# 2

# *SISTEM UPRAVLJANJA BAZOM PODATAKA*

---

Sistem upravljanja bazom podataka je ključni deo svake baze podataka. U ovom poglavlju ćemo razmotriti detalje u vezi tog sistema. odnosno:

- zadatke sistema upravljanja bazom podataka;
- korisnike i načine rada sa bazom podataka;
- vezu programskih jezika sa bazom podataka;
- strukturu i rad sistema upravljanja bazom podataka.

## 2.1 Zadaci sistema upravljanja bazom podataka

Ako se podsetimo definicije baze podataka, možemo zaključiti da je sistem upravljanja bazom podataka (SUBP) "aktivni" deo baze podataka, ono što smo naznačili kao "skup elementarnih postupaka za održavanje i korišćenje".

### 2.1.1 Definicija baze podataka

SUBP se javlja kao posrednik između korisnika i operativnog sistema, i kao takav je zadužen za sledeće transformacije podataka u oba smera:

- datoteke ↔ sveukupni podaci baze podataka;
- sveukupni podaci baze podataka ↔ podaci koji odgovaraju potrebama korisnika.

Za obavljanje navedenih transformacija, SUBP mora da raspolaže sa definicijom baze podataka, odnosno sa detaljnim opisom njene strukture. U tom pogledu, sa gledišta realizacije SUBP postoje dva moguća rešenja:

- ugradnja definicije baze podataka u SUBP; ovo podrazumeva dogradnju (programiranje) SUBP za svaku konkretnu bazu podataka;
- definicija baze podataka izvan SUBP, pri čemu SUBP treba da interpretira tu definiciju; ovo podrazumeva univerzalni (interpreterski) karakter SUBP.

Kao univerzalno, usvojeno je drugo rešenje. To u vezi same definicije baze podataka podrazumeva sledeće:

- jezik visokog nivoa, na kome se može formulisati definicija bilo koje konkretne baze podataka koja je bliska prirodnom jeziku i razumljiva;
- jezik niskog nivoa, na kome se može formulisati definicija bilo koje konkretne baze podataka koja je pogodna za efikasnu interpretaciju od strane SUBP;
- poseban deo u okviru SUBP koji prevodi bilo koju definiciju visokog nivoa u odgovarajuću niskog nivoa;
- zapis definicije niskog nivoa u vidu datoteka kojima pristupa SUBP;
- deo SUBP koji interpretira zapisanu definiciju baze podataka radi njenog kreiranja.

U vezi sa prethodnim, uobičajeni su sledeći termini:

- DDL (od "Data Definition Language"): jezik visokog nivoa za opis baze podataka, kako sveukupno tako sa gledišta pojedinih korisnika;
- DDL Prevodilac: deo SUBP koji prevodi DDL definiciju na niski nivo i formira definicioni zapis baze podataka;
- Šema: definicija sveukupne baze podataka
- Podšema: definicija korisničkog viđenja baze podataka;
- Rečnik podataka: definicioni zapis baze podataka, koji sadrži šemu i sve podšeme.
- DDL interpreter: deo SUBP koji interpretira definicioni zapis baze podataka.

### 2.1.2 Manipulacija nad bazom podataka

U vezi realizacije dela SUBP za manipulaciju nad bazom podataka, odnosno za njeno održavanje i korišćenja, takođe su moguća dva pristupa:

- ugradnja konkretnih manipulacija nad konkretnom bazom podataka u SUBP; podrazumeva se dogradnja SUBP za svaku konkretnu bazu podataka;
- zadavanje konkretnih manipulacija nad konkretnom bazom podataka izvan SUBP; SUBP treba da na osnovu svake takve manipulacije i definicije baze podataka obezbedi interpretaciju, odnosno izvršavanje odgovarajuće sekvence elementarnih manipulacija nad bazom podataka; ovakav SUBP je univerzlnog karaktera.

Iz razumljivih razloga, i ovde je usvojeno drugo rešenje, čime je obezbeđena potpuna univerzalnost SUBP u smislu nezavisnosti od konkretne baze podataka i konkretnih operacija nad njom. Takav pristup podrazumeva sledeće:

- jezik visokog nivoa, na kome se na način blizak prirodnom jeziku može formulisati bilo kakva konkretna manipulacija nad bazom podataka;
- jezik niskog nivoa, na kome se može formulisati bilo kakva konkretne manipulacija nad bazom podataka, pri čemu mora da postoji mogućnost efikasne interpretacije od strane SUBP;
- poseban deo u okviru SUBP koji prevodi bilo koju formulaciju manipulacije iz visokog nivoa u odgovarajuću niskog nivoa;
- poseban deo u okviru SUBP koji interpretira formulaciju niskog nivoa za manipulaciju nad bazom podataka.

U vezi sa navedenim, u upotrebi su sledeći termini:

- DML (od "Data Manipulation Language"): jezik visokog nivoa za opis manipulacija nad bazom podataka;
- DML Prevodilac: deo SUBP koji prevodi DML formulaciju na formulaciju manipulacija niskog nivoa;
- DML Interpreter: deo SUBP koji interpretira formulacije manipulacija niskog nivoa.

### 2.1.3 Kontrola pristupa bazi podataka

Pored definicije baze podataka i formulacije manipulacija nad njom, kao značajan zadatak SUBP nameće se i kontrola pristupa bazi podataka. To podrazumeva mogućnost definicije kontrole u više nivoa, odnosno:

- *ko* može da pristupa bazi podataka, odnosno ko je sve korisnik baze podataka;
- *kojim* delovima baze podataka korisnici mogu da pristupe;
- *šta* korisnici mogu da rade sa delovima baze podataka kojima imaju pristup.

I ovde se kao prirodno nametnulo univerzalno rešenje po kome se sve te definicije nalaze izvan SUBP koji ih interpretira. To podrazumeva:

- jezik visokog nivoa, na kome se može formulisati definicija bilo kakvih prava pristupa bazi podataka koja je bliska prirodnom jeziku;
- jezik niskog nivoa, na kome se može formulisati definicija bilo kakvih prava pristupa bazi podataka koja je pogodna za efikasnu interpretaciju od strane SUBP;
- poseban deo u okviru SUBP koji prevodi bilo koju definiciju visokog nivoa u odgovarajuću niskog nivoa;
- zapis definicije niskog nivoa u jednu ili više datoteka kojim pristupa SUBP;
- korišćenje zapisane definicije prava pristupa bazi podataka u okviru ostalih funkcija SUBP.

U vezi sa kontrolom prava pristupa bazi podataka koriste se sledeći termini:

- DCL (od "Data Control Language"): jezik visokog nivoa za opis prava pristupa bazi podataka;
- DCL Prevodilac: deo SUBP koji prevodi DCL definiciju na niski nivo i formira definicioni zapis prava pristupa bazi podataka;
- Registar korisnika: definicioni zapis prava pristupa bazi podataka.

### 2.1.4 Ostali zadaci sistema upravljanja bazom podataka

U suštini, svi zadaci SUBP proizilaze iz opštih zahteva koji važe za bazu podataka i koje smo naveli u prethodnom poglavlju. Tri najznačajnija zahteva smo već detaljno obradili, a ovde sažeto navodimo još tri koje nećemo dalje razrađivati pošto to prevazilazi okvire ove knjige:

- upravljanje konkurentnim radom: SUBP u uslovima rada više korisnika mora da obezbedi pravilnu sinhronizaciju njihovog rada, kako bi se izbegle neželjene situacije kao što su korišćenje zastarelih podataka, uzajamno blokiranje i slično;
- upravljanje sistemom oporavka: SUBP u uslovima rada više korisnika treba primenom dnevnika aktivnosti (tzv. "log fajla") i drugim tehnikama da obezbedi podatke neophodne za oporavak baze podataka u slučaju nasilniog prekida rada (tzv. "pad sistema"), kao i da automatski izvrši oporavak po ponovnom pokretanju;
- statistika korišćenja baze podataka: SUBP treba da obezbedi statističke podatke o tome kojim delovima bazre podataka i na kakav način se najčešće pristupa, kako bi se na osnovu toga po potrebi mogla sprovesti reorganizacija baze podataka radi što boljih performansi.

## 2.2 Korisnici i načini rada sa bazom podataka

Pod korisnicima baze podataka podrazumevamo osobe koje imaju pristup bazi podataka. Prilikom dodele prava pristupa, svakom korisniku baze podataka dodeljuju se dva podatka:

naziv: javni skraćeni naziv korisnika, podatak koji može biti poznat svima;

lozinka: tajna šifra korisnika, podatak poznat samo korisniku.

Da bi uopšte mogao da radi sa bazom podataka, korisnik mora na početku da sprovede postupak *najave* (tzv. "login"), tokom koje ukucava prvo svoj naziv a zatim i lozinku. Prilikom najave, SUBP poredi uneti naziv i lozinku sa sadržajem registra korisnika i samo u slučaju saglasnosti dozvoljava dalji rad, i to u granicama prava dodeljenih tom korisniku.

Kao prirodno se nameće pitanje: Ko unosi podatke o korisnicima i njihovim pravima pristupa? Očigledno je da mora postojati neki "super-korisnik", osoba koja jedina ima ta prava nad bazom podataka. To nas dovodi do sledeće klasifikacije korisnika baze podataka:

- Administrator: osoba sa pravom kreiranja novih korisnika i dodele prava pristupa, i najčešće bez ikakvih ograničenja u radu sa bazom podataka;
- Korisnik (obični): osoba sa pravom pristupa bazi podataka u granicama koje je odredio administrator.

Pravilo je da obični korisnici nemaju prava kreiranja novih korisnika, ali zato mogu imati prava definicije svojih delova baze podataka, kao i dodele prava pristupa drugim korisnicima nad tim delovima.

Načine rada sa bazom podataka koje samim tim treba da podržava SUBP možemo klasifikovati po više osnova. Prva klasifikacija je po sadržaju rada i svodi se na sledeća dva načina rada:

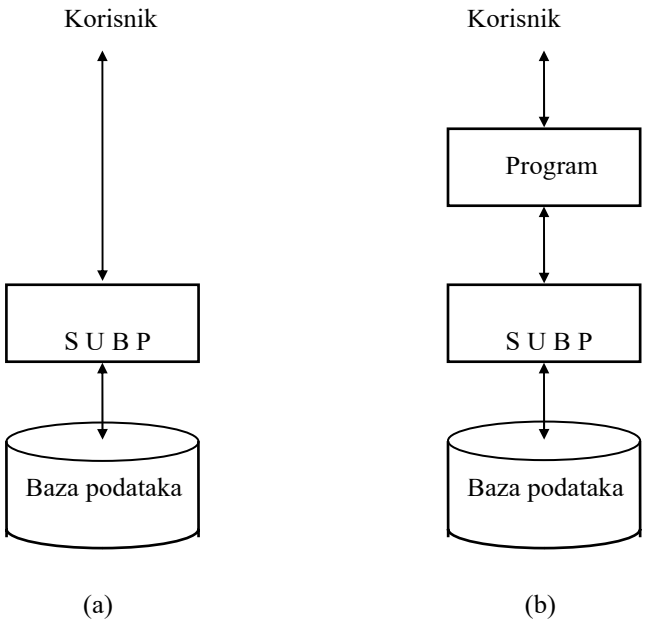
- definicioni: obuhvata unos definicija baze podataka i prava pristupa;
- manipulativni: obuhvata korišćenje baze podataka.

Administrator po pravilu radi sa bazom podataka na definicioni, a korisnik na manipulativni način. U situacijama kada definiše svoje delove baze podataka ili vrši dodelu prava pristupa nad tim delovima, i korisnik radi na definicioni način.

Druga klasifikacija polazi od forme rada i svodi se na sledeće načine rada:

- interaktivni: korisnik unosi jednu po jednu DML, DDL ili DCL naredbu, i SUBP ih jednu po jednu izvršava; u ovakvom načinu rada, korisnik komunicira direktno sa SUBP, a jedina ograničenja su ona koja proizilaze iz prava pristupa zapisanih u registru korisnika;
- programirani korisnik pokreće program preko koga vrši sav unos i pregled podataka, a program u sebi ima ugrađene DML naredbe; u ovakvom načinu rada, korisnik ne komunicira direktno sa SUBP već posredstvom programa, a prava pristupa koja ima mogu dodatno biti ograničena programom.

Na slici 2-1 šematski su prikazana oba načina rada sa bazom podataka.

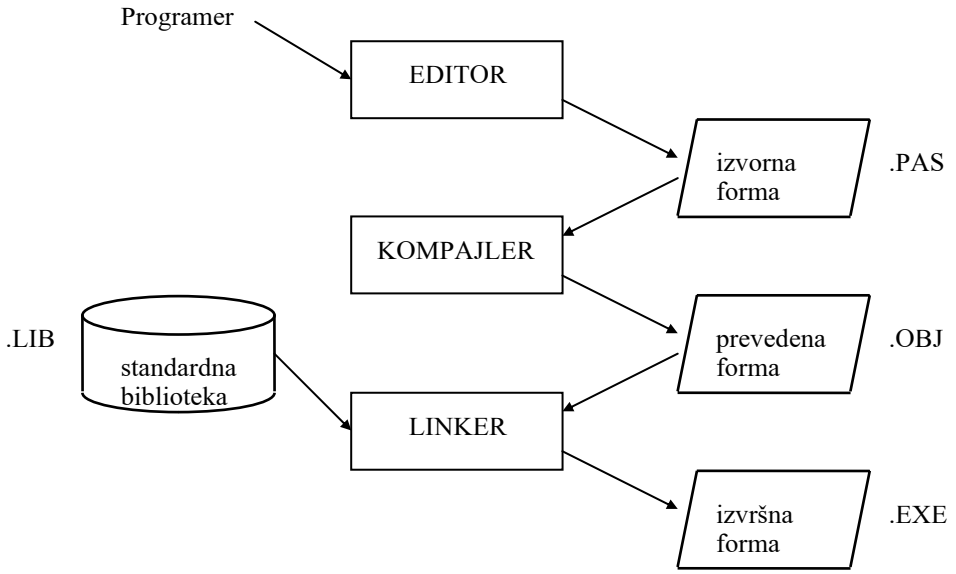



Slika 2-1: Interaktivni (a) i programski (b) način rada sa bazom podataka.

Program za rad sa bazom podataka je najčešće u izvršnoj formi, dobijen prevođenjem sa nekog programskog jezika, ali može biti i u formi DML procedure.

## 2.3 Programski jezici i baze podataka

Radi objašnjenja načina na koji je ostvarena veza između standardnih programskih jezika i baze podataka, neophodno je da se podsetimo na korake razvoja "običnog" programa (koji ne radi sa bazom podataka), od njegovog unosa u formi teksta na programskom jeziku visokog nivoa do formiranja izvršne forme koja može da se izvršava na računaru. U tu svrhu poslužiće nam šematski prikaz na slici 2-2.




Slika 2-2: Koraci formiranja "običnog" programa.

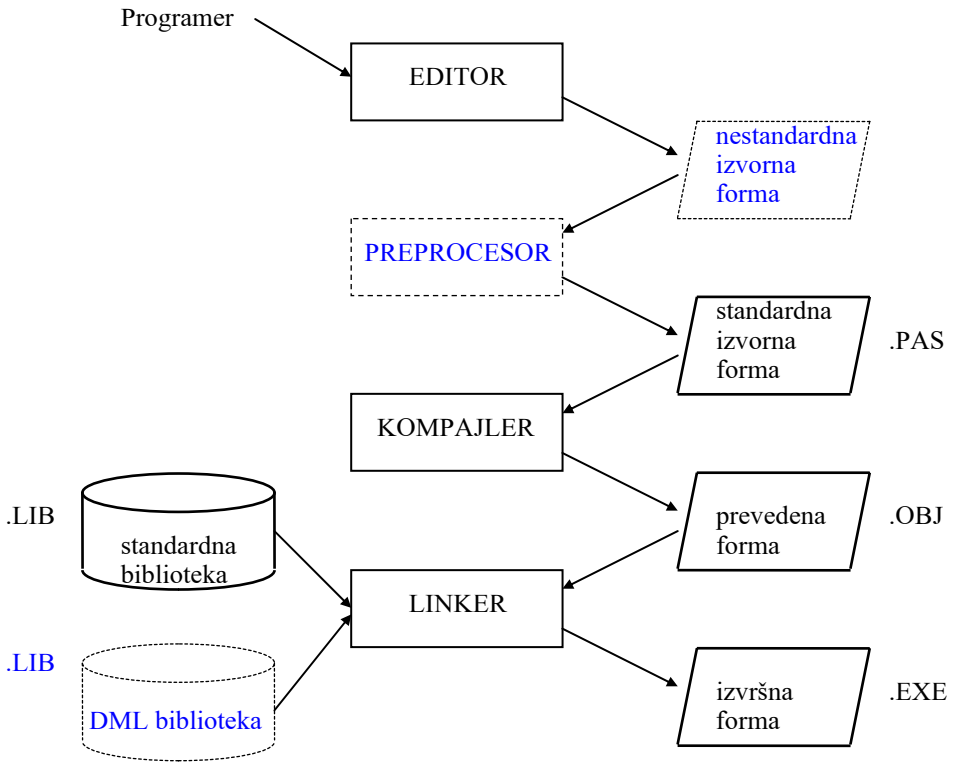
Za slučaj formiranja programa na PASCAL-u, imamo sledeće korake:

- pomoću EDITOR-a (programa za obradu teksta) formiraju se program i potrebni potprogrami u izvornoj PASCAL formi i nastaju jedna ili više datoteka sa ekstenzijom PAS;
- izvorna forma programa se prevodi u prevedenu formu pomoću KOMPAJLER-a (programa za prevođenje) i nastaju jedna ili više datoteka sa ekstenzijom OBJ;
- iz prevedene forme se uz korišćenje biblioteke standardnih procedura i funkcija pomoću LINKER-a (program za povezivanje) dolazi do programa u izvršnoj formi i nastaje datoteka sa ekstenzijom EXE.

Sa pojavom baza podataka i sve većim potrebama za programiranim radom sa njima, nastao je problem kako omogućiti da se iz standardnih programskih jezika pristupa bazi podataka. Na raspolaganju su bile samo dve mogućnosti:

- izmena standarda dopunom sintakse za potrebe rada sa bazom podataka;
- dodavanje podrške radu sa bazom podataka u okvirima postojećih standarda.

Dopuna sintakse je odbačena pošto bi to dodatno opteretilo i onako obimne standarde za programske jezike i dovelo do dodatne nerazumljivosti i pada performansi (u tom pogledu, poučan je primer programskog jezika PL/I koji je doživeo krah upravo zbog toga što je u taj jezik uneto isuviše sintaksnih mogućnosti). Usvojeno je rešenje sprezanja programskih jezika sa bazom podataka čija je suština prikazana na slici 2-3a, na kojoj su dodatni elementi u odnosu na sliku 2-2 prikazani isprekidano.




Slika 2-3a: Koraci formiranja programa za rad sa bazom podataka.

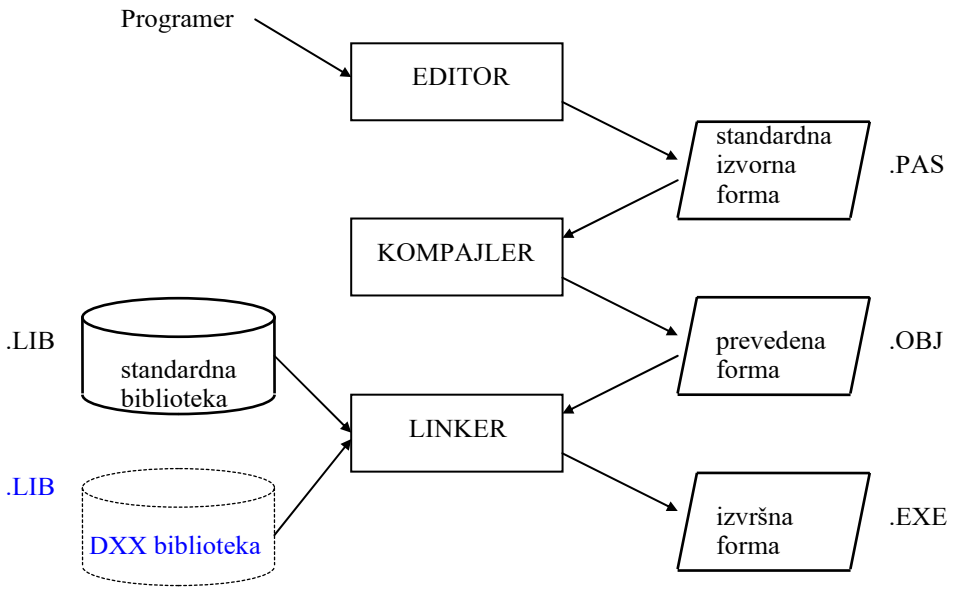
Suština rešenja prikazanog na slici 2-3a je u sledećem:

- pomoću EDITOR-a se kreira nestandardna izvorna forma koja pored standardnih instrukcija programskog jezika sadrži i DML instrukcije za rad sa bazom podataka;
- pomoću PREPROCESOR-a (specijalni program za obradu DML instrukcija), sve DML instrukcije se prevode u pozive posebnih DML procedura i funkcija za rad sa bazom podataka koji su po standardu za programski jezik; na taj način se dobija standardna izvorna forma programa;
- dalji postupak je isti kao i za "obične" programe, s tim što se u koraku povezivanja LINKER-om pored standardne biblioteke koristi i DML biblioteka koja sadrži sve DML procedure i funkcije za rad sa bazom podataka.

Da bi prethodno opisani postupak bio moguć, uz SUBP treba da bude isporučeno i sledeće, i to za svaki programski jezik od interesa:

- preprocesor DML instrukcija;
- biblioteka prevedenih DML procedura i funkcija.

Uz prethodno navedeni način sprezanja programskih jezika sa bazom podataka naknadno se pojavio još jedan koji je prikazan na slici 2-3b:




Slika 2-3b: Koraci formiranja programa za rad sa bazom podataka.

Suština ovog rešenja je u dopuni standarda, ali ne dopunom sintakse što je svojevremeno sasvim opravdano odbačeno, nego putem dopune predefinisanih funkcija i procedura skupom funkcija i procedura za rad sa bazom podataka. Mana ovog rešenja je to što standardizacija nije univerzalna nego je na nivou konkretnog programskog jezika (različita za svaki programski jezik), ali je prednost to što je osim manipulativnog obuhvaćen i definicioni rad sa bazom podataka (zato je oznaka dodatne biblioteke na slici 2-3b DXX umesto DML). Drugim rečima, sve što je korisniku raspoloživo u režimu interaktivnog rada sa bazom podataka može se ostvariti i u režimu programskog rada. Ovakvo rešenje je standardom od 1992 godine kojim je uvedeno dobilo naziv "Interfejs poziva" (Call-Level Interface). Danas dve najpoznatije implementacije interfejsa poziva su CLI za programski jezik C/C++ i SQLJ za programski jezik Java.

U okviru poređenja opisana dva načina sprezanja programskih jezika sa bazom podataka treba ukazati i na jednu bitnu razliku. Kod oba rešenja postoji dodatna biblioteka prevedenih funkcija i procedura, ali su te komponente kod prvog rešenja "sakrivene" od programera (pozive generiše preprocesor), dok su kod drugog rešenja "vidljive" programeru (on ih eksplicitno poziva unutar programa).

## 2.4 Struktura i rad sistema upravljanja bazom podataka

Na osnovu svega do sada izloženog možemo ukazati na osnovne komponente SUBP i njegovu strukturu, rad i vezu sa podacima baze podataka.

Kao prvo, primetimo da se podaci baze podataka nalaze izvan SUBP koji je univerzalnog karaktera. Ti podaci se javljaju u tri vida:

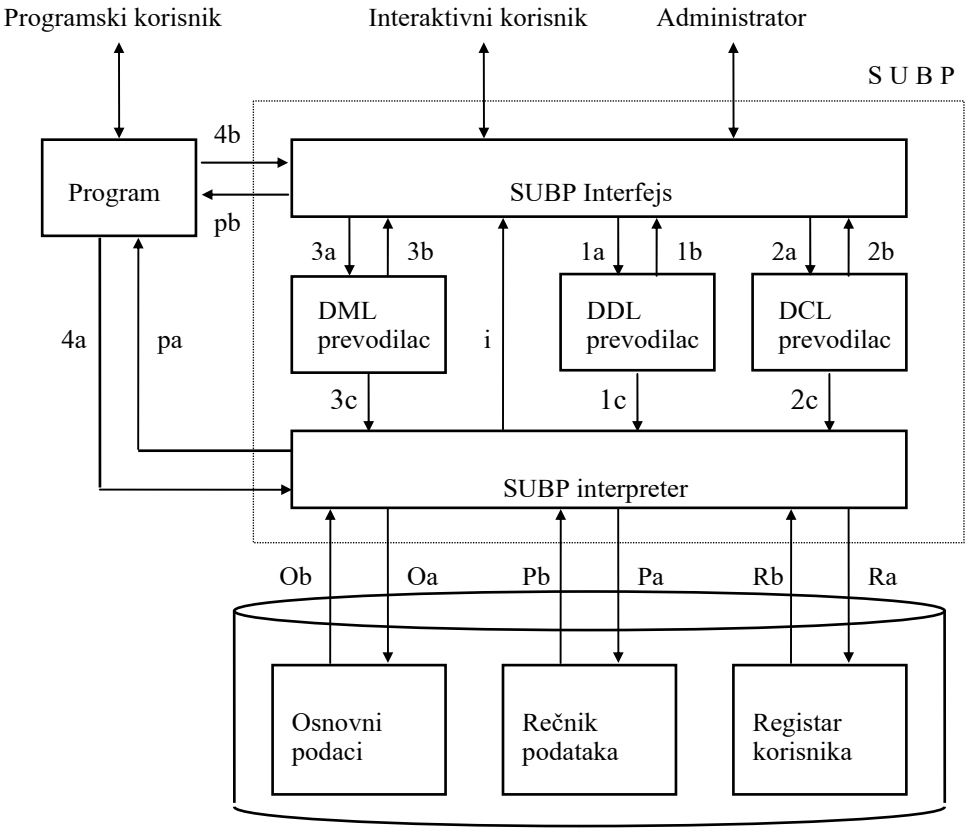
- osnovni podaci: podaci u smislu korisnog (korisničkog) sadržaja baze podataka;
- rečnik podataka: "podaci o podacima", definicioni opis baze podataka;
- registar korisnika: podaci o korisnicima i njihovim pravima pristupa.

Isto tako, i za sve korisničke programe za rad sa bazom podataka smatraćemo da se nalaze izvan SUBP i da čine posebnu komponentu informacionog sistema koju ćemo nazvati "Sistem korisničkih programa" ili skraćeno SKP. Po takvom gledištu, i SUBP i SKP čine ono što nazivamo "skup postupaka za održavanje i korišćenje".

Kao osnovne komponente SUBP uočavamo:

- SUBP interfejs (sprega sa korisnikom): deo SUBP zadužen u uslovima interaktivnog rada za svu kumunikaciju sa korisnikom;
- DDL prevodilac: prevodi DDL definicije baze podataka u definiciju niskog nivoa pogodnu za interpretiranje;
- DCL prevodilac: prevodi DCL definicije korisnika i prava pristupa u definiciju niskog nivoa pogodnu za interpretiranje;
- DML prevodilac: prevodi DML opise manipulacija u formu niskog nivoa pogodnu za interpretiranje;
- SUBP interpreter: centralna komponenta SUBP, interpretira sve opise niskog nivoa i tako omogućava održavanje i korišćenje svih podataka.

Na slici 2-4 prikazana je struktura SUBP, njegov rad i veza sa podacima.




Slika 2-4: Struktura SUBP, način rada i veza sa podacima.

Integracija DML, DDL i DCL funkcija u jedinstveni SUBP interpreter spovedena je i zahvaljujući tome da se i osnovni podaci i rečnik podataka i registar podataka zapisuju na isti način u bazi podataka. Time je omogućeno da se za održavanje i korišćenje svih tih podataka koriste isti elementarni postupci.

Način rada SUBP u raznim uslovima možemo razjasniti uz pomoć slike 2-4 na kojoj su označeni bitni tokovi podataka. Posmatrajmo prvo interaktivni rad sa bazom podataka tokom koga se formira definicija baze podataka. Redosled događaja je:

- preko SUBP interfejsa unosi se DDL naredba; interfejs prosleđuje tu naredbu DDL prevodiocu (1a);
- u slučaju greške koja se može utvrditi samo na osnovu prevođenja, DDL prevodilac dostavlja interfejsu poruku o tome (1b); interfejs prikazuje tu poruku i time se završava obrada unete naredbe;
- u slučaju da nema greške navedenog tipa, DDL prevodilac formira odgovarajuću sekvencu naredbi niskog nivoa i prosleđuje je SUBP interpreteru (1c);
- SUBP interpreter izvršava prosleđenu sekvencu naredbi niskog nivoa; prvo se ispituje njena saglasnost sa rečnikom podataka (Pb); u slučaju saglasnosti, ažuriraju se rečnik podataka (Pa, upis ili izmena definicije) i osnovni podaci (Oa, kreiranje "prazne" ili izmena postojeće strukture);
- SUBP interpreter prosleđuje SUBP interfejsu poruku o ishodu prethodne aktivnosti (i) uključujući tu i situacije greške; interfejs prikazuje tu poruku.

U slučaju interaktivnog definisanja prava pristupa bazi podataka redosled događaja je nešto drugačiji, ali samo u delu koji obavlja SUBP interpreter:

- preko SUBP interfejsa unosi se DCL naredba; interfejs prosleđuje tu naredbu DCL prevodiocu (2a);
- u slučaju greške koja se može utvrditi samo na osnovu prevođenja, DCL prevodilac dostavlja interfejsu poruku o tome (2b); interfejs prikazuje tu poruku i time se završava obrada unete naredbe;
- u slučaju da nema greške navedenog tipa, DCL prevodilac formira odgovarajuću sekvencu naredbi niskog nivoa i prosleđuje je SUBP interpreteru (2c);
- SUBP interpreter izvršava prosleđenu sekvencu naredbi niskog nivoa; prvo se ispituje njena saglasnost sa rečnikom podataka (Pb) i registrom korisnika (Rb); u slučaju saglasnosti, ažurira se registar korisnika (Ra, upis ili izmena definicije);
- SUBP interpreter prosleđuje SUBP interfejsu poruku o ishodu prethodne aktivnosti (i); interfejs prikazuje tu poruku.

Za interaktivni manipulativni rad sa bazom podataka, kada se menjaju ili koriste osnovni podaci, imamo sledeći redosled događaja:

- preko SUBP interfejsa unosi se DML naredba; interfejs prosleđuje tu naredbu DML prevodiocu (3a);
- u slučaju greške koja se može utvrditi samo na osnovu prevođenja, DML prevodilac dostavlja interfejsu poruku o tome (3b); interfejs prikazuje tu poruku i time se završava obrada unete naredbe;
- u slučaju da nema greške navedenog tipa, DML prevodilac formira odgovarajuću sekvencu naredbi niskog nivoa i prosleđuje je SUBP interpreteru (3c);
- SUBP interpreter izvršava prosleđenu sekvencu naredbi niskog nivoa; prvo se ispituje njena saglasnost sa rečnikom podataka (Pb) i registrom korisnika (Rb); u slučaju saglasnosti, čitaju se (Ob) ili menjaju osnovni podaci (Oa), zavisno od prirode zadate manipulacije;
- SUBP interpreter prosleđuje SUBP interfejsu poruku o ishodu i eventualne rezultate prethodne aktivnosti (i); interfejs sve to prikazuje.

U vezi interaktivnog rada napomenimo još i to da većina SUBP dozvoljava mogućnost formiranja sekvenci naredbi u vidu tekst datoteka koje se zatim kao celina prosleđuju SUBP interfejsu. Pri tome je potrebno na odgovarajući način navesti naziv takve datoteka. U takvoj varijanti rada, SUBP će sve rezultate izvođenja tih naredbi i poruke o ishodu zapisati u posebnu datoteku koju korisnik može pažljivo da pregleda. Ovakav način rada je naročito pogodan kod zadavanja obimnih definicija baze podataka.

Preostaje još da razmotrimo programski rad sa bazom podataka, koji je manipulativnog karaktera. Tu postoje dve mogućnosti. Ako je prilikom formiranja programa u program preko odgovarajućih procedura i funkcija uključen opis manipulacija na niskom nivou pogodnom za interpretiranje, jasno je da nema potrebe za SUBP interfejsom i DML prevodiocem. Redosled događaja je sledeći:

- program prosleđuje SUBP interpreteru odgovarajuću sekvencu naredbi niskog nivoa (4a);
- SUBP interpreter izvršava prosleđenu sekvencu naredbi niskog nivoa; prvo se ispituje njena saglasnost sa rečnikom podataka (Pb) i registrom korisnika (Rb); u slučaju saglasnosti, čitaju se (Ob) ili menjaju osnovni podaci (Oa), zavisno od prirode zadate manipulacije;
- SUBP interpreter prosleđuje programu poruku o ishodu prethodne aktivnosti i eventualne rezultate (pa); na programu je kako će to da koristi.

Sa druge strane, ako je u program uključen opis manipulacija na visokom nivou, SUBP interfejs i DML prevodioc su neophodni i redosled događaja je sledeći:

- program prosleđuje SUBP interfejsu odgovarajuću sekvencu DML naredbi (4b); interfejs prosleđuje tu naredbu DML prevodiocu (3a);
- u slučaju greške koja se može utvrditi samo na osnovu prevođenja, DML prevodilac dostavlja interfejsu poruku o tome (3b); interfejs prosleđuje tu poruku programu (pb) i time se završava obrada unete naredbe;
- u slučaju da nema greške navedenog tipa, DML prevodilac formira odgovarajuću sekvencu naredbi niskog nivoa i prosleđuje je SUBP interpreteru (3c);
- SUBP interpreter izvršava prosleđenu sekvencu naredbi niskog nivoa; prvo se ispituje njena saglasnost sa rečnikom podataka (Pb) i registrom korisnika (Rb); u slučaju saglasnosti, čitaju se (Ob) ili menjaju osnovni podaci (Oa), zavisno od prirode zadate manipulacije;
- SUBP interpreter prosleđuje programu poruku o ishodu prethodne aktivnosti i eventualne rezultate (pa); na programu je kako će to da koristi.

Ovaj poslednji način programskog rada je sporiji s obzirom da uključuje rad SUBP interfejsa i DML prevodioca, ali je zato fleksibilniji. Pošto se zadaju u izvornom obliku, DML naredbe nisu “fiksirane” u programu nego se mogu generisati tokom rada programa zavisno od dijaloga sa korisnikom i drugih okolnosti.

Uz navedene, postoje i dodatni delovi SUBP zaduženi za funkcionalnosti baze podataka koje ne razmatramo u ovoj knjizi. To su:

- optimizator operacija (querry optimizer): deo koji na osnovu strukture operacije, strukture baze podataka i statistike korišćenja baze podataka vrši optimizaciju (ubrzanje) postupka izvršenja operacija;
- upravljač memorije (buffer manager): deo koji je zadužen za efikasno i brzo korišćenje memorijskog prostora organizovanog u tri nivoa: I primarna memorija (RAM), II sekundarna memorija (HD), III arhivska memorija;
- upravljač transakcija (transaction manager): deo koji je zadužen za upravljanje višekorisničkim konkurentni radom (radom više korisnika istovremeno);
- upravljač oporavka (recovery manager): deo koji je zadužen za oporavak baze podataka u slučaju kvara.

Uz sve navedeno što je ugrađeno u SUBP, danas postoje i brojni pomoćni programi za olakšanje poslova administratoru baze podataka, kao što su upravljanje sistemom korisnika, praćenje i podešavanje performansi, obezbeđenje sigurnosti, itd.

Šta na osnovu svega prethodnog možemo da zaključimo o sistemu upravljanja bazom podataka (SUBP)? Kao prvo SUBP je jedan izuzetno složen sistem univerzalnog karaktera, u smislu da je primenjiv za bilo koju konkretnu bazu podataka.
Može se slobodno zaključiti da SUBP predstavlja vrhunski domet informatičkog inženjerstva kod koga je do savršenstva ugrađen jedan od najznačajnih principa u savremenoj informatici. To je princip parametrizacije, odnosno izrade softvera koji generalno rešava čitavu jednu klasu problema, pri čemu se za konkretne probleme iz te klase naknadno zadaju parametri koje taj softver očitava iz odgovarajućih zapisa i interpretira ih tokom rada.