosa između podataka je preko odnosa "jedan prema više", odnosno 1:N; sredstva predstavljanja su skupovi slogova, slogovi, podaci u slogovima i ukazatelji na slogove;
- mrežni model podataka: jedini način predstavljanja odnosa između podataka je preko odnosa "više prema više", odnosno M:N, što kao specijalni slučaj uključuje i odnos 1:N; sredstva predstavljanja su ista kao i kod hijerahijskog modela - skupovi slogova, slogovi, podaci u slogovima i ukazatelji na slogove;
- relacioni model podataka: jedini način predstavljanja odnosa između podataka su relacije, odnosno datoteke sa slogovima, i njihov sadržaj; sredstva predstavljanja su u suštini samo skupovi slogova, slogovi i podaci, odnosno tzv. šeme relacija.

Prva dva modela se odnose na formatizovane (pokazivačke) baze podataka koje smo pomenuli na kraju poglavlja 1.

Razlike u strukturnoj komponenti uslovile su i značajne razlike u ostalim dvema komponentama navedenih vrsta modela podataka. To je posebno izraženo kod manipulativne komponente:

- hijerahijski a posebno mrežni model podataka podrazumevaju složene operacije održavanja i korišćenja podataka, kao što su: "nađi početak liste", "nađi kraj liste", "nađi prethodnika", "nađi sledbenika", "nađi roditelja u stablu", "nađi prvog potomka u stablu", "ubaci u listu", "izbaci iz liste", itd;
- relacioni model podataka zbog svoje jednostavnosti predstavljanja podataka iziskuje samo tri elementarne operacije održavanja i pet elementarnih operacija korišćenja podataka (o tome će biti više reči u narednom poglavlju); sve moguće manipulacije nad relacionom bazom podataka mogu se izraziti preko navedenog skupa elementarnih operacija.

Sve tri navedene vrste modela spadaju u grupu tzv. semantički nižih modela podataka, pošto uključuju i internu strukturnu komponentu koja je neposredno povezana sa realizacijom baze podataka u smislu fizičkog zapisa podataka na odgovarajućim medijumima. Modeli podataka koji su semantički višeg nivoa su pre mešavina modela sistema i modela podataka nego "čisti" modeli podataka, i kao takvi služe za prvu fazu modeliranja prilikom projektovanja neke baze podataka. Pri tome je za takav model poželjna osobina jednoznačnog prevođenja u neku nižu formu, najčešće relacioni model. Jedan takav model višeg nivoa, tzv. "model objekata i odnosa", izložen je u poglavlju 8.

3.2 Osnovni pojmovi iz struktura podataka

Pod strukturama podataka se podrazumeva deo informatike koji se bavi organizacijom podataka, načinom njihovog zapisivanja, kao i postupcima pretraživanja podataka. U ovom odeljku osvrnućemo se samo na neke od pojmova iz te oblasti.

3.2.1 Podatak, slog i datoteka

Pojmove kao što su podatak, slog i datoteka smo već koristili kada smo govorili o modelu podataka. I pored toga što su nam ti pojmovi uglavnom poznati i jasni sami po sebi, nije na odmet da ih i formalno definišemo:

- podatak: "šifrirano" svojstvo, predstava nekog svojstva izražena nekim jezikom (u informatici je to binarni jezik, tako da se podatak javlja u vidu niza bitova, od kojih svaki ima vrednost 0 ili 1);
- slog: celina sastavljena od jednog ili više podataka u određenom redosledu (celina u smislu manipulacije, odnosno čitanja i pisanja);
- datoteka: u najopštijem smislu, bilo koji podaci zapisani na nekom medijumu i koji kao celina nose zajedničko ime.

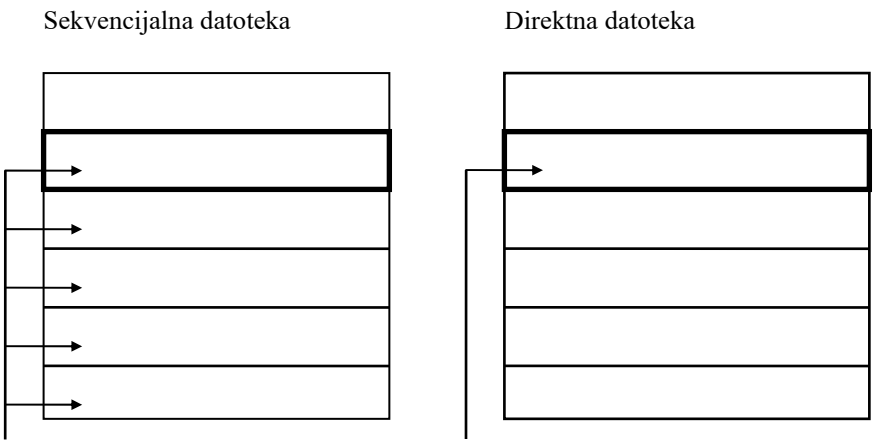
Datoteke možemo klasifikovati na više načina. Prva klasifikacija je prema organizaciji podataka u datoteci i po tom osnovu postoje dve vrste datoteka:

- neslogovne datoteke: datoteke u kojima podaci predstavljaju celinu koja se ne može rastavljati na delove (slogove); primeri za takve datoteke su zapisi crteža, slika i slično;
- slogovne datoteke: datoteke koje se sastoje iz slogova, pri čemu slogovi mogu biti iste ili različite strukture (u smislu podataka koji ulaze u njihov sastav).

Drugi osnov za klasifikaciju predstavlja način pristupa podacima. Po tome razlikujemo sledeće dve vrste datoteka:

- sekvencijalne datoteke: kod ovih datoteka je radi pristupa nekim podacima potrebno proći kroz sve podatke koji se u datoteci nalaze ispred toga;
- direktne datoteke: kod ovih datoteka se neposredno može pristupiti bilo kojim podacima bez prolaženja kroz deo podataka ispred toga.

Razjasnimo ovu podelu na primeru slogovnih datoteka prikazanih na slici 3-4, od kojih je jedna sekvencijalna a druga direktna.



Slika 3-4: Sekvencijalni i direktni pristup.

Za sekvencijalnu datoteku važe sledeća:ograničenja:

- čitanje sloga: da bi se pročitao 5. slog (prikazana situacija), moraju se redom pročitati 4 sloga ispred toga;
- pisanje sloga: da bi se zapisao 5. slog, moraju se redom zapisati 4 sloga pre toga.

Sa druge strane, kod direktne datoteke nemamo navedena ograničenja. Konkretno:

- čitanje sloga: 5. slog se čita direkno; operativni sistem na osnovu poznate dužine sloga izračunava poziciju u datoteci koja odgovara tom slogu;
- pisanje sloga: slog se zapisuje direktno na poziciji koju operativni sistem izračunava na osnovu poznate dužine sloga; ako se ta pozicija nalazi iza dotadašnjeg kraja datoteke, operativni sistem kreira nedostajući prazan deo datoteke i zapisani slog postaje novi kraj datoteke.

Navedeni postupak direktnog pristupa povlači za sobom sledeće:

- poželjno je da datoteka bude sa istom dužinom slogova, pošto se vrlo teško ostvaruje direktni pristup sa slogovima promenljive dužine;
- mora biti raspoznatljiva situacija kada se iz praznog dela datoteke pokušava čitanje sloga koji pre toga nije zapisan;
- kriterijum pristupa određenom slogu mora na neki način da se pretvori u redni broj sloga u datoteci.

Poslednja klasifikacija datoteka koju pominjemo je po nameni. U tom pogledu postoje dve vrste datoteka:

- osnovne datoteke: datoteke koje sadrže osnovne podatke, odnosno slogove podataka od interesa za korisnike;
- pristupne datoteke: datoteke koje sadrže podatke koji obezbeđuju direktni pristup osnovnim podacima.

Uobičajeno je da se pristupne datoteke nazivaju indeksnim datotekama ili indeksima.

3.2.2 Direktan pristup podacima

Direktan pristup željenom slogu ostvaruje se zadavanjem rednog broja tog sloga u datoteci. Sa gledišta krajnjeg korisnika, ovo je zadovoljavajuće samo ako su identifikatori neki rastući celi brojevi. Takav slučaj imamo u našem primeru biblioteke, gde smo knjige redom označili šiframa (inventarnim brojevima) 001, 002 i tako redom.

U praksi postoje situacije kada je neposredno preslikavanje celobrojnih šifara na redne brojeve slogova nepodesno. Primer za to bi bila evidencija stanovnika Beograda na osnovu tzv. jedinstvenog matičnog broja. Poznato je da je matični broj sastavljen iz ukupno 13 cifara, 12 osnovnih i jedne kontrolne. Usvajanje takvih šifara za redne brojeve slogova bi značilo da čuvanje podataka od oko 2.000.000 stanovnika Grada iziskuje datoteku sa prostorom za oko 320.000.000.000 slogova. Iskorišćenje takve datoteke bilo bi oko 0.0006%.

Još češće su situacije kada šifre uopšte nisu celobrojne, nego kombinacije slova i brojeva, kada uopšte nije moguće neposredno preslikavanje šifara u redne brojeve slogova. Takav slučaj imamo u našem primeru biblioteke: šifre članova čine dva slova i redni broj.

Jedno od mogućih rešenja za navedene dve situacije jeste organizacija osnovne datoteke po metodi poznatoj pod nazivom "transformacija ključa u adresu":

- prvih N slogova datoteke čini tzv. primarno područje;
- ostatak datoteke, od sloga N+1 pa dalje, čini tzv sekundarno područje;
- definiše se funkcija transformacije takva da svaku moguću vrednost šifre (ključa) po kojoj se pristupa preslikava na ceo broj u intervalu [1,N];

Sa takvom organizacijom, postupak je sledeći

- bez obzira na to da li je u pitanju upis ili čitanje, pomoću funkcije transformacije se na osnovu zadate šifre odredi ceo broj K u navedenom intervalu;
- postupak dodavanja odnosno upisa novog sloga je sledeći: ako je K-ti slog primarnog područja slobodan, tu se upisuje novi slog; ako se u K-tom slogu primarnog područja nalazi neki ranije upisani slog koji predstavlja početak liste slogova čija se šifra preslikava na isti ceo broj K, novi slog se dodaje u sekundarnom području kao poslednji element liste;
- postupak traženja odnosno čitanja sloga sa zadatom šifrom je sledeći: ako je K-ti slog primarnog područja slobodan, ne postoji slog sa zadatom šifrom; ako u K-tom slogu primarnog područja postoji slog čija šifra odgovara zadatoj,u pitanju je traženi slog; ako u primarnom području postoji K-ti slog ali sa neogovarajućom šifrom, pretražuje se lista slogova u sekundarnom području sve do nailaska na slog sa odgovarajućom šifrom (traženi slog postoji) ili do nailaska na kraj liste (traženi slog ne postoji).

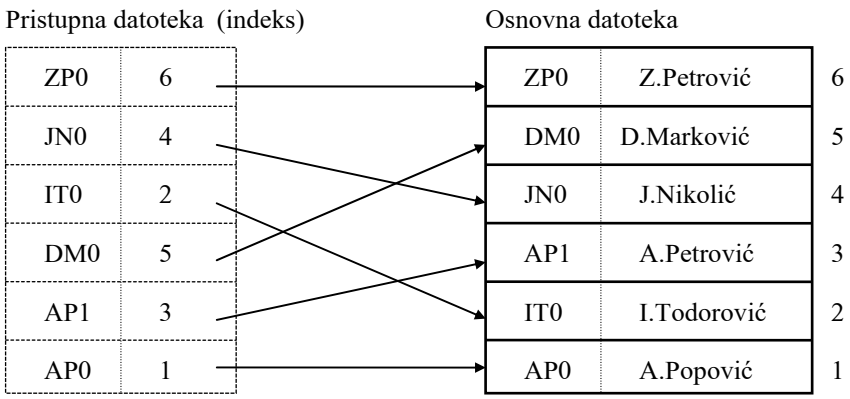
Organizacija pristupa transformacijom ključa u adresu razrešava problem pristupa podacima po bilo kakvoj šifri, ali se odlikuje i određenim manama:

- pristup se vremenom usporava kako se datoteka puni slogovima čije se šifre preslikavaju na iste cele brojeve;
- primarno područje nikad nije potpuno iskorišćeno, a naročito na početku rada, kada u datoteci ima relativno malo slogova;
- navedenom organizacijom nije moguće obezbediti ono što vrlo često treba pri korišćenju podataka, a to je uređenost u rastući ili opadajući redosled po nekom izabranom kriterijumu.

Sve navedene nedostatke otklanja organizacija direktnog pristupa preko posebne pristupne datoteke, odnosno indeksa:

- uz osnovnu direktnu datoteku koristi se pristupna direktna datoteka; svakom slogu u osnovnoj datoteci odgovara jedan slog u pristupnoj;
- svaki slog pristupne datoteke sadrži dva podatka: šifru osnovnog sloga i njegov redni broj u osnovnoj datoteci;
- radi što bržeg pretraživanja, slogovi pristupne datoteke su ulančani u stablo uređeno po rastućoj ili opadajućoj vrednosti šifre.

Kao ilustracija navedene organizacije neka nam posluži slika 3-5 koja prikazuje način zapisivanja podataka o šiframa i imenima autora naslova u biblioteci.



Slika 3-5: Direktni pristup pomoću indeksa.

Na slici je prikazan redosled slogova indeksa kakav se dobija kada se redom prolazi kroz sve slogove, a ne stvarni zapis tih slogova i njihova ulančanost u stablo. Slogovi u osnovnoj datoteci su zapisani redom, po hronologiji dodavanja. Prilikom svake izmene (dodavanje, brisanje) u osnovnoj datoteci uporedo se vrši i odgovarajuća izmena u indeksu.

Indeksi predstavljaju najbolju organizaciju direktnog pristupa, pošto obezbeđuju praktično sve što je potrebno pri manipulaciji podacima:

- brz direktni pristup traženim podacima;
- uređenost podataka po traženom kriterijumu.

Uz to, indeksi pružaju i neke dodatne pogodnosti:

- uopšte nismo ograničeni na samo jedan indeks po jednoj osnovnoj datoteci; možemo formirati onoliko indeksa koliko nam treba kriterijuma pristupa ili uređenosti podataka; tokom rada sa tako ustrojenim podacima, pre svake manipulacije moramo naglasiti koji od indeksa je tzv. aktivni indeks;
- pri formiranju indeksa nismo ograničeni na proste kriteriujume koji se svode na jedan podatak u slogu osnovne datoteke; možemo koristiti i složene indeksne izraze sastavljene od više podataka, što je naročito značajno za ostvarivanje željenih redosleda osnovnih podataka.

3.2.3 Strukture podataka u okviru baze podataka

Na kraju, pokušajmo da na osnovu svega do sada izloženog utvrdimo koje strukture podataka su pogodne za primenu u bazi podataka. Možemo da zaključimo:

- datoteke moraju biti slogovnog tipa, s obzirom da celu klasu nekog sistema predstavljamo datotekom, a instancu slogom;
- slogovi svake datoteke moraju imati istu strukturu, s obzirom da za svaku klasu nekog sistema uvodimo klasifikaciona svojstva a instance te klase opisujemo skupom vrednosti tih svojstava;
- datoteke moraju biti direktne, sa mogućnošću što bržeg direktnog pristupa podacima i sa mogućnošću ostvarivanja željene uređenosti podataka; ovo podazumeva i korišćenje transformacije ključa u adresu i indeksa.

PREPORUKA ČITAOCIMA

Od narednog poglavlja pa na dalje koriste se sintaksna i algoritamska notacija koje su izložene u prilogu B na kraju ove knjige. Čitaocima se preporučuje da pre prelaska na naredna poglavlja prouče navedeni prilog.