dr Alempije V. Veljović
dr Angelina Njeguš

Osnove relacionih i analitičkih baza podataka

Megatrend univerzitet primenjenih nauka
Beograd, 2004.

1.UVOD

Sistem se najopštije definiše kao skup objekata (entiteta) i njihovih međusobnih veza. Objekti u sistemu mogu da budu neki fizički objekti, koncepti, događaji i drugo. Objekti se u modelu nekog sistema opisuju preko svojih svojstava (atributa) i skupa relacija koje povezuju te objekte, kao i osobina tih relacija.

Imajući u vidu gornju definiciju sistem za upravljanje bazom podataka (SUBP) (Database Management System - DBMS) definiše se kao softverski sistem za čuvanje i pretraživanje podataka. Kao softverski sistem on, u razvoju računarstva, zamenjuje i eliminiše nedostatke sistema datoteka koji su se koristili u klasičnoj obradi podataka.

U svetlu ovoga razmatraće se relacione baze podataka i veza sa klijent server arhitekturom i analitičke baze podataka..

Relaciona baza podataka stvara takav tip strukture koji se koristi za izražavanje odnosa između podataka u obliku jednostavnih dvodimenzionalnih tabela. Za razliku od većine drugih baza podataka (hijerarhijskih i mrežnih), relaciona baza podataka ima solidne teoretske osnove, za koje ima najveće zasluge dr E.F.Codd, koji je objavio prve rezultate svojih istraživanja već 1969. godine.

Obrada podataka po modelu *klijent/server* podrazumeva primenu distribuirane obrada podataka jer se koriste prednosti mrežnog rada i obrade na centralnom računaru i omogušuje se primena CASE alati tj. do izrazaja dolazi primena standarda ISO 9000. Slabost klijen server arhitekture je slabija pouzdanost usled heterogenosti.

Analitičke baze podataka pojavio se kao posledica:

- potreba za pretvaranjem transakcionih podataka u informacije,
- omogućavanja veće inicijative korisnicima, koji ne moraju da znaju SQL i detalje o postavci informacionog sistema,
- zahteva za automatizacijom rada analitičara, komercijalista i menadžera.

2.OPŠTA TEORIJA SISTEMA

Grčka reč sistema označava skup elemenata ili celinu sastavljenu od delova. Opšta teorija sistema predstavlja naučnu oblast koja se bavi izučavanjem sistema i zakonitosti koje u njima nastaju. Jedna od najvažnijih karakteristika teorije sistema jeste u pristupu, a to je da se svaka celina posmatra kao deo neke veće celine. Drugim rečima, sistem se izučava u povezanosti sa okolinom. Nastajanje opšte teorije sistema dovelo je do stvaranja sistemskog pristupa, kao i do novih tehnika i metoda analize sistema.
Kada smo upotrebili termin "sistemski pristup", time smo naglasili da se svi predmeti i pojave posmatraju u njihovoj dinamičnosti i celovitosti u odnosu na okruženje.

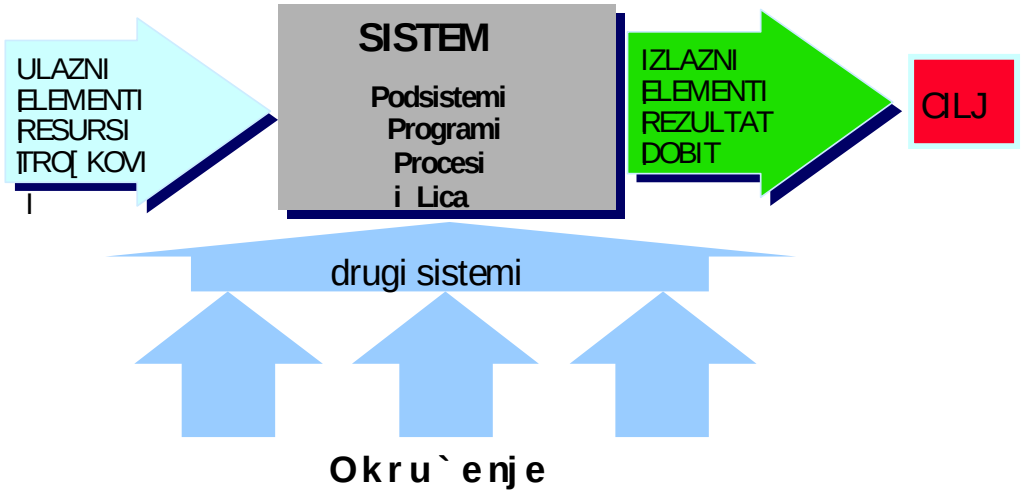
Kao glavni objekat proučavanja teorije sistema istaknuti su fenomeni rasta i razvoja sistema. Naime, ovi procesi svuda prate iste zakonitosti, bez obzira da li se radi o problemima rasta u biologiji, demografiji, ekonomiji ili bilo kojoj drugoj naučnoj disciplini.

Moderna teorija sistema ima svoje izvorište u opštoj teoriji sistema; međutim, po obimu područja istraživanja, saznanja i metoda, nešto je šireg značaja. Poznate su tako, naprimer, matematičke teorije sistema, zatim teorija samotranscendentalnih sistema i druge.

Cilj opšte teorije sistema je da služi kao jedinstveni metodološki i pojmovno-kategorijalni okvir sporazumevanja ljudi različitih specijalnosti. Takođe, njen cilj je da obuhvati i objedini fundamentalne istine i pojmove koji važe u svim specifičnim sistemima i teorijama koje se njima bave. Imajući u vidu opštu teoriju sistema, u daljem tekstu definisaće se pojam sistema i njegove karakteristike.

Sistem se najopštije definiše kao skup objekata (entiteta) i njihovih međusobnih veza. Objekti u sistemu mogu da budu neki fizički objekti, koncepti, događaji i drugo. Objekti se u modelu nekog sistema opisuju preko svojih svojstava (atributa) i skupa relacija koje povezuju te objekte, kao i osobina tih relacija.

Dejstvo okoline na sistem opisuje se preko ulaza u sistem, a dejstvo sistema na okolinu preko njegovih izlaza, kao što se vidi na sledećoj slici.



Slika 1. Grafički prikaz realnog sistema

Dinamičko ponašanje realnog sistema standardno se predstavlja na sledeći način: ulazi u sistem menjaju stanja sistema. Stanje sistema se definiše kao skup informacija o prošlosti i sadašnjosti sistema koji je potreban da bi se, pod dejstvom budućih poznatih ulaza, mogli odrediti budući izlazi. U stanju sistema koncentrisana je celokupna istorija realnog sistema. Izlazna transformacija definiše neki način merenja ili posmatranja dinamičkog ponašanja realnog sistema i daje, na osnovu stanja sistema, njegove izlaze.

Sistem uvek predstavlja neku celinu, koja je eksplicitno određena vezama elemenata. Između elemenata sistema postoje određene međuzavisnosti i, zahvaljujući tome, sistem postaje takva celina u kojoj su svi elementi u međusobnoj vezi, na neposredan ili posredan način. Upravo zbog te činjenice, sistem i njegove osobine se ne mogu shvatiti bez ovih međusobnih veza.

Svaki sistem moguće je dekomponovati na podsisteme i elemente. Istovremeno, svaki sistem je deo nekog šireg sistema. *Hijerarhičnost* se mora uzeti u obzir prilikom istraživanja: ponašanja, funkcionisanja, razvoja i izgradnje i upravljanja sistemima.

Pod *dinamičnošću* sistema podrazumeva se da se veze između elemenata sistema ostvaruju razmenom materije, energije i informacija između njih. Ukoliko je ova razmena između elemenata sistema značajna za njegovo postojanje, tada se govori o dinamičkim sistemima. Za razliku od njih, kod statičkih sistema dinamika između elemenata u okviru sistema nije primarna za njihovo postojanje.

Otvorenost predstavlja komunikaciju između elemenata sistema i elemenata iz njegovog okruženja. Ova komunikacija ostvaruje se razmenom materije, energije i informacija. Odatle potiče i podela na otvorene i zatvorene sisteme. Otvoreni sistemi imaju vezu sa okruženjem, dok kod zatvorenih sistema postoji izvesna veza, ali ona nije značajna za njihovo postojanje.

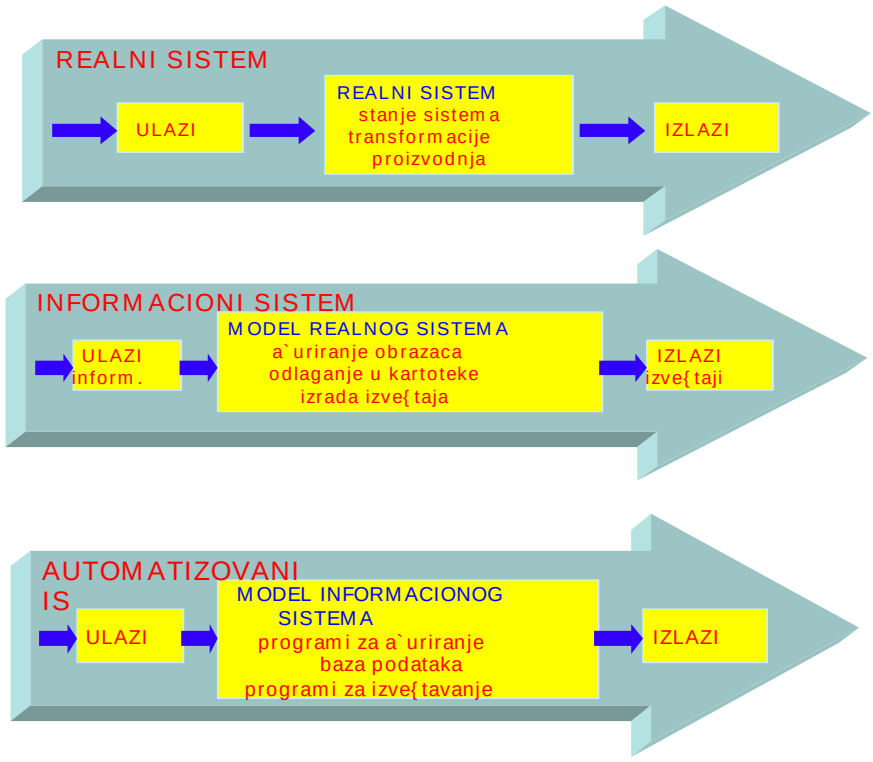
Upravljivost sistema označava mogućnost njegove regulacije. To znači da se sistem ponaša u skladu sa zahtevima upravljača. Polazeći od mesta upravljača, koji reguliše ponašanje sistema, razlikuju se samoupravljivi sistemi i sistemi kojima se upravlja van sistema. Samoupravljive sisteme karakteriše postojanje upravljačkog podsistema koji reguliše ponašanje sistema u celini.

Što se tiče sistema kojima se upravlja van njih, prisutna je relativnost u stepenu uticaja na regulaciju ponašanja, jer je određeni stepen regulativnosti ponašanja sadržan i u samom sistemu.

Predmet daljih razmatranja vezan je za definisanje informacionih sistema.

3. INFORMACIONI SISTEM

Polazeći od tumačenja pojmova "sistem" i "informacija", proizilazi definicija "informacionog sistema". *Informacioni sistem* se može definisati kao sistem u kome su relacije između objekata i relacije između atributa objekata ostvarene prenosom informacija (16). Informacioni sistem nastaje preslikavanjem realnog sistema kao što je pokazano na sledećoj slici.



Slika 2. Model Informacionog sistema

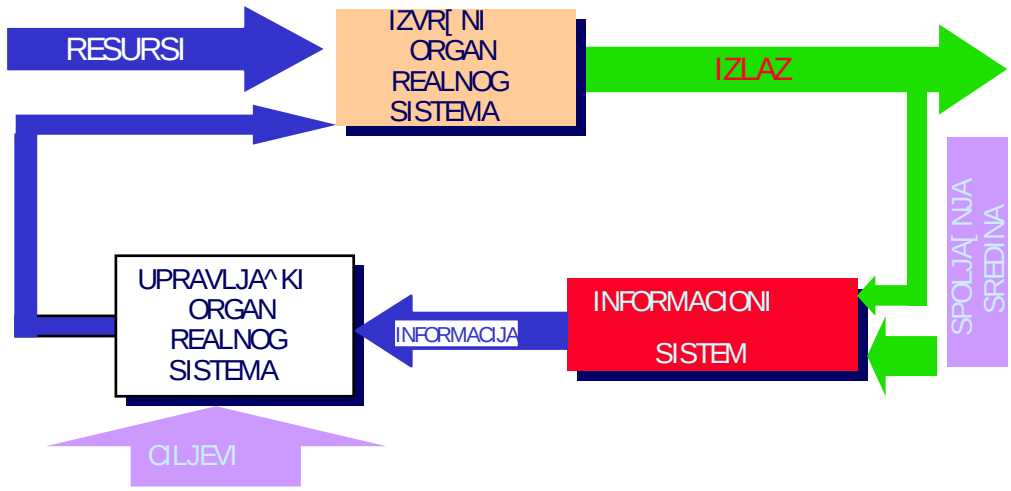
Preslikavanje realnog sistema u informacioni sistem izvodi se postupkom modeliranja procesa i

podataka (18).

Izgradnja informacionih sistema zasnovanih na primeni računarske tehnologije značila je automatizaciju osnovnih funkcija postojećeg informacionog sistema. Zbog toga se najčešće takvi informacioni sistemi nazivaju automatizovani informacioni sistemi. Automatizovani informacioni sistemi nastaju fizičkim modeliranjem podataka tj. definisanjem šeme baze podataka i definisanjem korisničkog interfejsa.

Osnovu automatizovanog informacionog sistema čini baza podataka, jer ona predstavlja fundamentalne, stabilne, sporo izmenljive karakteristike sistema, objekte u sistemu i njihove međusobne veze. Ako je baza podataka dobar model stanja realnog sistema, ako programi za održavanje dobro modeliraju dejstvo ulaza na stanje realnog sistema, onda će se bilo koja informacija potrebna za upravljanje (izlazi), čak i one unapred nepredviđene, moći dobiti iz IS. Time se dobrim delom zaobilazi ključni problem u konvencionalnom pristupu razvoju IS, specifikacija zahteva za informacijama, postupak projektovanja se ne bazira na tim stalno promenljivim zahtevima, već na modeliranju fundamentalnih, stabilnih karakteristika sistema (18).

Zbog svog značaja, informacioni sistem je zauzeo veoma istaknuti položaj unutar nekog organizovanog realnog sistema, kao što se može videti na sledećoj slici.



Slika 3. Položaj Informacionog sistema u odnosu na organizovani realni sistem

U organizacionom sistemu se uvek nešto dešava, odvijaju se radni procesi, troši se energija, materijalni resursi i informacija kao resurs da bi se stvorile nove vrednosti. Informacija kao resurs egzistira u raznim oblicima dokumentacije, koja se tokom odvijanja procesa u organizacionom sistemu koristi i stvara.

3.1. Arhitektura informacionih sistema

Arhitektura informacionih sistema obezbeđuje jedinstveni kostur po kome će različiti ljudi sa različitim perspektivama (pogledima) organizovati i videti fundamentalne "blokove" razvoja informacionih sistema (Slika 5).

Pretpostavimo da želite da projektujete jedan informacioni sistem. Različiti ljudi će imati različite poglede na sistem. Menadžeri, korisnici, tehničari, svi oni će posmatrati sistem na različit način i sa različitim nivoom detalja. Ove ljude nazivamo nosiocima informacionog sistema, odnosno **stakeholders-**ima (stejkhould*ə*(r)s). Oni se grubo mogu klasifikovati u četiri grupe:

- *Vlasnici sistema (System Owners)* finansiraju razvoj i održavanje informacionog sistema. Oni poseduju sistem, postavljaju prioritete u sistemu i određuju politiku za njegovo korišćenje. U nekim slučajevima, vlasnici sistema mogu biti i korisnici sistema.
- *Korisnici sistema (System Users)* su ljudi koji za obavljanje svojih poslova, koriste informacioni sistem. Danas korisnici sistema rade rame uz rame sa projektantima sistema.
- *Projektanti sistema (System Designers)* projektuju sistem kako bi izašli u susret zahtevima korisnika. Oni projektuju baze podataka, ekrane, mreže i programe. U nekim slučajevima, projektanti sistema mogu biti i graditelji sistema.
- *Graditelji sistema (System Builders)* su tehnička lica koja konstruišu, testiraju i isporučuju sistem.

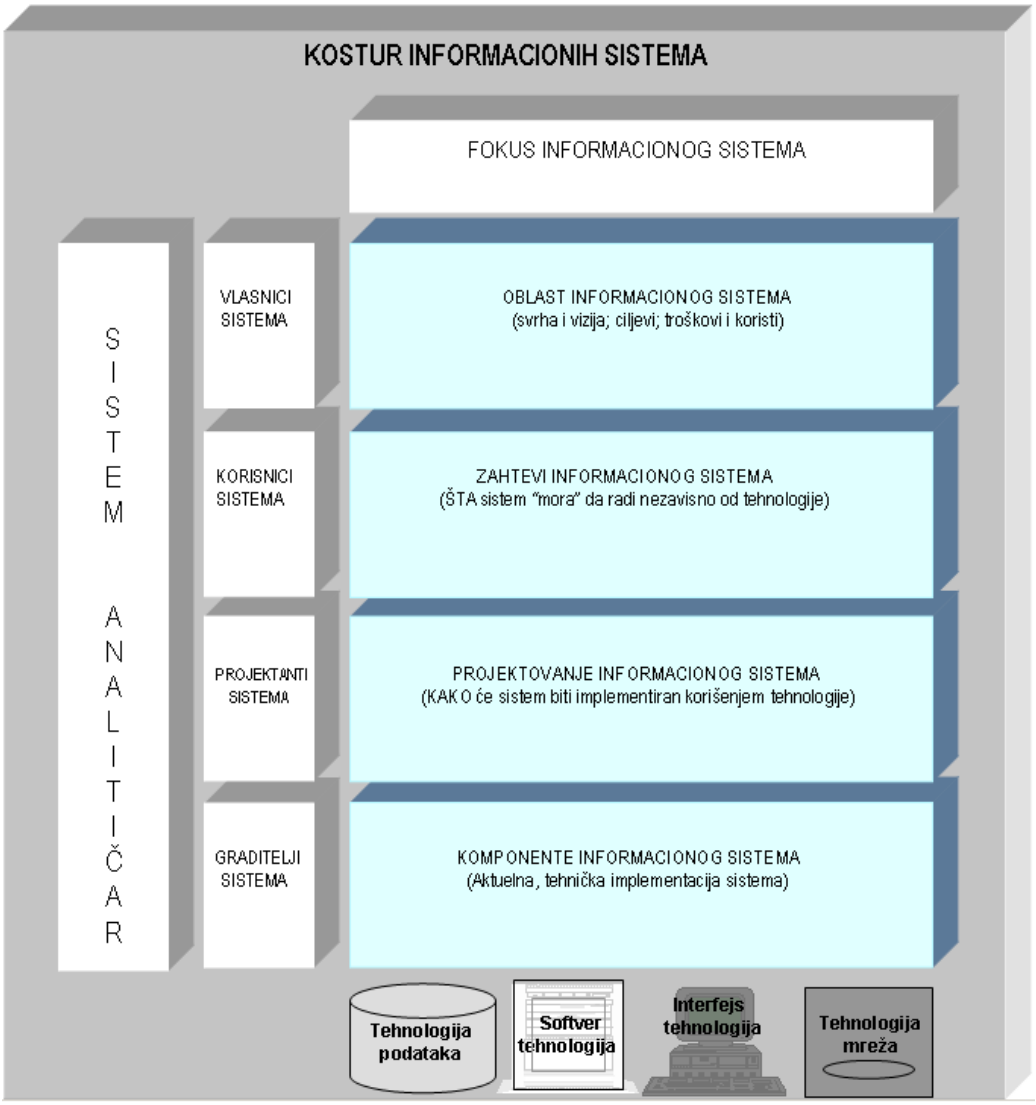
Kao što je prikazano na slici 5, svaka grupa stakeholders-a je jedan red na kosturu informacionog sistema i može se videti, da generalno postoje četiri perspektive ili pogleda nad informacionim sistemima, a to su perspektive vlasnika, korisnika, projektanata i graditelja. Svi oni zajedno čine informacione radnike. Informacioni radnici su oni ljudi koji se bave kreiranjem, sakupljanjem, obrađivanjem, distribucijom i korišćenjem informacija.

Različiti stakeholders-i se mogu usredsrediti na različite aspekte sistema. Na primer, jednom projektantu se može dodeliti da projektuje bazu podataka, dok se drugom projektantu zadaje razvijanje programa. Danas se mogu identifikovati najmanje tri različita fokusa sistema (Slika 6), a to su:

PODACI – sirov materijal koji se koristi za kreiranje informacija.

PROCESI – aktivnosti koje izvršavaju misiju poslovanja.

INTERFEJSI – pokazuju kakav je međusoban uticaj sistema na ljude i druge sisteme.



Slika 5. Perspektive informacionog sistema

Preseci perspektiva (redova) i svakog fokusa (kolona) definišu fundamentalne blokove ili ćelije informacionog sistema (Slika 6). U zavisnosti od toga da li ste vlasnik, korisnik, projektant ili graditelj i u zavisnosti od toga na šta želite da se fokusirate, da li na podatke, procese ili interfejse, vaš *pogled* na arhitekturu sistema će se razlikovati od drugih pogleda. Projektant baze podataka vidi šemu baze podataka, programer vidi aplikacije itd.

Fundamentalni blokovi informacionog sistema

Blokovi informacionog sistema ne egzistiraju izolovano, već moraju biti sinhronizovani kako bi se izbegle nedoslednosti i nekompatibilnosti unutar sistema. Na primer, projektant baze podataka i programer imaju svoj sopstveni pogled na arhitetkturu sistema, ali ti pogledi moraju biti koordinirani i kompatibilni, da bi sistem bio uspešan.

Fundamentalni blokovi podataka

Kada inženjeri projektuju jedan proizvod, oni moraju da sačine sastavnicu tog proizvoda. Sastavnica ne govori o funkciji tog proizvoda, već o njegovim sastavnim delovima, odnosno pokazuje koje su to sirovine, poluproizvodi i druge komponente koje učestvuju u izgradnji finalnog proizvoda. Ista analogija se koristi i za informacione sisteme. Podaci se mogu posmatrati kao sirovine koje se koriste da bi se "proizvele" informacije.

Podaci se mogu smatrati kao jedan od primarnijih fundamentalnih blokova razvoja informacionog sistema. Ovde je cilj prikupiti i uskladištiti podatke korišćenjem tehnologije baze podataka, koja će znatno olakšati njihovo čuvanje.

Pogled vlasnika sistema na sistem podataka

Prosečan vlasnik sistema, obično nije zainteresovan za sirove podatke. Vlasnik je zainteresovan za resurse poslovanja, kojih čine kupci, proizvodi, oprema, zgrade, porudžbine ili plaćanja. Njegov domen jeste da za svaki objekat i relacije između objekata identifikuje eventualne probleme, mogućnosti, ciljeve i ograničenja. Zajedno ti podaci čine kompletan kontekst podataka za informacioni sistem.

Na primer, za jedan sistem prodaje, objekti su KUPCI, PROIZVODI, PRODAJNE PROGNOZE, PRODAJNI REGIONI, PORUDŽBINE I KOMERCIJALISTI. Za date objekte treba prikupiti i uskladištiti podatke. Slično tome, relacije se mogu izraziti jednostavnim, deklarativnim rečenicama kao što su:

- Kupci dostavljaju porudžbine.
- Porudžbine prodaju proizvode.
- Kupci su locirani u prodajnim regionima.

Vlasnici sistema su veoma retko zainteresovani za detaljnije podatke u vezi objekata i relacija, sem ukoliko nisu i korisnici sistema.

Pogled korisnika sistema na sistem podataka

Korisnici informacionog sistema su eksperti za podatke koji opisuju poslovni sistem. Oni kao informacioni radnici svakodnevno prikupljaju, skladište, obrađuju, uređuju i koriste te podatke. Za njih su podaci smešteni po fasciklama, knjigama, organizovani po *spreadsheets* datotekama ili uskladišteni unutar baza podataka. Izazov u sistemskoj analizi jeste da se identifikuju i verifikuju zahtevi za podacima. Zahtevi za podacima su neophodni korisnički podaci koji se predstavljaju u obliku objekata, atributa (svojstava), relacija i pravila.

Razmotrimo sledeći primer. Vlasnik sistema želi da ima sve podatke o objektu KUPAC. Korisnik sistema će nas upozoriti o tome da treba razlikovati POTENCIJALNE KUPCE, STVARNE KUPCE I NEAKTIVNE KUPCE, zbog različitih tipova podataka koji opisuju svaki tip kupca. Takođe, korisnik sistema će nam reći koji su to podaci koji se moraju memorisati za svaki tip kupca. Na primer, objekat AKTIVAN KUPAC će imati sledeća svojstva: šifra kupca, naziv, adresa, kredit i tekući bilans kupca. Za opisivanje zahteva za podacima koristi se model podataka, koji će detaljnije biti objašnjen u ovoj knjizi.

Pogled projektanta sistema na sistem podataka Dok korisnici sistema definišu zahteve za podacima, projektanti sistema prevode te zahteve u računarske datoteke i baze podataka, koje će potom biti dostupne putem informacionog sistema. Projektanti sistema projektuju informacioni sistem pomoću već zadate informacione tehnologije. Najčešće su to već standardizovani alati kao što su *Oracle* ili *DB2*, dok za PC-jeve, alati baze podataka koji se najčešće koriste su *Access* ili *dBASE*. Pogled projektanta sistema na sistem podataka je u obliku šeme baze podataka

Pogled graditelja sistema na sistem podataka

Graditelji sistema su najbliži korisnici tehnologije baze podataka. Oni moraju da predstavljaju podatke u veoma preciznoj jezičkoj formi. Najkorišćeniji standardni upitni jezik koji omogućava komunikaciju sa bazom podataka jeste *SQL* (od početnih slova engleskih reči: *Structured Query Language*).

Fundamentalni blokovi procesa

Kada inženjeri projektuju nov proizvod, taj proizvod bi trebao da obezbedi odgovarajući nivo funkcionalnosti ili usluge. Potencijalni kupci definišu željenu funkcionalnost proizvoda, a inženjer kreira dizajn proizvoda kako bi obezbedio datu funkcionalnost. Neke procese obavljaju ljudi, a neke mašine, uključujući i računare. Neki procesi se odvijaju repetativno, dok se drugi odvijaju ne tako često ili čak retko. Cilj je da se automatizuju odgovarajući procesi pomoću softver tehnologije (Slika 6 – druga kolona). Kao što se vidi sa slike 6, različiti *stakeholders*-i imaju različite poglede na procese sistema.

4. RELACIONE BAZE PODATAKA

U relacionom sistemu je baza podataka izražena skupom vremenski promenljivih "relacija". SUBP omogućava sve potrebne operacije sa relacijama i pri tom dozvoljava korišćenje najrazličitijih kombinacija postojećih relacija, da bi time korisniku omogućio traženje odgovora na sva pitanja koja imaju smisla s obzirom na sadržaj baze podataka.

Svaka relacija ima sva svojstva skupa. Osnovno svojstvo svakog skupa je da se elementi koje on sadrži međusobno razlikuju. Tako se i svi redovi relacije međusobno razlikuju. To znači da u relaciji uvek postoji neki atribut (ili neka kombinacija atributa), koji omogućava jednoznačnu identifikaciju svakog retka. Takav atribut (odnosno takva kombinacija atributa) koristi se kao "ključ" relacije.

Kolona (atribut) u relaciji je uvek homogena u tom smislu i sadrži samo vrednosti iz jednog domena. Redosled redova u relaciji nije važan, jer se redovi ne identifikuju na osnovu svog položaja već na osnovu ključa. Takođe, ni redosled kolona nije važan, jer smisao kolone nije određen relativnom pozicijom kolone u relaciji, već imenom atributa.

Teoretske postavke relacionih modela

Struktura relacionih modela je jednostavna i prihvatljiva za svakog korisnika, jer relaciona baza podataka predstavlja skup tabela. Same relacije iz skupa tabela (baze podataka) generišu izlaz, takođe, u obliku jednostavne i lako prihvatljive tabele.

S druge strane, jednostavna je i formalno matematička interpretacija tabela, tj. tabela se može definisati kao matematička relacija.

Da bi se pristupilo razmatranju relacionih modela potrebno je dati definicije za:

- kartenzijanski (Dekartov) proizvod,
- relacije,
- domen relacije,

- stepen relacije,
- kardinalnost relacije,
- atribute relacije,
- ključeve,
- primarni ključ,
- spoljne ključeve,
- bazne relacije,
- izvedene relacije,
- šemu relacione baze podataka i
- null vrednost.

Kartezijanski proizvod od n skupova D1 x D2 x...Dn je skup svih mogućih uređenih n-torki

<d1,d2,....dn> tako da je d1 ∈ D1, d2 ∈ D2 ...dn ∈ Dn

Primer: Data su dva skupa brojeva:

D1 = š1,2,3,4ć i D2 = š4,5ć

D1 x D2 = š<1,4>,<1,5>,<2,4>,<2,5>,<3,4>,<3,5>,<4,4>,<4,5>

Relacija se definiše kao podskup Dekartovog proizvoda nad n-skupova, tj. podskup sadrži one n-torke Dekartovog proizvoda koje zadovoljavaju zadatu relaciju.

R ⊆ D1 x D2 x ...x Dn

Primer: Definisati za prethodni primer nad skupovima D1 i D2 relaciju:

R: A x B =š <d1,d2>đ d1 = d2/2ć

R = š<2,4>ć

Domen relacije R definisan je okvirima skupova D1, D2,...Dn.

Stepen relacija je definisan brojem domena nad kojima je definisana neka relacija. Razlikuje se unarni stepen relacije nad jednim domenom, binarni nad dva i n-arni nad n-domena.

Kardinalnost relacije je broj n-torki u relaciji.

Po definiciji, Dekartov proizvod predstavlja skup uređenih n-torki i redosled elemenata u jednoj n-torki je bitan. Ako bi se vrednostima elemenata n-torki pridružila semantička imena domena, redosled elemenata u n-torkama postaje beznačajan.

Na primer, za definisane domene:

```
SIFRAR = š827369,827499,827521ć
PREZIME= šSTEVIC,ALAGIC,VUKICć
RUKOV = š827902,827698,ć
```

relacija se definiše ovako:

```
RADNIK ⊆ SIFRARxPREZIMExRUKOV =
š<827369,STEVIC,827902>,
<827499,ALAGIC,827698>,
<827521,VUKIC,827698>ć
```

Prvi element trojke uzima vrednost iz prvog, drugi iz drugog, a treći iz trećeg skupa.

Ako bi se vrednostima elemenata u n-torkama pridružila imena domena, redosled elemenata u n-torkama je beznačajan:

```
RADNIK ⊆ SIFRAR x PREZIME x RUKOV =
š <SIFRAR:827369,PREZIME:STEVIC, RUKOV:827902>,
<PREZIME:ALAGIC, SIFRAR:827499, RUKOV:827698>,
<RUKOV:827698,SIFRAR:827521,PREZIME:VUKIC>ć
```

Atributi relacije

Atributi relacije su imenovani domeni sa imenom koje definiše ulogu domena u relaciji. Atributi relacije RADNIK su SIFRAR, PREZIME, RUKOV i dr.

Ovako definisan koncept atributa omogućuje predstavljanje relacija kao tabela, pa se relacija RADNIK može predstaviti sa:

SIFRAR	PREZIME	RUKOV
827369	STEVIC	827902
827499	ALAGIC	827698
827521	VUKIC	827698

Razlika između relacije i tabele je u tome što je relacija skup, a svaka tabela to nije. Stoga se definišu uslovi koje tabela mora da zadovolji da bi bila relacija:

- ne mogu postojati duple vrste tabela;
- redosled vrsta nema značaja;
- redosled kolona nema značaja;
- nisu dozvoljeni atributi ili grupe atributa sa ponavljanjem.

Prikazana tabela zadovoljava uslove da jedna tabela bude relacija. Ako je zadovoljen poslednji uslov, onda se kaže da se tabela nalazi u prvoj normalnoj formi.

Ključevi

Ključevi se definisu kao jedan ili više atributa čija vrednost jedinstveno identifikuje jednu n-torku

u relaciji, tj. jedan red u tabeli. Ako je ključ definisan samo jednim atributom, onda je to prost ključ (npr., u relaciji RADNIK ključ je SIFRAR). Ako je ključ definisan sa više atributa, onda je to složeni ključ (npr., relacija CERTIFIKAT-ključ je SIFRAR, SIFRAJ).

Ako se definiše relacija R, onda se može reći da ključ relacije R predstavlja kolekcija K njenih atributa koja zadovoljava :

- osobinu jedinstvenosti i
- osobinu neredundantnosti.

Osobina *jedinstvenosti* je vezana za ograničenje da ne postoje bilo koje dve n-torke sa istom vrednošću K.

Osobina *neredundantnosti* se odnosi na gubljenje osobina jedinstvenosti ako se bilo koji atribut izostavi iz K.

U jednoj relaciji može postojati više različitih kolekcija K atributa koje zadovoljavaju definiciju ključa; sve se one nazivaju kandidati za ključ.

Primarni ključ je jedan od izabranih kandidata za ključ koji služi za identifikaciju n-torke relacije. Ostali kandidati za ključ postaju alternativni ključevi.

Atributi koji učestvuju u ključevima nazivaju se ključni atributi, dok se ostali nazivaju opisni atributi.

Spoljni ključ ili preneseni ključ je atribut ili grupa atributa u relaciji R, čija se vrednost koristi za povezivanje sa vrednošću primarnog ključa u nekoj relaciji R2. Spoljni ključevi i njima odgovarajući primarni ključevi definisani su nad istim domenom. Dakle, spoljni ključevi služe da se uspostave veze između relacija u relacionoj bazi podataka. Na primer, u relaciji RADNIK spoljni ključ SIFRAO vezuje sa relacijom ODELJENJE.

Bazna relacija je relacija koja se ne može izvesti iz ostalih relacija u relacionoj bazi podataka.

Izvedena relacija je relacija koja se izvodi iz skupa baznih i izvedenih relacija, i to preko operacija koje se definišu nad relacijama.

Kako se u tekstu koriste termini vezani za relacije, tabele i klasičnu obradu podataka, to se u sledećoj tabeli definišu njihove veze:

RELACIONA	TABELARNA	KLASICNA OBRADA
DOMEN	SKUP VREDNOSTI	TIP
ATRIBUT	KOLONA	POLJE
N-TORKA	RED	SLOG (RECORD)
PRIMARNI KLJUC	IDENTIFIKATOR REDA	JEDINSTVEN KLJUC
RELACIJA	TABELA	DATOTEKA
STEPEN	BROJ KOLONA	BR.POLJA U SLOGU
KARDINALNOST	BROJ REDOVA	BR.SLOGOVA U DAT.

Ekstenzija relacije je skup svih n-torki date relacije, odnosno predstavljanje tabele navođenjem svih vrsta. Tabela RADNIK predstavlja ekstenziju odgovarajućih relacija.

Intenzija relacije je generalizacija ekstenzije; ona je vremenski nepromenljiva i označava se na sledeći način:

```
RADNIK (#SIFRAR,PREZIME,RUKOV)
```

(Ispred zagrade je naziv relacije, u zagradi su navedeni atributi, a primarni ključ je obeležen masnim otiskom.)

Šema relacione baze podataka

Šema relacione baze podataka je predstavljanje strukture relacione baze kao skupa intenzija relacija.

```
RADNIK (#SIFRAR, PREZIME, RUKOV, DATUMZ, PLATA,
 STIMUL,*SIFRAO)
ODELJENJE (#SIFRAO, NAZIVO, MESTO)
```

Null vrednosti

Null vrednosti se koriste da se označe još nepoznate vrednosti za neki atribut ili neprimenjivo svojstvo za neki objekat ili vezu koju predstavlja tabela (označavaće se znakom ?).

Relaciona algebra

Relaciona algebra definiše skup operacija pomoću kojih se, na proceduralan način, može dobiti željena relacija, tj.tabela iz skupa datih relacija.

Relacija se definiše kao podskup Dekartovog proizvoda i može se tretirati kao skup n-torki.

Za nivo relacione algebre operacije: unija, presek i diferencije nad relacijama R1 i R2 moraju zadovoljiti uslov kompatibilnosti (union compatible relations), tj. relacije R1 i R2 moraju imati isti broj atributa (isti stepen), a odgovarajući atributi su definisani nad istim domenom.

Na nivou relacione algebre, operacije su:

- unija,
- diferencija,
- presek,
- Dekartov proizvod,
- projekcija,

- selekcija (restrikcija),
- spajanje (join),
- deljenje,
- operacije sa null vrednostima.

Unija

Ako su date relacije R1 i R2 koje zadovoljavaju uslove kompatibilnosti, onda je rezultat operacije unije

```
R3 = R1 U R2
```

relacija R3 koja sadrži sve n-torke koje se pojavljuju bilo u R1, bilo u R2.

Diferencija

Ako su date relacije R1 i R2 koje zadovoljavaju uslov kompatibilnosti, onda rezultat operacije diferencije

```
R3 = R1 - R2
```

predstavljaju n-torke relacije R1, koje nisu istovremeno i n-torke relacije R2.

Presek

Ako su date relacije R1 i R2 koje zadovoljavaju uslov kompatibilnosti, onda je rezultat operacije preseka

```
R3 = R1 ∧ R2
```

relacija R3 koja sadrži n-torke koje se pojavljuju u obe relacije R1 i R2.

Kada se izdvoje operacije unije i diferencije koje su pogodne za ažuriranje baze podataka, operacijom unije se mogu u neku relaciju dodati nove n-torke, a operacijom diferencije se mogu iz neke relacije izbaciti neželjene n-torke. Izmena vrednosti pojedinih atributa u nekoj relaciji vrši se uzastopnom primenom operacije diferencije, pomoću koje se izbaci cela n-torka u kojoj se nalazi posmatrani atribut, a zatim se operacijom unije "vraća" izbačena n-torka sa promenjenom vrednošću atributa.

Dekartov proizvod

Dekartov proizvod se može primeniti na bilo koje dve relacije. Rezultat je ove operacije

```
R3 = R1 x R2
```

relacija R3 čije su n-torke svi "parovi" koje čine jedna n-torka relacije R1 i jedna n-torka relacije R2.

Projekcija

Operacija projekcije se definise kao unarna operacija koja iz neke relacije selektuje skup navedenih atrubuta, odnosno "vadi" vertikalni podskup iz odgovarajuće tabele.

Formalno, projekcija se definiše na sledeći način:

Neka je R (A1,A2,...,An) relacija, a X podskup njenih atributa.

Ako se sa Y označi komplement šA1,A2,...Anć-X

rezultat operacije projekcije relacije R po atributima X glasi:

PŠXć, = šx đ tako da postoji y da je <x,y> ∈ Rć

Operacijom projekcija eliminišu se duplikati n-torki.

Selekcija (Restrikcija)

Selekcija je unarna operacija koja iz date relacije selektuje n-torke koje zadovoljavaju zadati uslov ("vadi" horizontalni podskup tabele).

Formalno se definiše na sledeći način:

Dati su relacija R (A1,A2,...An) i predikat P, definisan nad njenim atributima. Rezultat operacije selekcije glasi:

SŠPćR = šx đ x ∈ R i P (x)ć

Spajanje (Join)

Spajanje je binarna operacija koja spaja dve relacije na taj način da se u rezultatu pojavljuju oni parovi n-torki jedne i druge relacije koji zadovoljavaju uslov zadat nad njihovim atributima.

Formalno se definiše na sledeći način:

Date su relacije R1 (A1,A2,...,An) i R2 (B1,B2,...,Bm) i predikat θ, definisan nad njihovim atributima.

Ako se sa X i Y obeleže skupovi atributa relacija R1 i R2, respektivno, rezultat operacije spajanja ovih relacija (tzv. θ-spajanje) glasi ovako:

R1ŠxθćR2 = š<x,y> đ x ∈ R1 AND y ∈ R2 AND θ (x,y)ć

Oznaka xθ za operaciju spajanja ukazuje na činjenicu, očiglednu iz definicije θ spajanja, da ova operacija nije primitivna operacija relacione algebre, već da se može izvesti uzastopnom primenom operacije Dekartovog proizvoda (x) i selekcije po predikatu θ iz tako dobijene relacije.

Ako je predikat θ definisan sa Ak = Bj, s tim da su i atributi Ak i Bj definisani nad istim domenima, tada se takvo spajanje naziva ekvispajanje.

Očigledno je da se u rezultatu ekvispajanja uvek pojavljuju dve iste kolone. Ako se jedna od te dve kolone izbaci, takvo spajanje se naziva prirodno spajanje.

Deljenje

Deljenje je operacija pogodna za upite u kojima se javlja reč "svi".

Formalno se definiše na sledeći način:

Neka su A (X,Y) i B (Z) relacije, gde su X, Y i Z skupovi atributa takvi da su Y i Z jednakobrojni, a odgovarajući domeni su im jednaki.

Rezultat operacije deljenja je

```
AŠY ) ZĆB = R (X)
```

gde n-torka X uzima vrednosti iz A.X, a par <x,y> postoji u A za sve vrednosti y koje se pojavljuju u B (Z).

Operacija deljenja nije primitivna operacija relacione algebre, već se može izvesti na sledeći način:

```
A(X,Y)ŠY ) Z,B(Z) = PŠX,A - PŠX,((PŠX,A x B)- A)
```

Objašnjenje:

PŠXĆA daje sve n-torke koje mogu da učestvuju u rezultatu.

PŠXĆA x B daje relaciju u kojoj se za svaku vrednost z iz B pojavljuju parovi <x,z> sa svim vrednostima x.

PŠXĆA x B) - A ne sadrži ni u jednom paru <x,z> one vrednosti x za koje u relaciji A, kao vrednosti y, postoje sve vrednosti z.

Kompletan izraz, prema tome, jeste relacija koja sadrži one n-torke x za koje postoje u paru <x,y>, kao vrednosti y, sve vrednosti z.

Relaciona algebra je proceduralni jezik. Pomoću operacija relacione algebre sačinjava se procedura koja dovodi do odgovora na postavljeni upit.

Mada je proceduralni jezik, relaciona algebra je znatno moćnija od klasičnih programskih jezika, koji su, takođe, proceduralni. Razlog za to je što operand relacione algebre predstavlja relaciju (cela tabela), a operand operacija sa datotekama u klasičnim jezicima je rekord (vrsta tabele).

Može se pokazati da se bilo koja relacija (bilo koji upit), izvodljiva iz skupa datih relacija, može dobiti procedurom (algoritmom) od tri koraka:

1. Dekartov proizvod svih relacija koje se koriste;
2. selekcija n-torki za koje se predikat selekcije sračunava u T;
3. projekcija rezultata po atributima koji se prikazuju.

Ova procedura se može iskazati jednim opštim izrazom relacione algebre:

```
PŠA1, A2, ..., An, (SŠP, (R1 x R2 x ... x Rm))
```

Međutim, ovakvo izvođenje operacija bilo bi veoma "skupo", jer bi rezultat Dekartovog proizvoda relacija R1, R2, ..., Rm bio relacija sa k1 x k2 x ... x km n-torki, gde su sa ki označene kardinalnosti odgovarajućih relacija. Zbog toga se i definiše operacija spajanja, koja istovremeno obavlja Dekartov proizvod i selekciju, smanjujući na taj način broj n-torki koji se pretražuje. Pored toga, očigledno je da su znatno efikasniji algoritmi sa više koraka, u kojima bi se prvo vršile sve selekcije koje se mogu izvesti na pojedinačnim relacijama, zatim projekcije, pa tek onda spajanja.

Operacije sa nula vrednostima

Navedene operacije relacione algebre nisu uzimale u obzir nula vrednosti, koje mogu postojati u relacijama. Podrazumevalo se da se vrednosti predikata sračunavaju na osnovu standardnih dvovrednosnih tablica istinitosti. Isto tako, bez dvoumljenja je bilo moguće da se odrede duplikati n-torki, odnosno kardinalnost pojedinih skupova. Pojavljivanje nula vrednosti u relacionoj bazi podataka zahteva da se proširi skup definisanih operacija koje bi na neki način uključile i nula vrednosti. Osnovne postavke za operacije sa nula vrednostima su sledeće:

(1) Tablice istinitosti trovrednosne logike:

AND	T	?	F
T	T	?	F
?	?	?	?
F	F	F	F

OR	T	?	F
T	T	T	T
?	T	?	?
F	T	?	F

NOT	
T	F
?	?
F	T

(Znak ? predstavlja nula vrednost, a može se u okviru tablica istinitosti čitati i kao "možda".)

(2) Sračunavanje aritmetičkih formula. Neka " označava neki od aritmetičkih operatora (+, -, *, /) i neka su x i y dve numeričke vrednosti. Vrednost aritmetičkog izraza "x " y" je, po definiciji, nula vrednost, ako bilo x, bilo y, bilo oba dobiju nula vrednost.

(3) Skupovi sa nula vrednostima. Da li kolekcija š1,2,3,?ć, predstavlja skup? Ako ? uzme vrednost 1,2 ili 3, onda nije, ili se može tretirati kao skup sa kardinalnošću 3. Ako ? nije ni 1, ni 2, ni 3, onda gornja kolekcija predstavlja skup sa kardinalnošću 4. To pokazuje da postoje različite nula vrednosti: nula vrednost koja nije 1, nula vrednost koja nije 2, zatim nula vrednost koja nije

ni 1, ni 2, i tako dalje. Operacije koje bi uzele u obzir različitost nula vrednosti bile bi veoma složene. Zbog toga se operacije definišu jednom vrstom nula vrednosti, a ostale nula vrednosti se tretiraju kao duplikati.

Imajući u vidu ove opšte postavke, primeri nekih ranije definisanih operacija na relacijama koje poseduju nula vrednosti izgledaju ovako:

UNIJA

```
R3 = R1 U R2

R1   A   B       R2   A   B       R3   A   B
     a   b            x   y            a   b
     a   ?            ?   b            a   ?
     ?   b            ?   ?            ?   b
     ?   ?                             ?   ?
                                       x   y
```

DIFERENCIJA

```
R4 = R1 - R2

R4   A   B
     a   b
     a   ?
```

DEKARTOV PROIZVOD

Dekartov proizvod ostaje neizmenjen.

SELEKCIJA

Operacija selekcije ostaje neizmenjena, selektuju se one n-torke za koje se odgovarajući predikat sračunava u T na osnovu trovrednosnih tablica istinitosti. Ova operacija se često zove i "TRUE SELECTION" (istinita selekcija).

```
SŠA = a,R1    A   B
              a   b
              a   ?
```

PROJEKCIJA

Uzima se u obzir da su nula vrednosti duplikati (jedna vrsta nula vrednosti):

```
PŠA,R1        A
              a
              ?
```

SPAJANJE

Operacija spajanja ostaje neizmenjena, jer operacije Dekartovog proizvoda i selekcije ostaju

neizmenjene. U rezultatu se pojavljuju one n-torke za koje se predikat spajanja sračunava u T na osnovu trovrednosnih tablica istinitosti (TRUE TETA JOIN - istinito θ spajanje).

DODATNE OPERACIJE

Zbog postojanja nula vrednosti u bazi podataka, neophodno je da se definiše logička funkcija "IS_NULL", čiji argument može da bude neki skalarni izraz, a koja se sračunava u T ako je vrednost tog argumenta nula vrednost, a inače uzima vrednost F.

MOŽDA SELEKCIJA (MAYBE_SELECT)

Selektuju se one n-torke relacije za koje se predikat selekcije sračunava u nula vrednost na osnovu trovrednosnih tablica istinitosti.

```
MAYBE_SELECTŠAĆR3   A   B
                    ?   b
                    ?   ?
```

MOŽDA SPAJANJE (MAYBE_JOIN)

U rezultatu spajanja se pojavljuju one n-torke za koje se predikat spajanja sračunava u nula vrednost na osnovu trovrednosnih tablica istinitosti.

```
Ra = RbŠMAYBE_JOIN A = DĆRc

Rb  A   B   Rc  C   D   E   Ra  A   B   C   D   E
    a1  b1      c1  ?   e1      a1  b1  c1  ?   e1
    a2  b2      c2  d2  e2      a1  b1  c3  ?   e3
    ?   b3      c3  ?   e3      a2  b2  c1  ?   e1
                                a2  b2  c3  ?   e3
                                ?   b3  c1  ?   e1
                                ?   b3  c2  d2  e2
                                ?   b3  c3  ?   e3
```

SPOLJNO SPAJANJE (OUTER_JOIN)

Neka su date relacije R1 (A,B) i R2 (C,D). Pretpostavka je da se vrši operacija ekvispajanja ovih relacija R1ŠA = CĆR2. Ako je PŠAĆR1 # PŠCĆR2 (projekcije relacija po atributima spajanja su različite), tada će se u rezultatu spajanja "izgubiti" neke n-torke relacija R1 i R2. Ako se takvo gubljenje informacija ne želi, SPOLJNO_SPAJANJE ih može sačuvati na taj način što se u rezultat dodaju i ove n-torke, i to tako što za n-torke relacije R1 atributi C i D uzimaju nula vrednosti, a za n-torke relacije R2 atributi A i B uzimaju nula vrednosti.

Primer: R1ŠOUTER_JOIN A = CĆR2

```
R1  A   B       R2  C   D       R3  A   B   C   D
    1   2           3   4           2   1   2   2
    2   1           2   2           1   2   ?   ?
    4   ?                           ?   ?   3   4
                                    4   ?   ?   ?
```

SPOLJNA UNIJA (OUTER_UNION)

Operacija unije se, kao što je rečeno, može izvesti samo nad relacijama koje zadovoljavaju kriterijume kompatibilnosti (da su istog stepena i da su im odgovarajući atributi definisani nad istim domenima). Dodavanjem novih atributa i postavljanjem njihovih vrednosti na nula vrednosti mogu se uvek dve nekompatibilne relacije učiniti kompatibilnim. Spoljna unija podrazumeva da su, na taj način, dve nekompatibilne relacije učinjene kompatibilnim, pa je tada izvršena operacija unije.

```
Primer: R3 = R1 OUTER_UNION R2
```

R1	A	B	C
	a1	b1	c1
	a2	b1	c2

R2	A	D
	a1	4
	a1	7
	a2	5

R3	A	B	C	D
	a1	b1	c1	?
	a2	b1	c2	?
	a1	?	?	4
	a1	?	?	7
	a2	?	?	5

Relacioni račun

Relacioni račun je neproceduralni način iskazivanja operacija, gde se pomoću konstrukcije predikatskog računa prvog reda, u kome su promenljive date kao n-torka relacija ili domeni relacija, koriste za definisanje osobina relacija koje se žele dobiti.

Relacioni račun n-torki

Relacioni račun n-torki je predikatski račun prvog reda u kome domeni promenljivih predstavljaju n-torke relacija date baze podataka.

Osnovni pojmovi predikatskog računa su:

- afirmativna rečenica, koja ima smisla i koja je istinita ili neistinita i naziva se sud;
- afirmativna rečenica, koja ima smisla i koja sadrži jedan ili više promenljivih parametara i postaje sud uvek kada parametri iz rečenice dobiju konkretnu vrednost, naziva se predikat; broj parametara u predikatu se naziva dužina predikata (primer predikata je x + y # 1);
- predikatski ili kvantifikatorski račun, kao matematička teorija, čiji su objekti formule koje predstavljaju predikate; simboli koji se koriste da označe neki sud nazivaju se atomskim formulama ili atomima. Atomi u relacionom računu n-torki su:
- $A \in$, R gde je x n-torka promenljiva, a R relacija, odnosno promenljiv x uzima vrednosti iz skupa n-torki relacije R;
- x.A θ y.B gde su x i y promenljive (n-torke), A i B su atributi relacija R1 i R2 iz čijih n-torki, respektivno, promenljive x i y uzimaju vrednosti ($x \in$ R1, $y \in$ R2), a θ je operacija poređenja definisana nad domenom atributa A i B (A i B moraju biti definisani nad istim domenom);
- x.A θ c gde su x, A i θ kao i u prethodnom stavu, a c je konstanta koja ima isti domen kao i

A.

Formule se formiraju od atoma preko sledećih pravila:

- atom je formula;
- ako je P1 formula, tada su formule i NOT P1 i (P1);
- ako su P1 i P2 formule, tada su formule i P1 AND P2 i P1 OR P2;
- ako je P1(s) formula koja sadrži neku slobodnu promenljivu s, tada su i ∃ s (P1(s)) i ∀ s (P1(s), takođe, formule (∃ - "postoji", egzistencijalni kvantifikator, ∀ - "za svako", univerzalni kvantifikator).

Jedna promenljiva u nekoj formuli se može pojaviti više puta. Promenljive mogu biti "slobodne" i "vezane". Vezana promenljiva u formuli je neka vrsta "prividne" (dummy) promenljive i ima ulogu da poveže promenljivu iza kvantifikatora sa promenljivima u zoni dejstva kvantifikatora. Drugim rečima, vezana promenljiva u je (∃ u), (∀ u) ili (∃ u) A ili (∀ u) A, gde je A formula u kojoj se pojavljuje u. Sve promenljive koje u nekoj formuli nisu vezane, slobodne su u toj formuli.

Na primer, u formuli: ∃x (x > 3), x je vezana promenljiva, pa je gornja formula ekvivalentna sa ∃y (y > 3).

U formuli: ∃x (x > 3) AND x < 0, prvo pojavljivanje promenljive x je vezano, a drugo slobodno, pa je ova formula ekvivalentna sa ∃y (y > 3) AND x < 0.

Neka su R1, R2, ..., Rn relacije u nekoj bazi podataka. Neka su A, B ..., C atributi ovih relacija, respektivno, i neka je f - formula. Opšti izraz relacionog računa n-torki je tada:

```
t ∈ R1, u ∈ R2, ..., v ∈ Rn
t.A, u.B, ..., v.C GDE_JE f
```

(Prikazuju se vrednosti atributa A relacije R1, atributa B relacije R2, ... i atributa C relacije Rn za one n-torke koje zadovoljavaju uslov definisan formulom f.)

Relacioni račun domena

U relacionom računu domena promenljive uzimaju vrednosti iz nekih domena definisane relacione baze podataka. Ovde se, pored navedenih, definiše još jedna atomska formula, tzv. uslov članstva (membership condition). Uslov članstva ima sledeći oblik:

```
R (term, term, ...)
```

gde je R ime neke relacije, a svaki term ima oblik A:v, gde je A neki atribut relacije R, a v je ili promenljiva ili konstanta.

Uslov članstva se izačunava u T (TRUE) ako postoji n-torka u relaciji R, koja ima zadate vrednosti navedenih atributa.

Opšti izraz relacionog računa domena je:

```
x, y, ...,z GDE_JE f
```

gde su `x,...,z` promenljive, a f je formula koja uključuje i uslov članstva.

Relaciona algebra i relacioni račun (i račun n-torki i račun domena) fundamentalno su ekvivalentni jedno drugom. E.F Codd je pokazao da se svaki izraz relacionog računa može svesti na semantički ekvivalentan izraz u relacionoj algebri, a Ullman - da se svaki izraz u relacionoj algebri može svesti na izraz relacionih računa, pa se na osnovu toga može zaključiti da si ova dva formalizma logički ekvivalentna.

Sistemi za upravljanje bazama podataka (SUBP)

Sistemi za upravljanje bazama podataka imaju dve osnovne funkcije. Prva funkcija se koristi za memorisanje i održavanje podataka, koji izražavaju svojstva tabela (objekata posmatranja). Za izvođenje ove funkcije koriste se jezik za definisanje podataka i struktura podataka (Data Definition Language ili DDL).

Druga funkcija se koristi za kontrolisan pristup do memorisanih podataka i prikazivanje podataka (vrednosti svojstava tabela) na zahtev korisnika. Za izvođenje ove funkcije koristi se jezik za manipulaciju podacima (Data Manipulation Language ili DML).

Treba istaći da SUBP mora vršiti nadzor nad simultanim korišćenjem baze podataka. U načelu, svakom je korisniku data mogućnost da vrši manipulaciju sa memorisanim podacima. Tako, može se dogoditi da više korisnika istovremeno pokuša modifikovanje istih podataka. Stoga SUBP mora sinhronizovati simultane zahteve više korisnika, da ne bi došlo do gubljenja podataka.

Kada se to ima u vidu, rad sa SUBP treba da omogući:

- menjanje postojećih podataka u okviru baze podataka,
- brisanje postojećih redova,
- dodavanje novog reda,
- traženje (izdvajanje) konkretnog reda i
- pretraživanje baze podataka.

Mora se naglasiti da se sve promene u vezi sa poljem (unošenje novih podataka, menjanje postojećih i brisanje) moraju sprovoditi u okviru reda.

Postupak *menjanja* polja je sledeći:

- red u kome se vrši izmena podataka pronalazi se u memoriji pomoću adrese ili ključa;
- red se iz eksterne memorije prenosi u operativnu memoriju;
- vrši se izmena sadržaja datih polja;
- red se ponovo zapisuje u eksternu memoriju.

Postupak *brisanja* već postojećih redova zahteva pažnju - da se brišu pravi redovi, tj. da se poštuju pravila vezana za integritet baze podataka, kao i referencijalni integritet.

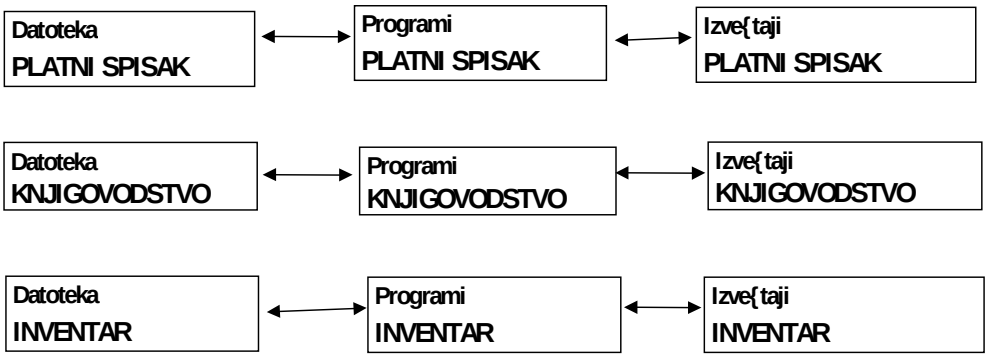
Postupak *dodavanja* novih redova zahteva obezbeđenje potrebnog memorijskog prostora i poštovanje integriteta baze podataka, kao i referencijalnog integriteta.

Traženje podrazumeva traženje konkretnog reda koje se obavlja preko primarnog ključa. Traženje je uspelo ako je vrednost primarnog ključa reda identična vrednosti ključa datog u argumentu.

Pretraživanje se vrši putem argumenta koji je dat kao LOGIČKI izraz. Sastavlja se lista zahtevanih svojstava i kriterijuma po kojima se vrši pretraživanje tabela i izdvajaju svi oni redovi koji u potpunosti udovoljavaju zahtevima (videti glavu 4.3).

Razlike između SUBP i datoteka

Korišćenjem klasičnih programskih jezika treće generacije svaki programer definiše programe i datoteke prema svojim potrebama i shvatanjima. Kao što se može videti na sledećoj slici parcelizovani način rada dovodi do redundanse (ponavljanja) sličnih datoteka u različitim aplikacijama.



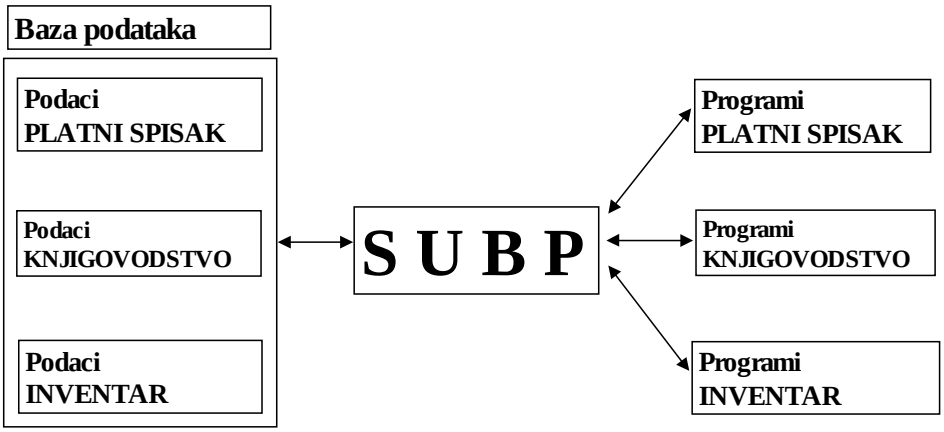
Slika 4. Primer korišćenja datoteka

Datoteke nastaju kao produkt izolovano razvijenih programa (aplikacija), korišćenjem programskih jezika. Kao što se može videti na slici 4.4, datoteke su izrazito usklađene s programima i prema njima imaju vlasnički odnos. Međutim, nisu prilagođene za upotrebu kod novih programa, jer oni nisu bili uzeti u obzir pri definisanju datoteke, a mogli bi uspešno da koriste neke podatke. Ovaj problem se polovično rešava promenom dizajna datoteke ili definisanjem dodatne datoteke za potrebe novog programa. Međutim, negativna reakcija je pojava višestrukog memorisanja podataka, ili tzv. redundanse podataka.

Za razliku od rada sa datotekama, korišćenje SUBP omogućuje da *neprogrameri* mogu pristupiti podacima i njima manipulisati:

- *direktnim* radom sa računarom svakog korisnika,
- *jednostavnim* definisanjem obrade (obradu definišu osobe koje se bave fizikom problema, a ne obradom podataka).

Dakle, SUBP treba da omogući svim ovlašćenim korisnicima korišćenje zajedničkih podataka (slika 4.5), skladištenje podataka sa minimalnom redundansom, logičku i fizičku nezavisnost programa od podataka, jedinstveno komuniciranje sa bazom podataka preko jezika bliskih korisniku. Nezavisnost programa i podataka se postiže kada se može dopunjavati i modifikovati struktura baze podataka bez posledica po postojeće korisnikove programe. Jezik blizak korisniku je tzv. SQL upitni jezik.



Slika 5. Primer korišćenja SUBP

Kriterijumi za ocenu relacionih SUBP-a

Definisano je 12 pravila koji omogućuju da se oceni do kog je nivoa neki SUBP relacioni.

1. Struktura SUBP se predstavlja samo tabelama.
2. Svaki podatak u bazi podataka dostupan je preko kombinacije imena tabele, vrednosti primarnog ključa i imena kolone, bez unapred zadatih pristupnih puteva i bez rekurzije ili iteracije.
3. Specifičan indikator, različit od "blanka" ("praznog") niza karaktera, nule ili bilo kog broja, koristi se za predstavljanje nula vrednosti, bez obzira na tip podatka.
4. SUBP treba da poseduje katalog (rečnik) podataka, koji se logički predstavlja na isti način kao i sama baza podataka.
5. SUBP mora posedovati bar jedan jezik čije se naredbe mogu izraziti kao niz karaktera sa dobro definisanom sintaksom i koji podržava: (1) definiciju podatka, (2) definiciju pogleda, (3) manipulaciju podatka, interaktivno i kroz programe, (4) definiciju pravila integriteta, (5) autorizaciju (sigurnost), (6) granice transakcija (BEGIN, COMMIT, ROLLBACK).
6. SUBP treba da poseduje efikasan algoritam pomoću koga može da odredi za svaki definisani pogled, u trenutku njegovog definisanja, da li se i koje operacije održavanja mogu da primene na taj pogled. Rezultat takvog algoritma se smešta u katalog baze podataka.
7. I nad tabelama i nad pogledima mogu se izvršavati ne samo operacije pretraživanja već i operacije održavanja baze podataka.
8. Aplikacioni program i interaktivna komunikacija treba da ostanu neizmenjeni kada se promeni fizička organizacija baze podataka.
9. Aplikacioni program i interaktivna komunikacija ostaju nepromenjeni kada se bilo koje promene, koje ne menjaju odgovarajući sadržaj tabele, unesu u baznu tabelu.
10. Pravila integriteta treba da se definišu u okviru definicije baze podataka i čuvaju se u rečniku podataka (Ne implementiraju se kroz aplikacione programe).
11. Sve navedene karakteristike treba da budu nezavisne od distribucije baze podataka.
12. Ako SUBP može da radi sa nekim jezikom treće generacije, u kome se obrađuje jedan red tabele u jednom trenutku vremena, kroz taj jezik se ne mogu zaobići pravila integriteta zadana preko samog relacionog SUBP.

5. KOMPONENTE KLIJENT / SERVER ARHITEKTURE

Komponente klijent/server (K/S) arhitekture su server baze podataka, Aplikacije klijenta, mreža u klijent/server sistemu i Neproceduralni upitni jezik - SQL

Server BP omogućuje upravljanje BP za pristup više korisnika,ali ono što je najinteresantnije je, zamena servera je bez izmena aplikacija. Server BP omogućuje kontrolu pristupa i bezbednost podataka, zaštita podataka i centralizovano obezbeđenje pravila integriteta.

Programer se ne bavi SUBP veš se svoja umeća koriste u okviru *aplikacija klijent* gde se radi sa redovima iz tabele tj. definiše seInterfejs prema korisniku GUI (Graphical User Interface). Aplikacija klijent Izvršavanje logike aplikacije izvodi provera ispravnosti ulaznih podataka i traženje i prijem podataka od servera.

Mreza i komunikacioni softver su sredstvo za prenos podataka, omogućuje razmenu poruka u mreži. Mreza i komunikacioni softver izvode podelu zadataka aplikacije između aplikacije klijent koja salje, prima i analizira podatke i servera BP koja obradjuje zahteve za pristupom BP.

Upravljanje podacima-Server BP

Definisanje načina upravljanja podacima je bitna funkcija organizacije podataka koja obuhvata:

- *skladištenje,* pod čime se podrazumevaju kontrola redosleda upisivanja podataka, načini pristupa i adresiranja podataka i način fizičkog predstavljanja podataka;
- *ponovno pristupanje,* tj. određivanje mesta nalaženja podataka (adresiranje), formiranje podataka i određivanje redosleda podataka;
- *kontrolu,* tj. unutrašnje regulisanje toka odvijanja postupka upravljanja podacima, određivanje prava pristupa podacima, čime osigurava podatke da ne dođe do gubitaka i obezbeđuje ažurnost podataka na sistemu.

Ova aktivnost se ostvaruje u okviru klijent/server-arhitekture i treba da se nalazi na jednoj ili više hardverskih platformi, gde server izvodi zajedničke servise za klijenta, tj. upravlja podacima preko ugrađenih poslovnih pravila i centalizovanih procedura. Klijent ograničeno i kontrolisano opterećuje server, distribuirano procesira informacije i zadržava samostalnost vezanu za rad u lokalu.

Upravljanje podacima treba da podrži:

- integritet baze podataka,
- transakcionu obradu podataka,
- sigurnost podataka,
- zaključavanje podataka i
- oporavak baze podataka
- logičko modeliranje podataka
- fizičko modeliranje podataka.

Integritet baze podataka

Integritet baze podataka treba da omogući tačnost, korektnost i konzistentnost podataka i da označi probleme zaštite baze podataka od pogrešnog ažuriranja (pogrešnih ulaznih podataka, greški operatera i programera, sistemskih otkaza).

U ovom delu je deo o problematici vezanoj za očuvanje integriteta pri izvodenju jedne transakcije. O transakcionoj obradi podataka biće više reči kasnije.

U okviru integriteta baze podataka treba definisati odgovarajuća pravila integriteta kojima se određuje koje uslove podaci u bazi podataka treba da zadovolje, kada se vrši provera i koje akcije treba preduzeti ako definisani uslovi nisu zadovoljeni.

Integritet baze podataka je definisan pravilima integriteta na serveru baze podataka i sadrži:

- *rečnik podataka,* tj. odgovarajuću meta-bazu podataka;
- *integritet domena* kojim se definiše dozvoljeni skup vrednosti;
- *integritet tabele* gde svaki red u tabeli mora da bude jedinstven;
- *referencijalni integritet* ili integritet relacija;
- *autoreferencijalni integritet* gde se u istoj tabeli definišu spoljni i originalni ključ;
- *poslovna pravila* ili tzv. korisnička pravila i ograničenja.

Rečnik podataka je baza podataka o bazi podataka (meta-baza podataka) i u njoj su opisi tabela,

relacija između tabela, kao i svi objekti tipa formi, izveštaja, upita, makroa i programa koji se čuvaju u meta-bazi, koja se logički predstavlja kao i sama baza podataka.

U rečnik podataka smeštaju se pravila integriteta koja su definisana pomoću nekog jezika visokog nivoa. Rečnik podataka je korektan pristup; no, međutim, neki SUBP ne podržavaju integritet na taj način, jer se podrška prebacuje u aplikacione programe, odnosno ostavlja se mogućnost da programer, uz pomoć generatora aplikacija, sam vodi računa o integritetu baze podataka.

Integritet domena je dozvoljeni skup vrednosti, gde svaka vrednost u polju mora da pripada domenu te kolone. Dakle, red se nalazi u tabeli ako su vrednosti u svim kolonama tog reda u domenu kolona, a integritet domena kolone je održan ako zapis može biti upisan u tabelu samo u slučaju da tip podatka u svakoj koloni odgovara dozvoljenom tipu za datu kolonu.

Integritet tabela je ispunjen ako je svaki red u tabeli jedinstven, tj. ako postoji jednoznačna identifikacija reda koji se definiše primarni ključem, u kojem se može nalaziti jedna ili više kolona. Kolona čija se vrednost koristi za identifikaciju ostalih kolona naziva se ključem. Postoje dve vrste ključeva: primarni i sekundarni.

Primarni ključ jednoznačno određuje red; stoga i ne mogu postojati dva reda sa istim vrednostima ključa. Ako u redu nema kolone koja bi jednoznačno odredila neki red, mogu se naći dva ili više koji zajedno određuju jednoznačnost. Tada se govori o *združenom* ključu. Sekundarni ključ je kolona po kojoj se vrši pretraživanje. Sekundarni ključ ne mora biti jednoznačan.

Referencijalni integritet ili integritet relacija omogućuje vezu između raznih kolona i tabela. Vrednosti u jednoj koloni ili grupa kolona *referišu* se vrednostima u drugoj koloni ili skupu kolona i pri tom se definiše za tabelu 'dete' preneseni ključ (foreign key), a za tabelu 'roditelj' - primarni ključ (primary key).

Autoreferencijalni integritet je definisan spoljnim i orginalnim ključem u istoj tabeli, gde je jedna kolona u tabeli povezana sa vrednostima u drugoj koloni iste tabele.

Poslovna pravila ili tzv. korisnička pravila su pravila za očuvanje integriteta podataka koja zadaje korisnik; npr., zabrana izmene podataka van radnog vremena. Uskladištena procedura je, takođe, primer za poslovna pravila, gde se definišu pravila za prevođenje skupa SQL iskaza i programski iskazi za kontrolu toka obrade. Takođe, poslovnim pravilima se deklarišu promenljive, kao i odgovarajući operatori za dodelu vrednosti. Uskladištena procedura automatski se "ispaljuje" pod određenim uslovima i nalazi se u serveru baze podataka, što omogućuje da se saobraćaj u mreži svede na minimum.

Ograničenjima se definiše integritet podataka između tabela; ta ograničenja su vezana za rang validnih vrednosti.

Transakciona obrada podataka

Transakcija je operacija kojom se izvodi serija izmena nad jednom ili više tabela, tj. transakcija je izvršenje neke logičke jedinice rada korisnika baze podataka. Osnovni cilj baze podataka je da omogući efikasnu obradu transakcija. Više izvršenja istog programa mogu se u sistemu odvijati konkurentno, svako od njih predstavlja transakciju.

Skup aktivnosti nad bazom podataka koje se izvršavaju po principu "sve ili ništa", tj. ili su sve aktivnosti uspešno obavljene ili je baza podataka ostala nepromenjena, atomski je skup aktivnosti. Očigledno je da "logička jedinica rada" predstavlja taj atomski skup aktivnosti, tj. "atomsku transakciju".

Definišu se dva tipa transakcija:

- DML transakcija kojom se povezuje veći broj DML naredbi, što se definiše još i kao eksplicitna transakcija i
- DDL transakcija kojom se izvodi samo jedna naredba, i to kao "automatska" (implicitna) transakcija.

Drugim rečima, transakcija predstavlja izvršenje jednog skupa operacija (npr., SELECT, UPDATE, INSERT i DELETE) nekog programa koji počinje sa prvom izvodljivom DML ili DDL naredbom (BEGIN TRANSACTION), a završava se:

- operacijom COMMIT (ako je ceo skup operacija uspešno izvršen),
- ROLLBACK (ako ceo skup operacija nije uspešno izvršen),
- DDL naredbom,
- važećom greškom (slično deadlock),
- log off,
- greškom mašine.

Nakom izvršenja jedne transakcije, sledeća SQL naredba automatski startuje drugu po redu transakciju.

Drugim rečima, na nivou SQL naredbi upravljanje transakcijom se upravlja sa:

- naredbom `COMMIT,` kojom se:
  - definišu stalne izmene u bazi podataka,
  - brišu svi `CHECKPOINT` u transakciji,
  - definiše kraj transakcije,
  - realizuje tzv. transakciono zaključavanje i

- omogućuje eksplicitni kraj transakcije;

- naredbom `CHECKPOINT` za duge transakcije kojom se deli transakcija u male 'porcije', tj. čuva rad u transakciji u jednoj tački sa opcijom za kasniji `COMMIT`;
- naredbom `ROLLBACK,` kojom se poništavaju sve izmene u tekućoj transakciji, odnosno kojom se:
  - brišu svi `CHECKPOINT` u transakciji i
  - oslobađaju sva transakciona zaključavanja;
- naredbom `ROLLBACKŠWORKĆ TO CHECKPOINT` kojom se poništava samo deo transakcije.

U radu sa bazom podataka koji se koriste u transakcijama definišu se dve operacije, i to:

- čitanje (read) sa baze podataka, korišćenjem naredbe SELECT i
- ispisivanje (write) u bazu podataka, korišćenjem naredbi INSERT, UPDATE i DELETE.

Jednim programom se može predstaviti sekvenca transakcija, iako je najčešće jedan program jedna transakcija.

Transakcije ne mogu biti "ugnježdene", tj. u jednom programu se operacija BEGIN TRANSACTION može izvršiti samo ako taj program nema nijednu drugu transakciju u izvršenju. Naredbe: COMMIT i ROLLBACK se izvršavaju samo ako je neka transakcija u izvršenju.

Transakcije moraju da prate i odgovarajuće izlazne poruke, koje ne smeju da se prenose direktno, već tek nakon planiranog završetka transakcije (COMMIT ili ROLLBACK).

Posebna pažnja se poklanja sistemu poruka u postupku oporavka baze podataka.

Poruke se ne prihvataju direktno, već se ulazne poruke stavljaju u ulazni red, a izlazne poruke u izlazni red. Pri planiranju završetka transakcije (COMMIT ili eksplicitni ROLLBACK) izvršava se:

- upisivanje log izlazne poruke,
- stvarno se prenose izlazne poruke i
- izbacuju se ulazne poruke iz ulaznog reda.

Pri neplaniranom ROLLBACK (izvršava se automatski ako dođe do neke greške u programu tipa "overflow", deljenja sa nulom i slično) dolazi do izbacivanja izlaznih poruka iz izlaznog reda.

Stoga su u MS ACCESS-u elementi transakcionog rada ugrađeni u okviru ACCESS BASIC, gde su mogući: definisani početak transakcije (BeginTrans), zapisivanje rezultata rada transakcije (CommitTrans) i vraćanje na stanje pre početka transakcije (Rollback).

Sigurnost podataka

Sigurnost podataka odnosi se na mehanizam zaštite podataka od neovlašćenog korišćenja, koji je ugrađen u SUBP. Zaštitom podataka se definiše koji subjekt zaštite (npr., Eremija) nad kojim tabelama (npr., RADNIK) može da izvede neku operacuju (npr., SELECT, UPDATE, INSERT i DELETE) i pod kojim uslovima (npr., samo za odeljenje 10).

Polazi se od činjenice da je korisnik vlasnik kreirane tabele i da samo on kao vlasnik može da je koristi osim ako i drugima ne dozvoli pristup. Za dodeljivanje privilegije korišćenja drugim korisnicima koristi se GRANT naredba koja se sastoji iz tri osnovne klauzule:

```
GRANT <Privilegija>
ON <tabele ili pogled>
TO <korisnik ili grupa korisnika>
ŠWITH GRANT OPTIONĆ;
```

Privilegije mogu biti :

- SELECT,
- UPDATE,
- INSERT,
- DELETE,
- INDEX,
- EXPAND (dodavanje atributa relacije),
- ALL (važi za sve navedene privilegije),
- RESOURCE (omogućuje korisniku kreiranje objekata baze podataka, kao što su: tabele, indeksi, klasteri),
- DBA (obavlja administrativne zadatke, kao što su: CREATE TABLESPACE i CREATE ROLLBACK SEGMENT),
- DBA privilegijom korisnik može definisati:
  - SELECT iz bilo koje tabele i pogleda,
  - CREATE objekata baze podataka za druge korisnike,
  - DROP drugih korisnika objekata baze podataka, uključujući tabele, sinonime i linkovane baze podataka,
  - GRANT varijante privilegija baze podataka,

- CREATE public sinonima i linkovanje baze podataka,
- EXPORT i IMPORT baze podataka.

WITH GRANT OPTION daje dozvolu davanja privilegija drugom korisniku.

Kada se, na primer, uvedu privilegije nad tabelom RADNIK, a onda treba da se nekom drugom da privilegija nad tom tabelom, i to korisniku PERI, piše se ovako:

```
GRANT SELECT ON RADNIK TO PERA;
```

Ako bi trebalo da se da nekom drugom privilegija za samo SELECT nad tabelom RADNIK, i to korisniku VLADA piše se ovako:

```
GRANT SELECT ON RADNIK TO VLADA;
```

Ukoliko bi trebalo da se da nekom drugom privilegija - za samo UPDATE (PLATA, STIMUL) nad tabelom RADNIK, i to korisniku STEFAN piše se ovako:

```
GRANT UPDATE (PLATA, STIMUL) ON RADNIK TO STEFAN;
```

Privilegije se mogu davati i za poglede nad naredbama: SELECT, INSERT, UPDATE i DELETE.

Na primer, ako se ne želi da korisnik ALEMPIJE ima pogled na kolone: PLATA i STIMUL, onda je bolje da se privilegije ograničavaju na pogled, a ne na tabele.

Prvo se definiše pogled na tabelu RADNIK koja ne sadrži kolone: PLATA I STIMUL.

```
CREATE VIEW ZAPS AS
SELECT SIFRAR,PREZIME,SIFRARM,RUKOV,DATUMZ,SIFRAO
FROM RADNIK;
```

Za davanje privilegije korisniku ALEKSANDAR nad pogledom ZAPS piše se:

```
GRANT SELECT ON ZAPS TO ALEKSANDAR;
```

Može se definisati, kao primer, i pogled nad tabelom RADNIK koja daje podatke samo o onim zaposlenima sa istim brojem odeljenja koji ima i osoba koja koristi pogled.

```
CREATE VIEW IMERAD AS
SELECT *
FROM RADNIK
WHERE SIFRAO IN
(SELECT SIFRAO
FROM RADNIK
WHERE PREZIME = USER);
```

USER definiše ime prijavljenog korisnika. Ako CEBIC, koji je u odeljenju 10, pristupio pogledu može videti samo zaposlene u odeljenju 10. Ovim pogledom može se dati korisniku CEBIC privilegija da vrši, npr., ažuriranje na sledeći način:

```
GRANT SELECT, UPDATE
ON IMERAD
TO CEBIC;
```

Ako neki zaposleni pređe u drugo odeljenje (polje SIFRAO se menja), novi rukovodilac automatski dobija pristup podacima dok ga stari rukovodilac gubi.

Kada treba poništiti privilegije koristi se klauzula REVOKE.

Npr., ako se želi oduzeti privilegija korisniku ALEMPIJE za ubacivanje podataka u tabelu RADNIK piše se:

```
REVOKE INSERT
ON RADNIK
FROM ALEMPIJE;
```

Zaključavanje podataka

Zaključavanje podataka je mehanizam za upravljanje konkurentnim pristupom podacima i izvodi se kada korisnik pokuša da izvrši izmenu nad podacima u bazi podataka.

Zaključavanje se vrši prilikom:

- sintaksne analize,
- izvršavanja komandi i
- pristupanja redovima.

Postupak zaključavanja prestaje u dva slučaja:

- kada se transakcija završi (COMMIT/ROLLBACK) i
- kada se kursor zatvori (logoff).

Postupak zaključavanja tabela i redova je najvažniji deo održavanja konzistentnosti i integriteta baze podataka korisnika.

Osnovna podela zaključavanja je nad:

- tabelama rečnika podataka - DDL (Data Definition Language),
- tabelama korisnika podataka - DML.

DDL način zaključavanja kontroliše pristup bazi podataka i izvodi se automatski nad tabelama rečnika podataka. Ovaj način zaključavanja upravlja sledećim SQL naredbama:

`CREATE TABLE...,ALTER TABLE..., DROP TABLE...i dr.`

DML način zaključavanja upravlja pristupom podacima u korisničkim tabelama.

Zaključavanje naročito dolazi do izražaja kada je u pitanju uporedna obrada transakcija. Transakcije se izvode uporedo sa drugim transakcijama u sistemu ako više transakcija zahteva isti slog baze podataka.

Mogu se definisati dva načina zaključavanja, i to:

- zajedničko (shared),
- isključivo (exclusive)).

Shared(s) zaključavanje je istovremeno zajedničko zaključavanje tabela da bi se obezbedio upit nad konzistentnim podacima cele tabele (bez transakcione obrade) ili reda (sa transakcionom obradom). Ovakvo zaključavanje može izvesti više korisnika. Ovaj način zaključavanja onemogućuje druge korisnike da vrše promene nad podacima i stavljaju exclusive zaključavanje, ali ih ne ograničava pri tom da vrše upite.

Exclusive zaključavanje je isključivo zaključavanje tabela ili redova da bi se omogućilo ekskluzivno unošenje promena podataka tako što se onemogućavaju drugi da istovremeno unose promene, tj. onemogućavaju drugi da istovremeno stave bilo kakvo zaključavanje nad istom tabelom.

Oporavak baze podataka

Jedan od veoma važnih elemenata, vezanih za server baze podataka, jeste mogućnost *oporavka baze podataka (recovery),* što predstavlja povratak baze podataka u stanje pre softverskog ili hardverskog otkaza sistema. Razlozi za otkaz sistema mogu biti usled greške u operativnom sistemu, greške u programiranju, greške u samom SUBP-u ili, pak, padanja glave diska, nestanka struje i dr.

Tehnika redundantnog pamćenja podataka i tehnika oporavka baze su veoma kompleksne. Poznato je da su do sada korišćene jednostavne procedure koje su se bazirale na periodičnom kopiranju baze podataka u neku arhivsku memoriju i svih transakcija koje su se u međuvremenu dogodile, kao i na jednostavnom dupliciranju baze podataka. Iako jednostavne, ove metode imaju čitav niz nedostataka.

Proces oporavka izvodi se u globalu u tri koraka:

- periodično kopiranje (dump) baze podataka na eksternu memoriju;
- zapisivanje promena baze podataka u žurnal (tzv.log), i to:
  - stara vrednost u before image i

    - nova vrednost u after image;
- ako je došlo do otkaza sistema zato što je:
    - sama baza podataka oštećena, kada se izvodi oporavljanje baze podataka na osnovu arhivske kopije, ili
- baza dovedena (nije oštećena) u neko nekonzistentno stanje i pomoću log-a se poništavaju sve nekorektne promene, a same transakcije koje su ih proizvele se ponove.

Da bi se omogućio oporavak baze podataka potrebno je, pre svega, razmotriti koje su to vrste otkaza:

- planirani ROLLBACK, tj. narušavanje integriteta u nekoj od transakcija koje program sam otkriva;
- otkazi u transakciji (lokalna greška, npr., deljenje nulom);
- sistemski otkaz koji utiče na sve transakcije, ali ne oštećuje bazu podataka (npr., nestanak struje);
- fizičko oštećenje baze podataka.

Otkazi u transakciji

Ako se transakcija završi pre naredbe COMMIT ili eksplicitni ROLLBACK, neophodno je da se izvrši automatski ROLLBACK. Proceduru ROLLBACK obavlja Recavery Manager, poništavajući sve promene unazad kroz log, dok se u log-u ne dostigne slog koji označava početak transakcije.

U toku poništavanja može doći do otkaza i u tom slučaju ROLLBACK procedura mora startovati ponovo, pa je i to jedan od uslova da transakcija bude što kraća.

Sistemski otkaz bez oštećenja baze podataka

Sistemski otkaz prouzrokuje prestanak rada celog sistema, pa se mora izvesti oporavak baze podataka pretraživanjem celog log-a i identifikovanjem transakcija koje su otpočete, a nisu završene.

Da bi se skratilo pretraživanje, uvode se tzv. tačke ispitivanja (checkpoint) na sledeći način:

- upisuje se sadržaj bafera za log na log;
- upisuje se "checkpoint record" na log;
- upisuje se sadržaj bafera baze podataka u bazu podataka;
- upisuje se adresa "checkpoint" rekord-a na "restart file".

Sadržaj bafera se prenosi tek kad se oni napune. "Checkpoint record" sadrži listu svih transakcija koje su aktivne u trenutku uzimanja "checkpoint-a" i za svaku transakciju adresu sloga u log-u kojem je poslednjem pristupio.

Algoritam oporavka izvodi se tako što se prvo formiraju liste: "undo", u kojoj su sve transakcije zapisane u "checkpoint "-u, i "redo", koja je u početku prazna.

Pretraživanje po log-u počinje od "checkpoint record "-a na sledeći način:

- svaka transakcija za koju se pronađe BEGIN TRANSACTION stavlja se u "undo" listu;
- svaka transakcija za koju se izvodi COMMIT smešta se u "redo" listu.

Ovakav način memorisanja omogućuje Recavery Manager-u da unazad izvršava "undo" transakciju, a potom unapred "redo" odgovarajuću transakciju.

Kako u intervalu upisivanja u samu bazu (Log Write-Ahead) i na log može doći do otkaza, a da bi se obezbedio oporavak baze podataka, prvo se vrši upisivanje na log.

Otkaz diska

Ako bi došlo do otkaza diska, baza podataka bi se oporavljala na osnovu arhivske kopije i log-a, obnavljajući sve transakcije od trenutka uzimanja kopije do otkaza. To je veoma dugotrajan proces i izvodi se, obično, kada nijedna transakcija nije aktivna.

Može se definisati i tzv. inkrementni dump, odnosno na kopiju se prenose samo oni slogovi koji su pretrpeli promene poslednjeg uzimanja kopije.

Logičko modeliranje podataka

Logičko modeliranje podataka ili metodologija odozdo na gore ili je ključni momenat gde do izražaja dolaze sposobnost i znanje visokostručnog kadra iz oblasti menadžmenta i informatike.

Ova aktivnost otvara "crnu kutiju", koja je budućim korisnicima uvek bila nepoznata, jer nisu mogli da prate razmišljanja projektanata informacionog sistema. Prvi put korisnici uzimaju aktivno učešće i u ovom delu i prvi put projektanti informacionog sistema crtaju ono što predstavlja njihovo iskustvo i saznanje o poslovanju konkretnog preduzeća i što su oni osmislili u svojoj glavi. Kroz identifikaciju entiteta, odnosno kroz definisanje objekata od interesa za posmatranje i definisanje veza definiše se ER model, postupkom *odozgo nadole,* tj. intervjuom sa budućim korisnicima.

U okviru ove aktivnosti koristi se tehniku za opisivanje strukture podataka i poslovnih pravila pod nazivom *model podataka***,** kojim se definišu entiteti *(entities).* Pri tom svaki entitet ima svoje osobine, tj. atribute *(attributes),* a sve je to povezano vezama *(relationships)*. Preciznije, može se razmišljati o nekom entitetu kao o setu ili kolekciji (skupu) individualnih objekata zvanih *primerci*

ili instance (instances). Jedan primerak je jedan pojavni oblik datog entiteta. Svaki primerak mora imati identitet različit od svih ostalih primeraka.

Postupak izvođenja ove aktivnosti sadrži sledeće korake:

- definisanje nezavisnih entiteta, tj. pronalaženje 'roditelja';
- definisanje zavisnih entiteta;
- definisanje veza.

Nezavisni entitet je objekat koji ima jednu osobinu koja ga može jednoznačno identifikovati, tj. nezavisni entiteti imaju vlastitu identifikaciju (ne zavise od drugih entiteta). Grafički se nezavisni entiteti prikazuju pravougaonikom u koji se upisuje naziv tipa entiteta u jednini.

Zavisni entiteti su entiteti čije egzistencija i identifikacija zavise od drugog ili drugih entiteta.

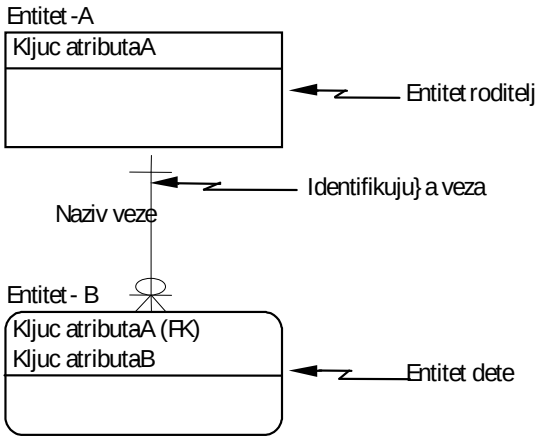
Definisanje veza - Veza je asocijacija između dva ili više entiteta, tj. predstavlja odnos koji postoji među objektima, bilo u realnosti ili u mislima. Entitet od koga je uspostavljena veza zove se "roditelj" (parent), a entitet ka kome je uspostavljena veza zove se "dete" (child).

Veza "roditelj"-"dete" je asocijacija između entiteta gde su svi primerci entiteta "roditelj"(Entitet-A) asocirani sa nula, jedan ili više primeraka entiteta "dete"(Entitet-B), a svi primerci entiteta "dete"(Entitet-B), su asocirani sa nula ili jedan – primerkom entiteta "roditelj"(Entitet-A).

Drugim rečima, način povezivanja dva entiteta ima osobine koje se nazivaju *kardinalnost, koja pokazuje "koliko nečega" od dva entiteta može biti uključeno (sadržano)*.

U daljem tekstu detaljno će biti razmatrani sledeći tipovi identifikujućih veze, koje entitet "dete" identifikuju kroz njegovu vezu sa entitetom "roditelj" i neidentifikujuće veze, koja ne identifikuje "dete" preko identifikatora "roditelj".

Identifikujuće veze - Veza se zove identifikujuća zato što ključevi entiteta "roditelj" predstavljaju deo identiteta entiteta "dete", tj. entitet "dete" zavisi od entiteta "roditelj" preko identifikatora. Dakle, ako se primerak entiteta "dete" identifikuje preko asocijacije sa entitetom "roditelj", onda se veza definiše kao identifikujuća veza, i svaki primerak entiteta "dete" mora biti povezan sa najmanje jednim primerkom entiteta "roditelj".



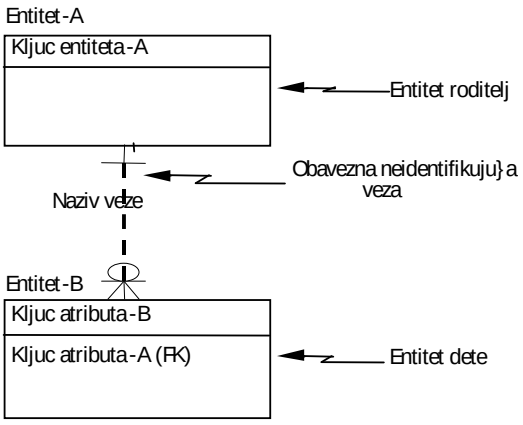
Slika 6. Identifikujuća veza

Identifikujuća veza je prikazana punom linijom i povezuje entitet "roditelj" sa entitetom "dete", sa tačkom na strani entiteta "dete".

U identifikujućoj vezi entitet "roditelj" ima svoj nezavisni primarni ključ (Ključ entiteta A), a entitet "dete" ima složeni ključ koji se sastoji od svog ključa (Ključ entiteta B) i prenesenog roditeljskog ključa (Ključ entiteta A(FK)). Dakle, instance entiteta "roditelj" se definišu nezavisno, a instance entiteta "dete" se ne mogu identifikovati bez identifikatora entiteta "roditelj".

Ako se svaki primerak entiteta "dete" može jedinstveno identifikovati, bez znanja veze sa primerkom entiteta "roditelj", onda se takva veza definiše kao *neidentifikujuća veza*.

Neidentifikujuće veze su prikazane isprekidanom linijom koja povezuje entitet "roditelj" i entitet "dete" sa tačkom na strani entiteta "dete".



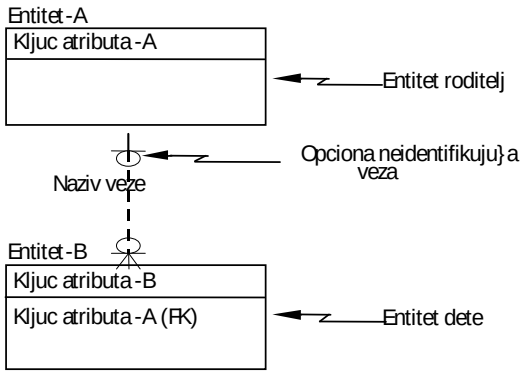
Slika 7. Neidentifikujuća obavezna veza

Neidentifikujuća ili slaba veza zavisi od načina definisanja ključeva od "roditelja" ka "detetu" na dva načina:

- kao obavezna neidentifikujuća veza i
- kao neobavezna (opciona) neidentifikujuća veza.

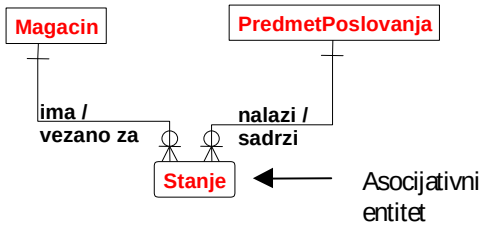
Ako je veza (relationship) obavezna (No Nulls ili Mandatory) iz perspektive "roditelj", onda je "dete" egzistencijalno zavisno od "roditelja". No nulls ili Mandatory znači da je obavezan unos prenesenog ključa entiteta "roditelj" u okviru entiteta "dete" (Ključ entiteta A (FK)).

Ako je veza neobavezna (Nulls Allowed ili Optional), tada "dete" niti je egzistencijalno niti identifikaciono zavisno, ali poštuje tu vezu. Null Allowed ili Optional znači da nije obavezan (može biti Null) unos prenesenog ključa entiteta "roditelj" u okviru entiteta "dete" (Ključ entiteta A (FK)).



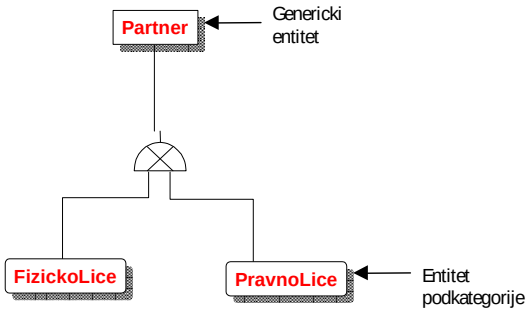
Slika 8. Neidentifikujuća neobavezna veza

Asocijativni entiteti su sastavljeni od više veza između dva ili više entiteta, kao što se može videti na sledećoj slici. Npr., ako Magacin ima više PredmetaPoslovanja i jedan PremetPoslovanja se nalazi u više Magacina, tada je Stanje asocijativni entitet koji opisuje vezu između entiteta: Magacin i PredmetPoslovanja. Dakle, asocijativni entiteti nose informaciju o višeznačnoj vezi.



Slika 9. Veza asocijativnog entiteta Stanje sa nezavisnim entitetima Magacin i PredmetPoslovanja

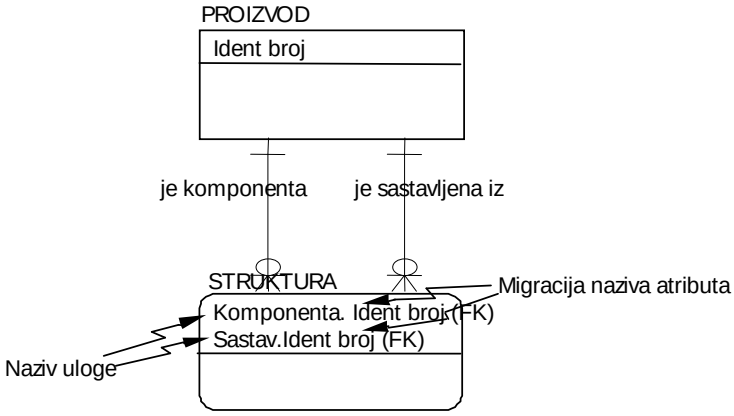
Entitet kategorija (category) zavisan je entitet, koji ima tzv. vezu tipa potkategorije (sub-category). Kod entiteta tipa kategorije definišu se: nadređeni entitet, koji ima zajedničke osobine (npr. entitet Parner) i podređeni entiteti (entiteti: FizickoLice i PravnoLice), koji se identifikuju ključem nadređenog i poseduju svoje specifične osobine (slika 30).



Slika 10. Primer potkategorije entiteta

Preneseni ključevi (skraćenica FK) mogu imati i drugo ime, što se definiše kao uloga atributa u entitetu. Ime uloge (Rolename) predstavlja novo ime za prenesene ključne atribute koji definišu ulogu koju ti atributi igraju u tom entitetu. Ime uloge deklariše novi atribut, čije ime treba da opiše poslovno pravilo ugrađeno preko veze koja zahteva preneseni ključ.

Definisanje imena uloge biće pokazano na primeru definisanja strukture proizvoda.



Slika 11. Primer za definisanje imena uloge

Kako entitet, STRUKTURA ima složeni ključ od istog nadređenog entiteta PROIZVOD, pa se definisanjem uloge definišu različita imena atributa i identifikuje entitet STRUKTURA.

Modeliranje podataka za Fakturu

Poći će se od EDIFACT standarda koji je definisan dokumentom UN/ECE WP.4 koji svojom preporukom broj 6, izdanje iz 1975. godine, preporučuje da se obrazac za fakturu u

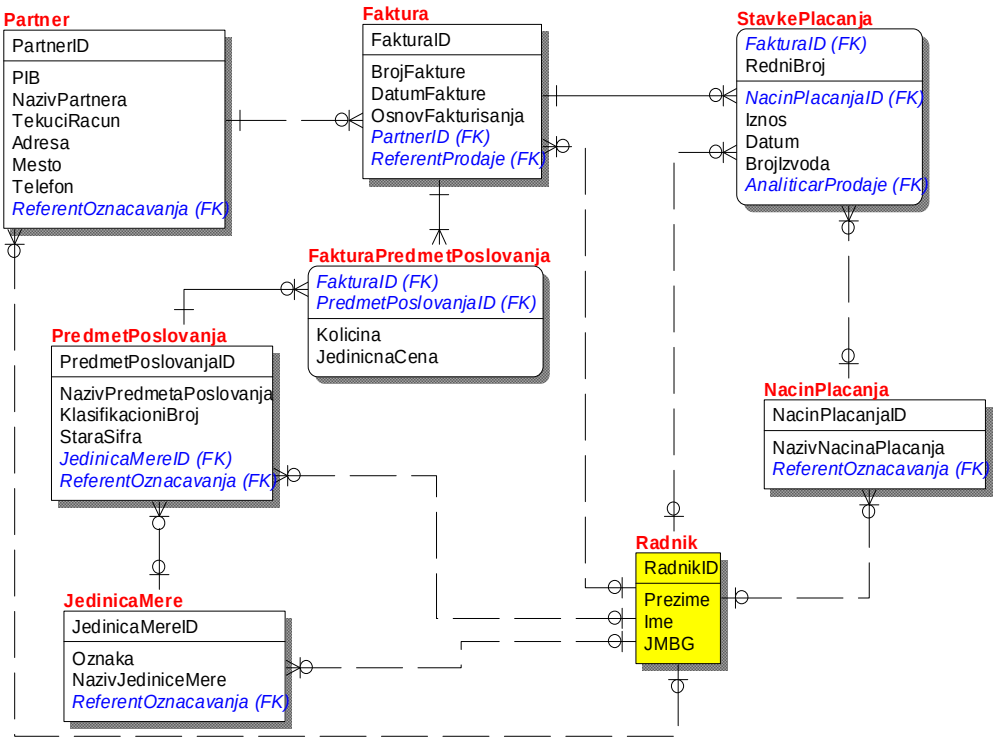
međunarodnoj trgovini zasniva na obrascu prema ISO 6422 (skraćeno: UNLK) – JUS ISO 6422.

Obrazac je baziran na principu "box design". Razmeštaj rubrika je fiksiran kao npr. adresa primaoca, koja je definisana zbog koverata sa prozirom i pritom su razmatrani administrativni, pravni i trgovinski aspekti. Prostor "za slobodno raspolaganje" u najnižem delu obrasca namenjen je za više posebnih potreba za individualne aplikacije. Ako trgovački partneri primenjuju dokumente koji sadrže šire polje podataka nego set UNLK, ili od onog što je propisano u standardima za odgovarajuće podatke, o tome se moraju posebno dogovoriti. Sve ovo je išlo u prilog da se prihvati jedan standardizovani obrazac (ISO 7372), kao što je prikazano na sledećoj slici.

ISPORUČILAC *PERIHARD INŽINJERING* *I. MILUTINOVICA 25* *11000 BEOGRAD*		DATUM I BR. FAKTURE *960321 547XRTW* **DRUGE REF**. *FAKTURA*		
PRIMALAC *SITJ KNEZA MILOŠA 9* *11000 BEOGRAD*		**KUPAC (AKO NIJE PRIMALAC)**		
		ZEMLJA POREKLA ***USA***		
DETALJI O PREVOZU *KAMION, AVION*		USLOVI ISPRUKE I PLAĆANJA		
OTPREMNE OZNAKE	**BROJ I VRSTA; OPIS ROBE; PAKOVANJE**		**MASA kg**	**ZAPREMINA m^3**
OPIS ARTIKLA (ŠIFRA I/ILI NAZIV)		**KOLIČINA**	**J. CENA**	**IZNOS**
LASERSKI STAMPAČ *TONER* *RAČUNAR*		*3* *2* *1*	*12000* *3000* *8500*	*36000* *6000* *8500*
	AMBALAŽA			
	PREVOZ			
	OSTALI TROŠKOVI			
	OSIGURANJE			
	UKUPNO			*50500*

Slika 12. Obrazac EDIFACT fakture

Na sledećoj slici prikazan je logički model podataka za EDIFACT fakturu.



Slika 13. Logički model podataka definisan IE metodologijom

Nezavisni entiteti, koji imaju vlastitu identifikaciju, su šifarnici (Partner, JedinicaMere, NacinPlacanja, PredmetPoslovanja, Radnik) i Faktura, i oni nisu zavisni od drugog entiteta.

Zavisni entiteti, to su entiteti čija je egzistencija i identifikacija zavisna od entiteta Faktura, i to su: StavkaFakture i FakturaPredmetPoslovanja.

Informacionim modeliranjem, završava se deo gde ne zavisimo od izbranog sistema za upravaljanje bazama podataka. Ovo se posebno naglašava jer do ovog trenutka ne treba kupovati računare već treba samo projektovati.

Fizičko modeliranje podataka

Fizičko modeliranje treba da omogući projektantima baze podataka da fizički kreiraju efikasnu bazu podataka i da pomogne projektantskom timu u razvoju aplikacije i odabiru načina pristupa podacima.

Da bi se jedna aplikacija okvalifikovala kao potpun sistem za upravljanje relacionom bazom

podataka, ona mora da izvršava sledeće:

- Organizacija podataka – obuhvata izradu i rukovanje tabelama koje sadrže podatke u konvencionalnom tabelarnom formatu koju Access naziva pogled (Datasheet).
- Povezivanje tabela i izdvajanje podataka – povezuje više tabela prema relacijama između podataka radi izrade privremenih tabela, koje sadrže izabrane podatke. Access koristi upite da bi povezao tabele i izabrao podatke koji će se čuvati u privremenoj tabeli, koja se naziva objekat Recordset. Objekti Recordset nazivaju se virtuelne tabele, jer se čuvaju u memoriji računara umesto u datotekama baze podataka.
- Unos i uređivanje podataka – zahteva projektovanje i implementaciju obrazaca za pregled, unos i uređivanje podataka kao alternativu tabelarnom prikazu. Obrasci su ti koji umesto aplikacije omogućavaju da kontrolišete prikazivanje podataka.
- Prikazivanje podataka – zahteva izradu izveštaja koji mogu da sumiraju podatke u skupovima zapisa (Recordset). Njih možete da pregledate, štampate ili objavljujete na internetu ili intranetu.

Fizičko modelaranje vezano je za:

- definisanje fizičkog dizajna
- generisanje šeme baze podataka

Definisanje fizičkog dizajna

"Definisanje fizičkog dizajna" prevodi logički u fizički model, tj. entitet u tabele, atribute u kolone, kao i odgovarajuća ograničenja. ERwin nudi mogućnost istovremenog definisanja logičkog i fizičkog nivoa i jednostavno prebacivanje sa logičkog na fizički pogled na model.

Prilikom prevođenja logičkog modela u fizički model, dolazi do sledećih konvertovanja:

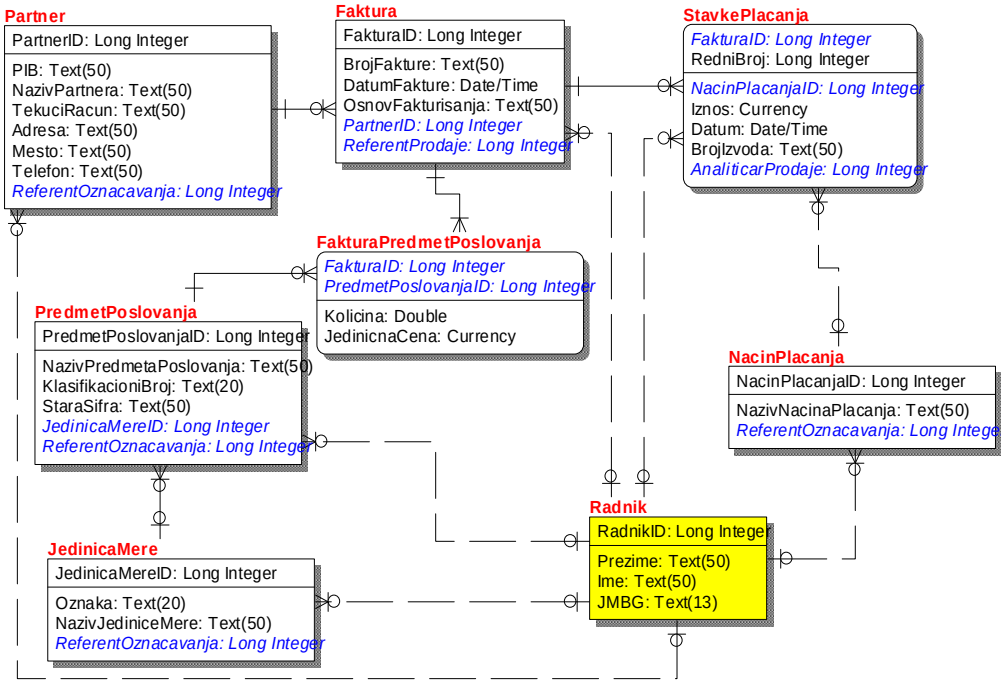
- entiteta iz modela podataka u tabele fizičke baze podataka;
- atributa u kolone u odgovarajućim tabelama;
- kandidati za ključeve entiteta postaju primarni ključevi u tabelama;
- veze između entiteta postižu da primarni ključevi u tabelama postaju spoljni ključevi u povezanim tabelama.

Definisanje fizičkog modela podataka, tj. implementacija entiteta i njihovih atributa u tabele i kolone nekog SUBP, korišćenjem ERWin-a, relativno je jednostavan posao. Programski modul ERWin-a za izgradnju fizičkog modela čita opis entiteta i atributa i formira tabele i polja fizičkog modela.

Dakle, ERwin definiše tabele i kolone automatski, tj. nazivi tabela po defaultu dobijaju imena na

osnovu naziva entiteta, a nazivi atributa po defaultu postaju nazivi kolona. I druge osobine se dodeljuju kao default setovane vrednosti (vrednosti koja će biti insertovana u kolonu)

Na sledećoj slici prikazan je fizički model podataka definisan IE metodologijom u ERwin-u za izabrani SUBP MS Access.



Slika 14. Fizički model podataka definisan IE metodologijom u ERwin-u

Generisanje šeme baze podataka

"Generisanje šeme baze podataka" definiše se za izabranu ciljnu platformu(MS Access), gde se definišu fizičke tabele, kolone i relacije.

Na primeru procesa FAKTURISANJE pokazaće se postupak generisanja šeme baze podataka.

Šema logičke baze podataka obuhvata poseban skup podataka (odgovarajući rečnik podataka) sa odgovarajućom semantikom i vezama među elementima baze podataka. Fizički, ove veze su smeštene u bazi podataka, za kasniju upotrebu.

Na osnovu fizičkog modela podataka, izvodi se generisanje šeme baze podataka koju čine fizičke tabele, kolone i relacije, koje se u CASE alatu automatski generišu iz fizičkog modela.

Proces generisanja šeme baze podataka iz fizičkog modela podataka naziva se direktni inženjering. Kada se generiše šema baze podataka, entiteti prelaze u tabele, atributi u kolone, a veze u relacije i definišu se referencijalni integriteti, trigeri, procedure, indeksi i druge osobine koje podržava izabrani SUBP, o čemu će više reči biti u sledećem poglavlju.

Dakle, da bi se generisala baza podataka potrebno je, prvo, preci na fizicki model, potom, izabrati odgovarajuću ciljnu platformu (izabran je MS Access) i potom se logovati na nju. Kada se korisnik loguje na izabranu platformu, ERWin kreira aktivnu bidirekcionu vezu sa sistemskim katalogom izabranog servera koja omogućava direktno kreiranje baze podataka.

Implementacija entiteta i njihovih atributa u tabele i kolone nekog SUBP, korišćenjem ERwin-a, relativno je jednostavan posao. Programski modul ERwin-a za izgradnju fizičkog modela čita opis entiteta i atributa i formira tabele i polja fizičkog modela.

Proces implementacije veza i njihovih definicija je mnogo složeniji posao. Radi ilustracije, sledi razmatranje prevođenja ERwin modela u MS Access dekstop SUBP.

ERwin definiše tabele i kolone automatski, tj. nazivi tabela po defaultu dobijaju imena na osnovu naziva entiteta, a nazivi atributa po defaultu postaju nazivi kolona. I druge osobine se dodeljuju kao default setovane vrednosti (vrednosti koja će biti insertovana u kolonu). U Target Server editoru (Erwin) postavljaju se default vrednosti za:

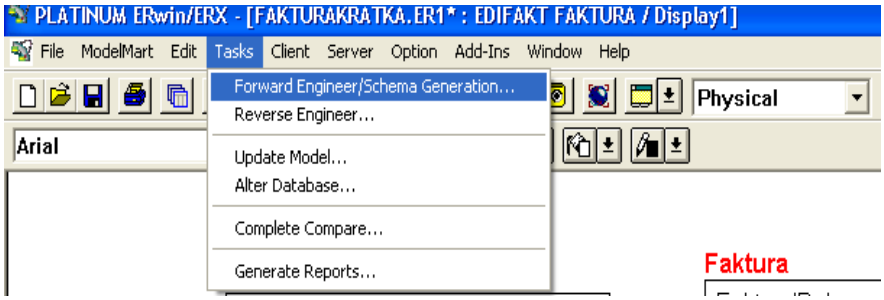
- Default Access Datatype kojim se definišu default tip i veličina kolone, npr., text (18);
- Reset Physical Name kojim se izvodi resetovanje fizičkih imena;
- Referential Integrity Default kojim se definišu default vrednosti referencijalnog integriteta.

Na osnovu definisanih default osobina prelazi se na realizaciju fizičkog modela tj. definisanje osobina kolona.

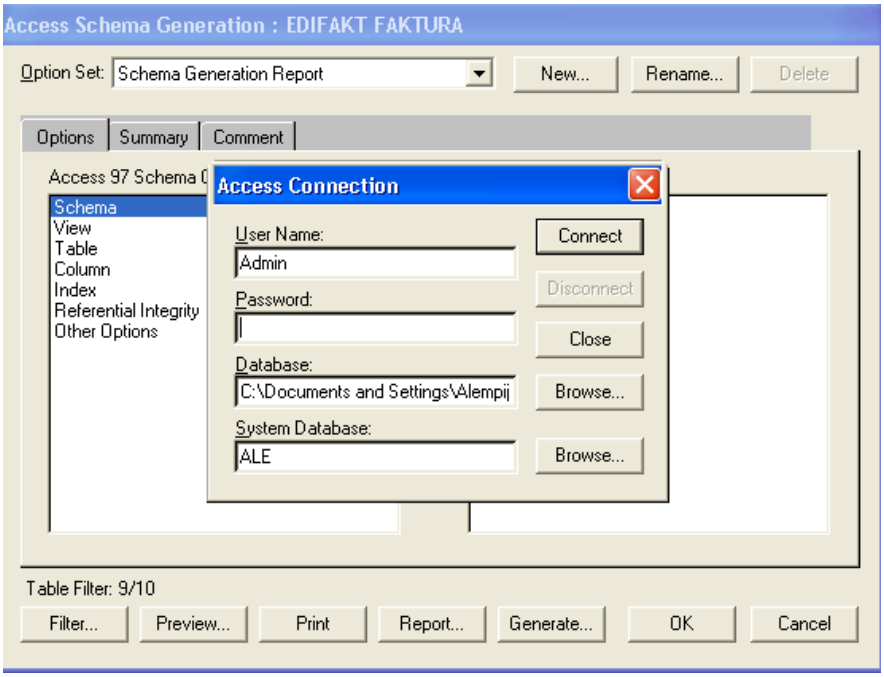
Osobine kolona se definišu korišćenjem editora za definisanje kolona (Column property Editor) gde se mogu praviti izmene nad default vrednostima kolona.

U okviru ovog editora prikazuju se ime selektovanog entiteta i ime odgovarajuće tabele, a u sledećem nivou spisak kolona sa definisanim osobinama. U donjem delu za izabranu kolonu definišu se osobine kolone, i to: tip podatka, domen, validacija i default vrednost.

Da bi se generisala sema baze podataka kao sto je prikazano na sledećoj slici potrebno je prvo definisati praznu bazu podataka (ako je MS Access 97 ekstenzija .mdb). Generisanjem baze podataka se izvodi tako sto se izabere opcija Task/ Forward Engineer. Unes se UserName Admin i u okviru Database pomocu Browse pozicioniramo se na predhodno definisana prazna baza podataka.

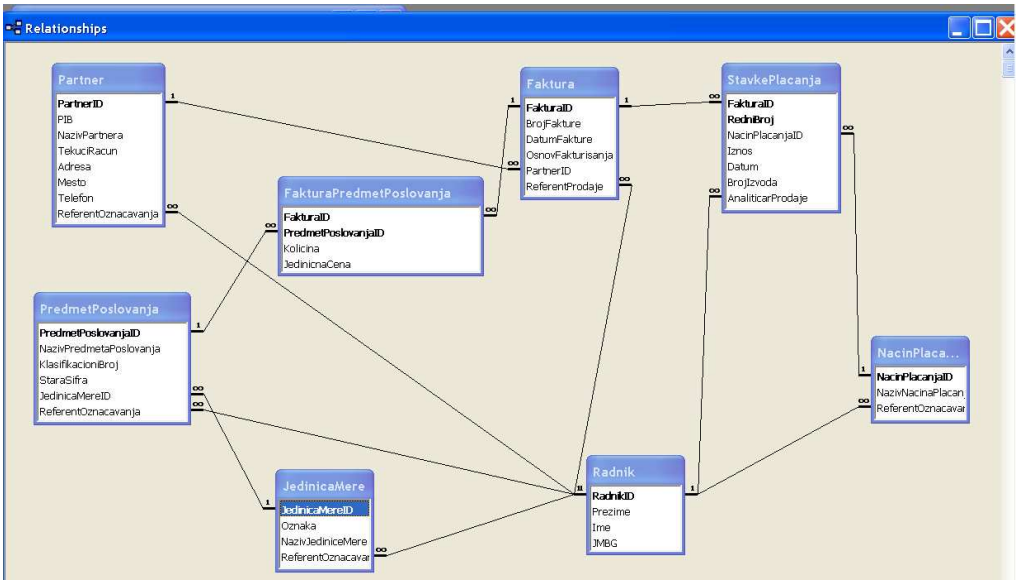


Slika 15 Izbor za generisanje seme baze podataka



Slika 16 Generisanje šeme baze podataka

Kada se izgeneriše baza podataka na sledećoj slici prikazana je šema baze podataka generisana iz Erwin-a u MS Access.



Slika 17. Šema MS Access baze podataka

Na osnovu generisanih tabela koje predstavljaju server stranu prelazi se na definisanje klijent strane tj. izradjuje se aplikacija.

Izrada aplikacije- klijent strana

Da bi se jedna aplikacija okvalifikovala kao potpun sistem za upravljanje relacionom bazom podataka, ona mora da izvršava sledeće:

- Organizacija podataka – obuhvata izradu i rukovanje tabelama koje sadrže podatke u konvencionalnom tabelarnom formatu koju Access naziva pogled (Datasheet).
- Povezivanje tabela i izdvajanje podataka – povezuje više tabela prema relacijama između podataka radi izrade privremenih tabela, koje sadrže izabrane podatke. Access koristi upite da bi povezao tabele i izabrao podatke koji će se čuvati u privremenoj tabeli, koja se naziva objekat Recordset. Objekti Recordset nazivaju se virtuelne tabele, jer se čuvaju u memoriji računara umesto u datotekama baze podataka.
- Unos i uređivanje podataka – zahteva projektovanje i implementaciju obrazaca za pregled, unos i uređivanje podataka kao alternativu tabelarnom prikazu. Obrasci su ti koji umesto

aplikacije omogućavaju da kontrolišete prikazivanje podataka.

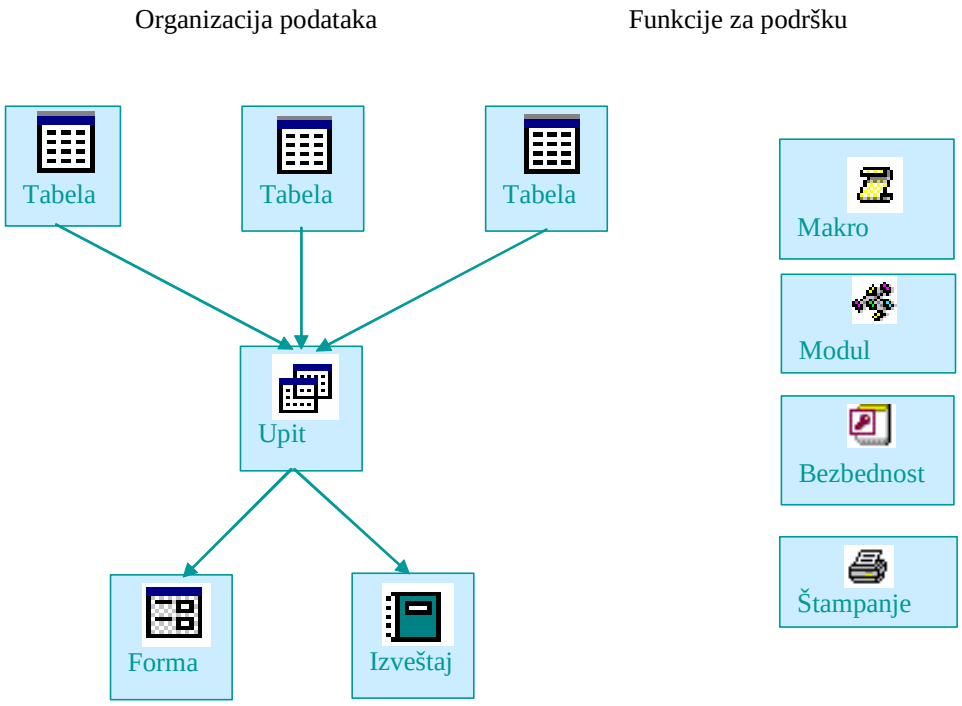
- Prikazivanje podataka – zahteva izradu izveštaja koji mogu da sumiraju podatke u skupovima zapisa (Recordset). Njih možete da pregledate, štampate ili objavljujete na internetu ili intranetu.

"Izradom aplikacija" treba da se realizuje korisnički pogled na podatke, tj. da se definišu meniji, forme, upiti i izveštaji.

Izrada aplikacije neposredno je vezana i za klijent/server arhitekturu. U klijent/server arhitekturi definiše se server u kojoj se nalazi baze podataka i SUBP i klijent-strana gde su definisane klijentove (korisničke) aplikacije, pa se izrada aplikacije izvodi na klijent-strani.

Aplikacija klijent: radi sa redovima iz tabele; poseduje interfejs prema korisniku, tzv. GUI (Graphical User Interface); izvršava logiku aplikacije; proverava ispravnost ulaznih podataka; traži prijem podataka od servera.

Na sledećoj slici prikazane su osnovne funkcije za podršku Accessa.



Slika 18. Osnovne funkcije za podršku Accessa.

U Access-u organizacija podataka definiše:

- *Tabele* koje su ili generisane iz ERwin-a kao što je u predhodnom tekstu pokazano ili koristići osnovne funkcije Accessa za generisanje tabela.
- *Forme* koje omogućuju korisniku predstavljanje podataka iz baze i unos podataka u bazu. Forme u sebi mogu imati veliki broj drugih objekata (kontrola). Većina SUBP, koji za osnovu imaju MS WINDOWS, podržava tzv. wizard metodologiju za kreiranje formi.
- *Upiti* koji predstavljaju set zajedničkih komandi i funkcija definisanih ISO standardom za SQL.
- *Izveštaji* koji su papirnati izglede postavljenih upita.
- *Meniji* koji treba da prate scenario odvijanja aktivnosti budućeg korisnika. Za definisanje menija moraju se koristiti odgovarajuća pravila za strukturiranje kojima se definiše mogući redosled pozivanja operacija.

Funkcije za podršku su sledeće:

- Makroi su sekvence aktivnosti, koje automatizuju operacije nad bazom podataka koje se ponavljaju. Pri radu sa bazama podataka Access 2000, za automatizaciju se koristi Visual Basic.
- Moduli su funkcije i procedure koje su napisane u programskom jeziku VB. Funkcije VB se koriste da bi se izvršavala složenija izračunavanja od onih koja se mogu lako izložiti pomoću niza konvencionalnih matematičkih simbola, ili za izračunavanja koja zahtevaju donošenje odluka. VB potprogrami napisani su za izvršavanje operacije koje prevazilaze mogućnosti standardnih aktivnosti makroa što je jedan od razloga da se u Accessu napušta podrška makroima. VB podprogrami se izvršavaju tako što se pridružuju odgovarajućim događajima, kao što je pritisak na dugme pomoću miša, koji se dešava kada je aktivni objekat neki obrazac ili izveštaj.
- Bezbednost sačinjavaju funkcije koje su dostupne kao stavke menija i preko VBA potprograma. Pomoću funkcija bezbednosti podataka može se dopustiti drugim osobama da koriste vašu bazu podataka, u višekorisničkom okruženju. Pristup možete dodeliti grupi korisnika ili pojedincima, ali i ograničite njihove mogućnosti za pregled ili modifikacije svih ili samo nekih tabela u bazi podataka.
- Štampanje dopušta da odštampate praktično sve što možete da pregledate u radnom režimu programa Accessa.

Access ima tri osnovna radna režima:

- Režim za pokretanje (Startup mode) omogućava da konvertujete, šifrujete, dešifrujete i popravite podatke iz baze, izborom komandi iz podmenija Database Utilities i Security, menija Tools, pre otvaranja baze podataka. Ove komande su dostupne samo ako baza

podataka nije otvorena.

- Režim projektovanja (Design mode) omogućava da napravite i modifikujete strukturu tabela i upita, razvijate obrasce za prikaz i uređivanje podataka, kao i da formatirate izveštaje za štampanje.
- Režim izvršavanja (Run mode) prikazuje dizajn tabela, obrasca i izveštaja u posebnim prozorima za dokument. Makroe izvršavate tako što jedan od njih izaberete, a zatim izaberete režim izvršavanja. Ovaj režim se ne primenjuje na VB module, jer se funkcije izvršavaju kada se pojave kao elementi upita, obrazaca ili izveštaja. Režim izvršenja za tabele i upite naziva se pogled Datasheet, za obrasce pogled Form, za strane za pristup podacima (DAP), pogled Page, a za izveštaje pogled Print Preview.

Termin aplikacija u Accessu označava bazu podataka sa sledećim karakteristikama:

- Sadrži upite, obrasce, izveštaje i makroe, koji su neophodni za prikaz podataka na razumljiv način i za ažuriranje tih podataka ako je neophodno. (Objekti aplikacije).
- Od korisnika baze podataka ne zahteva se da znaju kako se projektuje bilo koji od elemenata baze. Svi elementi baze podataka prethodno su u potpunosi definisani tokom projektovanja aplikacije. U većini slučajeva, želite da onemogućite da drugi korisnici namerno ili slučajno promene dizajn aplikacije.
- Aplikacija je automatizovana pomoću VB koda, tako da korisnici vrše izbor pomoću komandnih dugmadi ili opcija iz namenski projektovanih menija, umesto iz lista u prozoru Database.

Access sadrži glavnu datoteku baze podataka koja se naziva datoteka radne grupe, pod nazivom System.mdw. Ova datoteka sadrži sledeće informacije:

- Imena korisnika i grupa korisnika, koji mogu da otvore Access.
- Lozinke i jedinstveni binarni kod korisnika, koji se naziva System ID (SID) i koji identifikuje korisnika koji trenutno koristi Access.
- Operativna podešavanja, koje uspostavljate izborom stavki Tools, Options iz menija.
- Definicije prilagođenih paleta alatki u Accessu 2000, koje pravi svaki korisnik.

Još jedna kategorija datoteka, kod baza podataka u Accessu, pojavljuju se dopunski programi, koji se nazivaju i biblioteke.

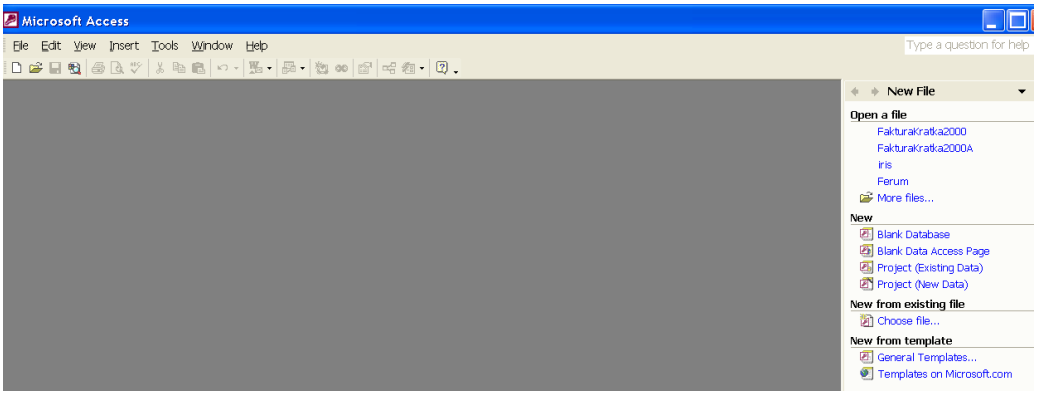
Dopunski programi predstavljaju bibliotečku bazu podataka Accessa, obično sa oznakama tipa .mde ili .mda, da bi se razlikovali od korisničkih baza podataka, a sa Accessom možete da ih povežete izborom alatke Add-In Manager (kojoj možete da pristupite izvorom opcije Tools, Add-Ins).

Kada povežete neku biblioteku Accessa, svi elementi te bibliotečke baze podataka biće vam

dostupni kada otvorite Access.

Start Me Up

Access se može pokrenuti na više načina: dvoklikom na shortcut ikonu na desktop-u, izborom opcije Access menija Start->Programs ili klikom na dugme sa prepoznatljivom ikonom Access-a na Microsot Office Shortcut Bar-u. Po pokretanju Access-a na samom početku pojaviće se uvodni prozor (slika 1) .



Slika 19 Prikaz Start Me Up

Opcija za kreiranje nove baze nudi dve mogućnosti kreiranja baze: otvaranje "blanko" baze ili kreiranje "kostur" aplikacije korišćenjem "čarobnjaka".

Metodologija rada sa "čarobnjacima" je , uglavnom, ujednačena i odvija se preko niza dijalog prozora u kojima se donosi odluka o uključenju ponuđenih vrednosti u objekat ili deo aplikacije. Sledeće opcije se javljaju u većini "čarobnjaka":

- Next - opcija koja se koristi po završetku izbora na aktuelnom dijalog prozoru. Ova opcija automatski otvara naredni dijalog prozor.
- Back - opcija koja vas vraća u predhodni dijalog ukoliko želite da ispravite neki podatak definisan na predhodnom dijalog prozoru.
- Finish - opcija kojom se završava rad sa "čarobnjakom" i on na osnovu izabranih podataka kreira željeni objekat. Ovu opciju moguće je koristi pre poslednjeg dijalog prozora, ukoliko ne želite da menjate ni jedan od ponuđenih podataka na narednim dijalog prozorima "čarobnjaka".
- Cancel - opcija pomoću koje se prekida rad sa "čarobnjakom".

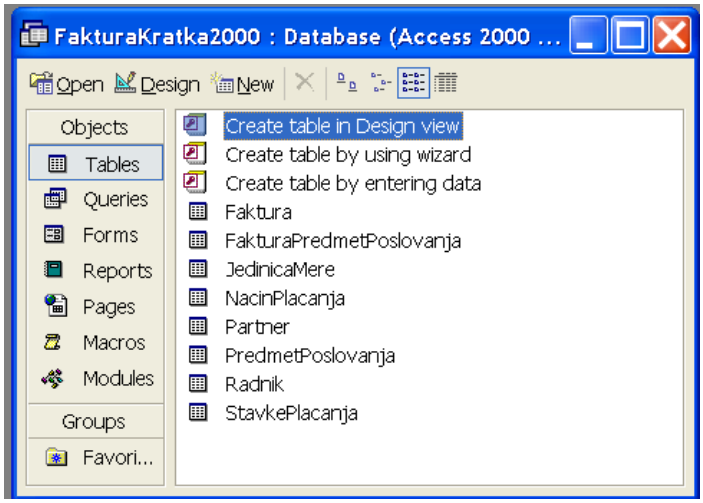
Ako ste početnik, najbolje rešenje je kreiranje kostura aplikacije putem Access-ovih

"čarobnjaka". Access nudi više tipova aplikacija koje se mogu kreirati korišćenjem "čarobnjaka" (adresar, kolekcija slika, vinoteka, fonoteka, videoteka, evidencija donacija, evidencija knjiga, evidencija članstva,...). Kad izaberete tip aplikacije, otvara se dijalog prozor za izbor tabela, polja u tabeli i po želji test podataka. Nakon izbora polja u narednim dijalog prozorima definišete stil aplikacije (izgled formi), tipove izveštaja i naziv aplikacije. Pritisnite dugme Finish i aplikacija će za par sekundi biti kreirana.

Database prozor (Database Window)

Bilo koji način otvaranje baze da odaberete, otvoriće vam se osnovni prozor za rad u Access-u - Database prozor. Podeljen je na na osnovne Access objekte:

- Tables (tabele) - fizički nosioci podataka
- Queries (upiti) - omogućavaju manipulaciju sa podacima iz tabela
- Forms (forme) - obrasci koji služe za unos podataka i za komunikaciju sa krajnjim korisnikom
- Reports (izveštaji) - obrasci za štampu
- Pages (stranice) - izrada Web stranice
- Macros (makroi) - definisani skup akcija za manipulaciju sa objektima
- Modules (moduli) - kolekcija procedura i funkcija pisanih u Visual Basic.



Slika 20. Osnovni prozor za rad u Access-u - Database prozor

Od svih objekata samo se tabele koriste za skaldištenje informacija. Ostale se koriste za analiziranje, preuzimanje, prikazivanje ili objavljivanje informacija iz tabela kao i za upravljanje i manipulaciju njima.

Tabele

Tabele(table) su osnovni objekat u Access-u i one predstavljaju fizički nosioce podataka. Na osnovu relacionog modela razvijena je teorija dizajniranja baze podataka. Dizajniranje baze podataka prestavlja preslikavanje realnog sveta može se izvoditi generisanjem fizičkog modela iz CASE alata (npr. ERwin) gde se definišu svi elementi buduće baze i njihover relacije ili manuelnim dizajniranjem tabela i njihovih relacija. Dobro dizajnirana baza podataka pretpostavlja prvi način rada dok drugi se preporučuje početnicima koji neznaju da rade sa CASE alatima i to samo za baze do 5 tabela.

Prednosti dobro dizajnirane baze podataka su mnogobrojne, a neke od njih su:

- unos, ažuriranje i brisanje podataka su mnogo efikasniji
- pretraživanje, sumiranje i kreiranje izvještaja su mnogo lakša i brža
- pošto je baza podataka izrađena na temeljima dobro formulisanog modela, ona je predvidiva
- pošto je informacija sačuvana u bazi podataka a ne u aplikaciji, baza podataka predstavlja neku vrstu samo-dokumentacije.

Tabele u relacionom modelu se upotrebljavaju da predstave "stvari" iz realnog sveta. Svaka tabela treba da prestavlja samo jedan tip "stvari". Ove stvari mogu biti objekti ili događaji iz realnog sveta. Primeri objekata iz realnog sveta su RADNIK, FAKTURA, PARTNER, a primeri događaja iz realnog sveta su IZRADA FAKTURA, EVIDENCIJA PLACANJA i dr.

Tabele su sačinjene od redova (slogovi) i kolona (atributi). Relacioni model nalaže da svaki red u tabeli mora biti jedinstven. Ako dozvolite dupliranje redova u tabeli, tada ne postoji način da programski odredite jedinstvenu adresu datog reda. Dupliranje redova doprinosi stvaranju svih oblika dvoznačnosti, a i problema. Jedini način da tabela (tj. redovi u tabeli) bude jednoznačno određena jeste da joj dodelite primarni ključ (primary key).

Primarni ključ je kolona ili skup kolona u tabeli koje sadrže jedinstvene vrednosti u datoj tabeli. Svaka tabela može imati imati jedan i samo jedan primarni ključ, čak i ako u tabeli postoji više kolona koje sadrže jednoznačne vrednosti. Sve kolone (ili kombinacije kolona) koje sadrže jedinstvene vrednosti su takozvani kanditati za ključ. Između tih kandidata za ključ biće izabran primarni ključ. Svi ostali, neizabrani kanditati za ključ su alternativni ključevi (alternate keys).

Ključevi mogu biti prosti (simple key) i složeni (composite key). Prosti ključ je ključ koji je

sastavljen od jedne kolone, a složeni ključ je sastavljen od dve ili više kolona. Odluka koji će do kandidata za ključ biti primarni zavisi od nekoliko principa odlučivanja:

- princip minimalnosti (izbor najmanje neophodnih kolona)
- princip stabilnosti (izbor ključa koji se najmanje menja)
- princip jednostavnosti (izbor ključa koji je i jednostavan i familirijan sa korisnikom).

Iako su primarni ključevi u funkciji induvidualnih tabela, ako vi kreirate bazu podataka koja se sastoji od tabela koje su međusobno nepovezane i nezavisne, tada i nemate baš potrebe za njim. Primarni ključ postaje neophodan kad vi počnete kreirati relacije koje udružuju više tabela međusobno u bazi tj. definišu se spoljni ključevi.

Spoljni ključ (foreign key) je kolona u jednoj tabeli koja se referencira na primarni ključ u drugoj tabeli. Na primer, neka imamo tabelu ODELJENJE u koju smo smestili sve nazive odeljenja kao i njihove šifre koje su ujedno i primarni ključ. U drugoj tabeli RADNIK se čuvaju osnovni podaci o radnicima firme kao što su šifra radnika, prezime, ime, datum rođenja, mesto rođenja , sifra odeljenja itd. Kolona sifra odeljenja predstavlja spoljni ključ jer se referencira na odgovarajući red u tabeli ODELJENJE. Mnogo je značajno da i spoljni ključ i primarni ključ "vuku" zajedničke vrednosti iz istog domena.

Jedini način da obezbedimo da vrednosti spoljnog ključa jedne tabele (u našem primeru mesto rođenja u tabeli RADNIK) budu podskup vrednosti primarnog ključa druge tabele (šifra odeljenja u tabeli ODELJENJE) je kreiranje relacije između tih tabela.

U Access-u, tabele mogu biti međusobno povezane pomoću tri vrste relacija:

- One-to-One (za svaki red iz prve tabele postoji najviše jedan red u drugoj tabeli)
- One-to-Many (za svaki red iz prve tabele postoji nula, jedan ili više redova u drugoj tabeli)
- Many-to-Many (za svaki red iz prve tabele postoji više redova u drugoj tabeli, i za svaki red u drugoj tabeli postoji više redova u prvoj tabeli).

Nakon ovog teorijskog uvoda o osnovama relacionog modela i definisanja osnovnih pojmova vreme je krenemo sa tabelama u Access-u.

Kreiranje tabela

Access omogućava nekoliko načina kreiranja tabela. Bilo koji način da izaberete, prvi korak je pritisak na dugme New u Database window-u (pod uslovom da ste izabrali karticu Table).

SQL Naredbom CREATE TABLE kreira se "prazna" tabela sa datim imenom i imenima i tipom kolona definisanih u okviru skript datoteke iz ERwin-a.

Naziv tabele i kolone moraju zadovoljiti uslove koji su već ispoštovani u predhodnom koraku modeliranja podataka:

- Prvo mesto mora biti slovo
- Duzina naziva je do 30 karaktera
- Naziv se može sastojati od alfanumerčkih karaktera,
- brojeva i specijalnih karaktera(Ž,#,$),
- Mogu se koristiti i mala i velika slova za isto ime i
- Ne mogu se duplicirati imena drugih tabela, sinonima
- ili pogleda u vašem account-u.

Osnovne smernice u definisanju imena tabela su:

- Koristiti opisna imena za tabele, kolone, indekse i druge objekte,
- Budite dosledni ako koristite skraćenice i
- Koristite ista imena za opis istih tabela ili atributa
- Usvojte za nazive tabela ili kolona jedninu ili mnozinu.

Primer: U cilju formiranja tabela ODELENJE i RADNICI pišemo naredbu:

```
CREATE TABLE ODELENJE (SIFRAP  CHAR (2) ,;
                       NAZIV  CHAR (14),;
                       MESTO  CHAR (13))
```

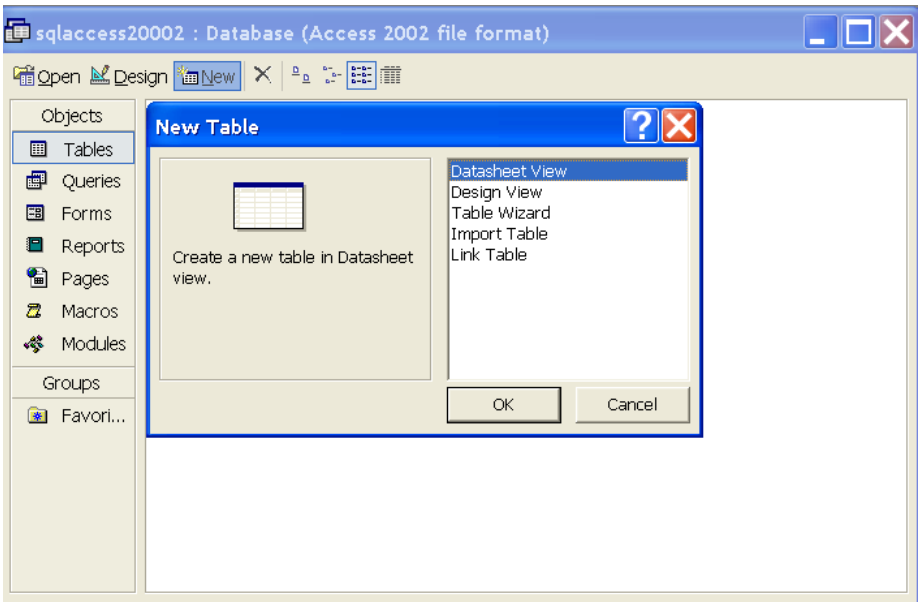
U `CREATE TABLE` naredbi specificira se:

- Naziv tabele(npr.ODELENJE)
- Imena kolona tabele ODELENJE (SIFRAP, NAZIV, MESTO) i
- Tip podatka koji sadrži svaka kolona.

Kreirajući tabelu ODELENJE specificirali smo kolonu SIFRAP, NAZIV i MESTO koje sadrže bilo koji karakter: slova, brojeve ili znake interpunkcije, itd.
Na kraju se specificira maksimalna dužina svake vrednosti koja može biti memorisana u kolonama. Npr. u našoj naredbi CREATE TABLE, mi smo specificirali da ni jedna vrednost kolone MESTO ne može biti duža od 13 karaktera - MESTO CHAR (13).
Tabelu RADNICI se kreira na sledeći način:

```
create table RADNICI;
( SIFRAR    CHAR(4),;
  PREZIME   CHAR(10),;
  DATUMZ    DATE,;
  PLATA     NUMBER(8,2),;
  STIMUL    NUMBER(8,2),;
  SIFRAP    CHAR(2),;
  RADNOM    CHAR(2);
   )
```

Na sledećoj slici prikazane su opcije za kreiranje tabela.



Slika 21. Opcije za kreiranje tabela

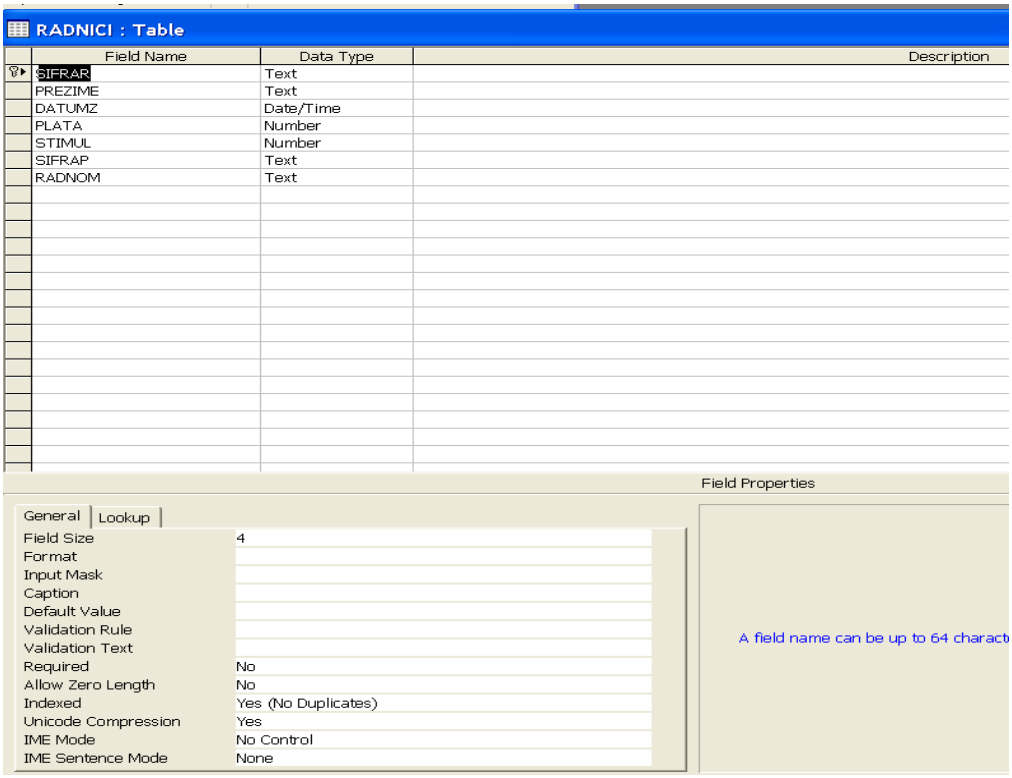
Pritiskom na dugme New otvara se dijalog prozor za izbor načina kreiranja tabela i to:

- Datasheet View - Opcija koja se pojavila u Access 95. Otvara se prikaz sličan kao u Excel-u (grid prikaz) gde možete da direktno unosite podatke u kolone, a na kraju Access će sam na osnovu unesenih podataka da dodeli nazive kolonama i odgovarajuće tipove podataka. Ukoliko niste zadovoljni dodeljenim imenima kolona možete ih preimenovati uz pomoć opcije Rename Column shortcut menija (aktivira se na klik desnog dugmeta miša). Takođe, naziv kolone i tip podataka možete promeniti i iz dizajn prikaza izborom opcije Design View menija View ili klikom na odgovarajuće dugme toolbar-a.
- Design View - Opcija koja se najčešće koristi kod kreiranja tabela. Prozor Design View-a je podeljen na dva dela, u prvom delu prozora je grid u kome unosite naziv kolone, tip podatak, i opis te kolone, dok je u drugom delu prozora opcije za definisanje obeležja kolone.
- Table Wizard - Opcija koja se uglavnom preporučuje početnicima. Access nudi, zaista, impozantnu biblioteku raznih vrsta tabela i kolona, kao i test podataka. Za ovu opciju dovoljno je malo znanje engleskog, odgovorite na nekoliko pitanja, izaberete odgovarajući tip tabele i vrste kolona i tabela je kreirana sa test podacima.
- Import Table - Omogućava da iskoristite tabele iz neke druge aplikacije, bilo da je ona kreirana u Access-u, DBase-u, FoxPro-u ili je to neka Excel tabela ili tekstualna datoteka. Ova opcija je korisna kada želite da unapredite neku aplikaciju koja je na primer pisana za rad pod DOS-om, a ne želite da zadržite strukturu, relacioni model i podatke.

- Link Table - Slična prethodnoj opciji. Osnovna razlika je što kod Import opcije vi uvozite (ugrađujete) tabelu u Access i ona postaje Access tabela, dok kod ove opcije vi zadržavate taj tip tabele i praktično samo koristite podatke.

Definisanje svojstava kolona (polja)

U prethodnom tekstu smo opisali načine kreiranja tabela, a u ovom poglavlju objasnićemo tipove podataka koje Access podržava kao i kreiranje indeksa i definisanje svojstava.



Slika 22 Definisanje svojstava kolona

Već je rečeno da je dizajn prozor podeljen na dva dela i u gornjem delu unosimo:

- Naziv kolone (Field Name) - prestavlja naziv polja u tabeli i on ne može da bude veći od 64 karaktera s tim što se ne smeju koristiti sledeći karakteri: tačka (.), uzvik (!), uglaste zagrade (Š Ć), apostrof (') i karaktera prazno.
- Tip podatka (Data Type) - definiše se tip podatka u polju (koloni). Access podržava sledeće tipove podataka u tabelama: Text, Memo, Number, Date/Time, Curency, AutoNumber, Yes/No, Ole Object, HyperLink i Lookup Wizard. U trenutku određivanja tipa polja u tabeli, u donjem delu dizajn prozora pojavljuje se niz novih polja u kojima se definišu svojstva za

izabranu kolonu u tabeli. Svojstva kolona su podeljene na dve grupe General (tabela 1) i Lookup. U Lookup grupi se vrši definisanje tipa kontrole koja će prikazivati vrednosti ovog polja na obrascima (formama). Podrazumevani tip kontrole za Text, Number, Memo, Date/Time, Currency, AutoNumber je Text Box, a za Yes/No polja je Check Box.

- Opis kolone (Description) - sadrži opis polja u tabeli. Opis može biti tekst dužine 255 karaktera. Uneti tekst u ovoj koloni pojavljivaće se na statusnoj liniji kada je fokus na tom polju ili na kontroli koja je nastala od tog polja.

Tabela 1

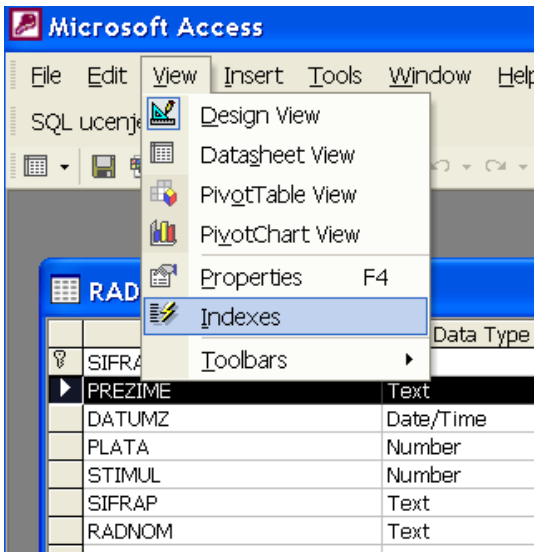
Naziv Obeležja	**Opis svojstva**
Field Size	veličina polja
Format	format u kojem se cuvaju podaci
Input Mask	maska prikaza podataka
Caption	naziv polja prilikom prikazavanja u formi
Default Value	inicijalna vrednost u polju
Validaton Rule	izraz koji definiše proveru tipa podatka
Validation Text	tekst koji se pojavljuje ukoliko nije zadovoljen izraz svojstva Validation Rule
Required	ukoliko je postavljeno na Yes, onda je u izabranom polju u tabeli obavezan unos
Allow Zero Length	ukoliko je postavljeno na Yes, onda je u izabranom polju u tabeli dozvoljen unos niza znakova dužine nula
Indexed	definisanje da li će izabrano polje u tabeli biti indeksirano
Decimal Places	mesto u polju gde se nalazi decimalni zarez

Na nivou tabele je, takođe, moguće definisati nekoliko svojstava i to : Description, Validation Rule, Validation Text. Ova svojstva imaju identična značenja kao i pri definisanju kolona.

Definisanje indeksa

Definisanje indeksa u tabeli je vrlo važna radnja i treba je obavljati dosta oprezno jer od toga zavise i same perfomanse aplikacije. Indeks se postavlja na određeno polje radi bržeg pretraživanja ili sortiranja podataka. Broj indeksa u tabeli može biti veliki, ali proporcijalno sa povećanjem indeksa smanjuju se perfomanse aplikacije. Indeks se može maksimalno sastojati

od deset polja iz tabele.



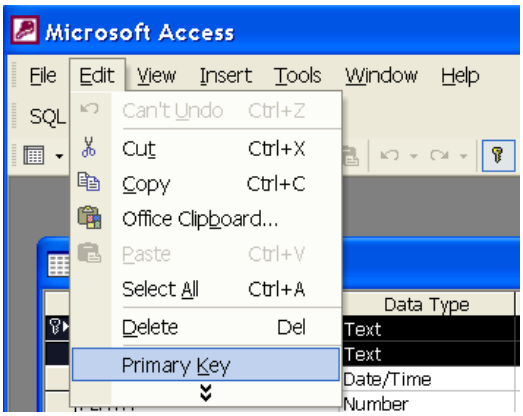
Slika 23 Definisanje indeksa

Definisanje indeksa je moguće uraditi u polju obeležja Indexed ili u prozoru za kreiranje indeksa (aktivira se izborom opcije Indexes menija View) i za svaki definisani indeks dodeljuju se vrednosti obeležjima:

- Primary - definisanje primarnog ključa
- Unique - indeks može biti jedinstven (Yes) ili moguće je postojanje više slogova sa istim sadržajem (No)
- Ignore Nulls - u indeksu su dozvoljene Null vrednosti (Yes) ili nisu (No).

Nad poljima čiji je tip podataka OLE Object ili Memo ne može se vršiti indeksiranje.

Primarni ključ se postavlja selektovanjem odgovarajućeg polja ili skupa polja (držite taster Ctrl) i izborom opcije Primary Key iz menija Edit ili klikom na dugme Primary Key na toolbar-u.



Slika 24 Definisanje primarnog kljuca

Access sve sistemske podatke koji predstavljaju aplikaciju čuva u deset sistemskih tabela čiji prikaz se aktivira izborom opcije Show System Object menija Tools->Options. Svaka tabela ima fiksnu strukturu i specijalizovana je za pojedine objekte u aplikaciji. Tabele su: MSysACEs, MSysColumns, MSysIMEXColumns, MSysIMEXSpaces, MSysIndexes, MSysMacros, MSysObjects, MSysQueries, MSysRelationships, MSysToolbars.

Povezivanje tabela (kreiranje relacija)

Relacija u Access-u se kreira preko posebnog editora relacija, koji se poziva preko opcije Relationships menija Tools. Aktiviranjem opcije Relationships otvara se prozor u kome se vrši definisanje relacija.

Prvi korak je dodavanje tabela za međusobno povezivanje i to se vrši preko opcije Show Table menija Relationships.

Kada smo izabrali tabele za prikazivanje, drugi korak je uspostavljanje relacija. Relacija se uspostavlja na sledeći način; mišem izaberete polje iz tabele "roditelja" i držeći pritisnuto dugme miša vučete ka tabeli "detetu". U tabeli "deteta" izaberete odgovarajuće polje sa kojim želite da uspostavite relaciju sa tabelom "roditelja" i pustite taster miša. Nakon što pustite taster miša, otvoriće se dijalog prozor u kome su prikazani polja između kojih je uspostavljena relacija.

U donjem delu prozora definišete da li želite da preko ove veze obezbeđujete kontrolu referencijalnog integriteta.

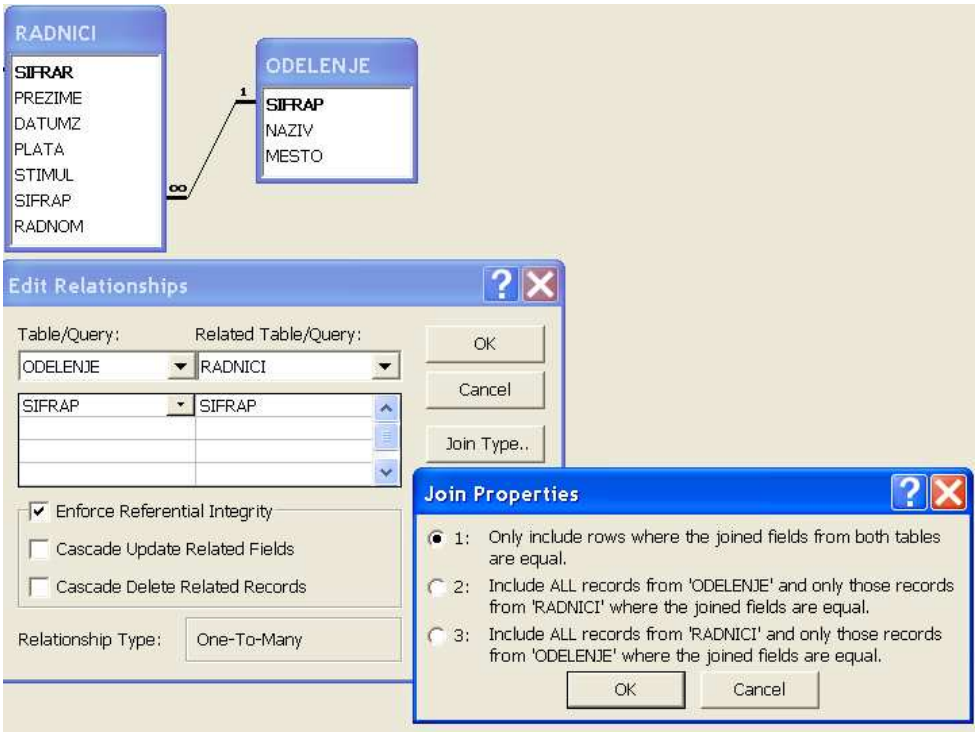
Ukoliko želite da vršite kontrolu referencijalnog integriteta izaberite opciju Enforce Referential Integrity.

Izborom ove opcije omogućeno je i definisanje tipa relacione veze, kao i definisanje akcija za održavanje integriteta.

Akcija Cascade Update Related Fields omogućava da u slučaju izmene sadržaja primarnog

ključa tabele "roditelja" se izvrši automatsko ažuriranje spoljnih ključeva vezanih redova u tabeli "deteta".

Kada se vrši brisanje redova (slogova) u tabeli "roditelja" akcija Cascade Delete Related Records izaziva brisanje vezanih redova (slogova) u tabeli "deteta".



Slika 25 Povezivanje tabela

Opcijom Join Type definiše se način povezivanja tabela:

1. biraju se samo redovi koji u veznim poljima imaju iste sadržaje (Iner Join);
2. biraju se svi redovi iz prve tabele i samo oni redovi iz druge tabele čiji je sadržaj veznih polja jednak sa sadržajem u prvoj tabeli (Left Join);
3. biraju se svi redovi iz druge tabele i samo oni redovi iz prve tabele čiji je sadržaj veznih polja jednak sadržaju veznih polja druge tabele (Right Join).

Kada u Access-u radite aplikaciju koja ima dosa tabela i veza tada je preporučljivo da relacioni model uradite u nekom CASE alatu (CA ERwin, Microsoft Visio 2000...). Nakon završenog modela moderni CASE alati imaju mogućnost generisanja tabela i relacija u Access-u.

SQL upitni jezik

Neproceduralni jezik SQL (Structured Query Language) dizajniran je tako da ga sa uspehom mogu koristiti i ljudi bez tehničkih znanja s područja obrade podataka, takozvani krajnji korisnici. SQL jezik je neproceduralan, jer specificira operacije u smislu ŠTA treba uraditi, a ne KAKO.

SQL upitni jezik omogućuje da korisnici mogu ad hoc formulisati i postavljati pitanja i veoma brzo dobijati odgovor, a da pri tom ne zavise od saradnje sa programerom. Programeri su na taj način rasterećeni i posvećuju se izradi kvalitetnih programa za aplikacije, koje zahtevaju mnogo tehničkog znanja (npr., veoma frekventne transakcije, kod kojih je važno da se postigne što kraće vreme odziva).

SQL omogućuje povezivanje sa klasičnim višim programskim jezicima, kao što su: COBOL, PL/I, PASCAL, FORTRAN, C i dr.

SQL jezik je usvojio Komitet Američkog nacionalnog Instituta za standarde (ANSI) kao standardni jezik relacionih baza podataka.

U relacionim bazama podataka se definiše struktura podataka u obliku tabela. SQL relacioni jezik pravi nove tabele, definišući podskupove i/ili kombinujući postojeće tabele. Dakle, SQL omogućuje da se jednom relacionom naredbom mogu čitati, ažurirati, ili brisati redovi memorisani u relacionoj bazi podataka.

SQL sadrži:

- konstrukcije analogne relacionoj algebri,
- konstrukcije analogne relacionom računu,
- jezik za opis baza podataka ,
- vezivanje SQL sa nekim standardnim jezikom (CURSOR operacija),
- kontrolne konstrukcije za upravljanje konkurentnom obradom,
- oporavak baze podataka,
- definisanje zaštite baze podataka,
- definisanje integriteta baze podataka.

SQL je jezik za:

- interaktivno definisanje baze podataka (Data Definition Language ili DDL) - jezik kojim treba da se omoguće kreiranje, dodavanje, brisanje tabela, pogleda, sinonima i indeksa, pa stoga poseduje sledeće komande:

CREATE TABLE	▶	kreiranje tabele,
CREATE VIEW	▶	kreiranje pogleda,
CREATE SYNONIM	▶	kreiranje sinonima, *
CREATE INDEX	▶	kreiranje indeksa,
ALTER TABLE	▶	dodavanje kolona ili redefinisanje u tabeli,
DROP TABLE	▶	brisanje tabele,
DROP VIEW	▶	brisanje pogleda,*
DROP SYNONYM	▶	brisanje sinonima,*
DROP INDEX	▶	brisanje indeksa;

- pretraživanje podataka gde treba da se omogući izbor redova iz tabele;
- manipulaciju podacima (Data Manipulation Language ili DML) koja treba da omogući ubacivanje redova, izmene i brisanje memorisanih podataka u tabeli:

INSERT	▶	ubacivanje redova u tabelu,
UPDATE	▶	izmena memorisanih podataka,
DELETE	▶	brisanje memorisanih podataka;

- upravljanje podacima (Data Control Language - DCL) kojim treba da se omoguće dodavanje i opoziv privilegija, prenos transakcija iz buffer-a u tabele, zaključavanje tabela i definisanje pogleda:

GRANT CONNECT	▶	dodeljivanje privilegije,
REVOKE	▶	opoziv privilegije,
COMMIT	▶	prenos transakcija iz bafera u tabelu,
ROLLBACK	▶	poništavanje izmena u tabelama pre commita,
LOCK TABLE	▶	zaključavanje tabele,*
AUDIT	▶	definisanje ORACLE pregleda.*

Napomena: Zvezdica (*) znači da su naredbe definisane u okviru ORACLE verzije SQL-a.

Na osnovu izabranog SUBP pristupa se u sledećem koraku - definisanju tabela i kolona.

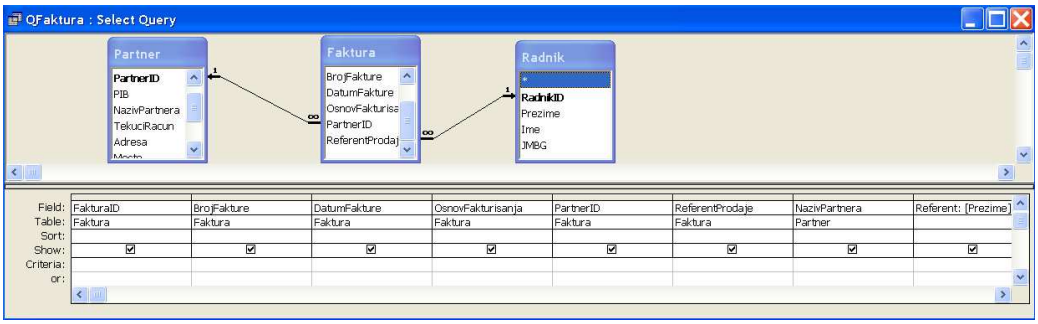
Upiti

Prava snaga sistema za upravljanje bazom podataka je sposobnost da vidite podatke koje želite, u redosledu koji vi želite. Sa upitima (query) vi možete postaviti pitanja o podacima koji se nalaze u vašim tabelama. Podatke u upitu možete "izvući" iz jedne ili više tabela. Nakon što Access pronađe sve podatke koje predstavljaju odgovor na postavljeno pitanje, vi možete da vidite podatke u tabelarnom prikazu, da ih analizirate ili pak da ovaj upit postavite kao osnovu za izradu obrasca (formi), izveštaja, grafikona ili čak možete iskoristiti za neki drugi upit.

Upit su mehanizmi za manipulaciju sa podacima. Najčešće korišćeni tipovi upita su takozvani

select upiti. Upiti za izbor (Select Querys) izdvajaju podatke iz jedne ili više tabela, i prikazuju te podatke u tabelarnom obliku. Sa select upitima, vi možete pregledati tražene podatke iz vaših tabela, analizirati podatke i čak menjati podatke. Nakon što pokrenete select upit, Access skuplja podatke koje ste tražili u dynaset.

Dynaset izgleda i radi kao tabele, ali to nije klasična tabela - to je dinamički "pogled" na podatke iz jedne ili više tabela, selektovani i sortirani kao što je specifirano u upitu. Vi možete da unosite i menjate podatke u dynaset poljima, identično kao i u tabelama.



Slika 26 Definisanje upita

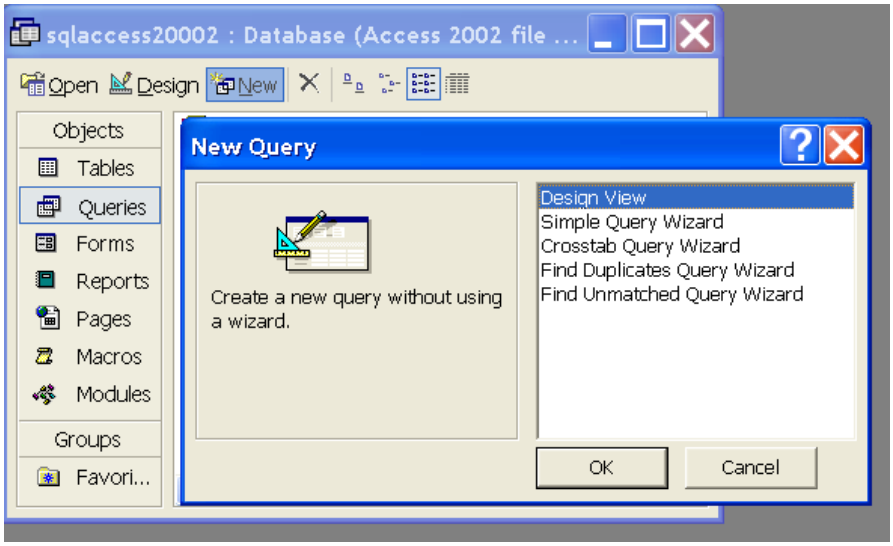
Select upit je najčešće korišćeni tip upita, ali Access podržava i druge tipove upita:

- Upiti unakrsnih tabela (Crosstab queries) sumiraju podatke iz jedne ili više tabela u obliku radne tabele. Ovakvi upiti su korisni za analiziranje podataka i izradu grafika ili dijagrama, na osnovu sume vrednosti numeričkih polja većeg broja zapisa. Sa crosstab upitima, vi možete da sabirate ogromnu količinu informacija u lako čitljivom formatu,
- Action upiti - pravi izmene nad velikim brojem slogova samo jednom operacijom. Upotrebom action upita možete napraviti novu tabelu sa željenom strukturom i podacima koje ste specificirali u upitu, da obrišete podatke iz tabele, dodate veći broj novih slogova u tabelu, napravite promene u određenim poljima za određene slogove u određenim tabelama,
- Union upiti - povezuje odgovarajuća polja iz dve ili više tabela,
- Pass-trough upiti - šalje komande SQL bazama podataka kao što su Microsoft SQL Server ili Sybase SQL Server,
- Data-definition upiti - kreira, menja ili briše tabele u Access bazi korišćenjem SQL naredbi.

Kreiranje upita

Slično kao i kod kreiranja tabela, kreiranje upita se može uraditi na nekoliko načina. Prvi korak je da u dabase window izaberete karticu pod nazivom Query. Nakon toga pritisnite dugme New i

otvoriće se dijalog za izbor načina kreiranja upita. U ovom dijalogu se nameću u osnovi dve metodologije izrade upita – izrada upita pomoću Wizard-a i izrada upita u dizajn modu.



Slika 27 Kreiranje upita

U dijalog prozoru su ponuđeni sledeći načini kreiranja upita:

- Design View - Otvara prozor za dizajniranje upita u dizajn modu.
- Simple Query Wizard - Omogućava da odgovorom na nekoliko jednostanih pitanja kreirate jednostavne upite. Kod kreiranja upita na ovakav način prvo se vrši izbor tabele ili upita nad kojim se pravi novi upit, kao i izbor odgovarajućih polja nad kojim se želi vršiti pretraživanje, sumiranje, prebrojavanje i slično. Drugi korak je izbor da li želite klasičan upit bez opcija za sumiranje ili sa. Na kraju se još daje naziv upitu. Pritiskom na dugme Finish upit je gotov. Kasnije možete preći u Design View i izvršiti još neke naknadne izmene.
- Crosstab Query Wizard - je "čarobnjak" koji vam pomaže pri izradi ovog tipa upita koji i nije tako jednostavno iz prve napraviti u Design View-u. Kao i kod svih "čarobnjaka" prvo se bira tabela ili upit nad kojom se pravi novi upit. Nakon što izaberete tabelu, u sledećem dijalog prozoru se bira polje ili polja čije će vrednosti biti nazivi redova, a u drugom dijalog prozoru birate polje čije će vrednosti biti nazivi kolona. Klasičan pristup do sada je bio da je broj redova bio promenljiv, a broj kolona statičan. Ovaj tip upita omogućava da broj kolona bude promenljiv. Nakon što definišemo nazive redova i kolona, u trećem dijalogu definišemo presek redova i kolona za koji vežemo treće polje iz tabele i jednu od ponuđenih agregacionih funkcija.
- Find Duplicates Query - je tip "čarobanjaka" koji vam omogućava kreiranje upita za pronalaženje istih vrednosti u jednom ili više polja tabele nad kojima je napravljen upit.

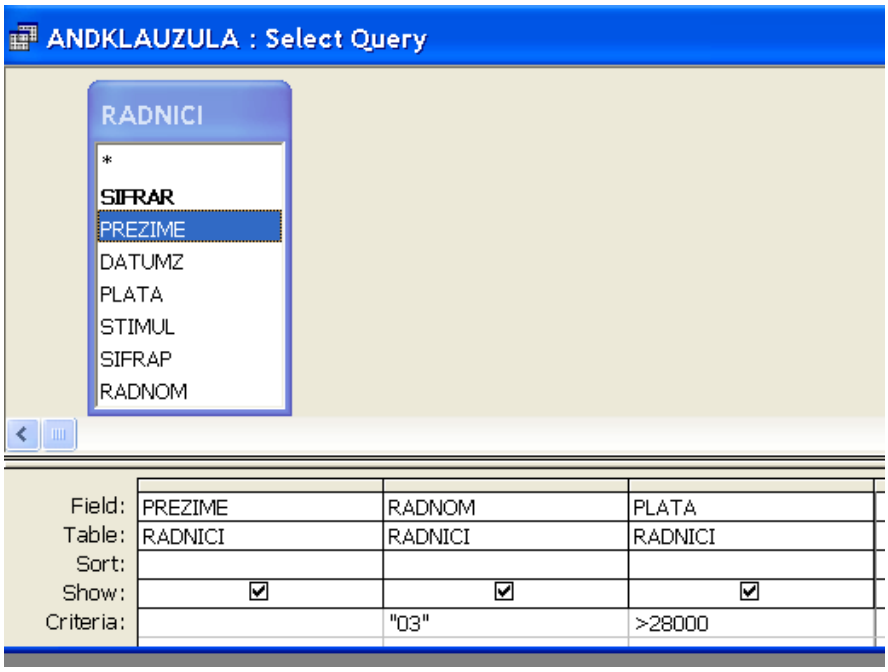
- Find Unmatched Query - je tip "čarobnjaka" koji kreira upit za pronalaženje vrednosti iz jedne tabele koji ne odgovaraju vrednostima iz druge tabele.

Query By Example (QBE)

Do pojave Access-a jedini način kreiranja upita je bio pisanje SQL naredbi. Njegovom pojavom promovisan je koncept vizuelnog kreiranja upita, a istovremeno i SQL koda. Prozor QBE dizajnera ili jednostavnije Design View-a je podeljen na dva dela. U gornjem delu prozora nalazi se prikaz selektovanih tabela nad kojim se pravi upit, a u donjem delu se nalazi spreadsheet u kome su prikazana polja nad kojim se vrši pretraživanje i definisanje kriterijuma za pretraživanje.

Na primer:

Izlistajmo prezime radno mesto i platu za sve radnike na radnom mestu 02.



Slika 28 Primer za Query By Example

Dodavanje tabela u upit se vrši preko opcije Show Table menija Query ili pritiskom na dugme Show Table na toolbar-u. Nakon toga se otvara dijalog prozor u kome su ponuđene sve tabele i upiti koji se nalaze trenutno u aplikaciji. Izaberete mišem odgovarajuću tabelu ili upit i kliknite na dugme Add.

U zaglavljima spreadsheet-a se nalaze nazivi izabranih polja (prvi red) i tabele (drugi red). Najčešći način dodavanja polja je duplim klikom na polje tabele prikazane u gornjem delu

prozora.

U trećem redu se vrši postavljanje kriterijuma za sortiranje - rastući (Ascending), opadajući (Descending) ili da se ne sortira uopšte (not sorted).

Sledeći red (Show) definiše da li će se to polje pri izvršavanju upita prikazivati; ako je "čekirano" tada se prikazuje, u suprotnom se ne prikazuje.

U redu Criteria definiše se uslov za prikaz podataka iz odgovarajućeg polja. U ovo polje se unosi logički izraz, a ako tih izraza ima više oni se spajaju pomoću logičkih operatora And i Or.

Svi uslovi u jednom Criteria redu se smatraju da su povezani And logičkim operatorom, dok uslovi u jednoj Criteria koloni smatraju se povezani Or logičkim operatorom.

Takođe, uslove možete kreirati korišćenjem Expression Builder-a koji se aktivira pomoću odgovarajuće opcije iz shortcut menija ili pritiskom na odgovarajuće dugme na toolbar-u.

Prilikom kreiranja upita za sumiranje, potrebno je uključiti red za prikazivanje agregacionih funkcija za sumiranje. Prikazivanje za sumiranje se vrši izborom opcije Totals menija View nakon čega će se u donjem delu prozora pojaviti još jedan red pod nazivom Totals. U redu Totals moguće je izabrati sledeće funkcije:

- Group by - vrši grupisanje po izabranom polju ili poljima. Ova opcija se podrazumeva kod običnih upita.
- Sum - vrši sumiranje izabranog polja u odnosu na polja po kojima se grupiše.
- Avg - vrši izračunavanje srednje vrednosti izabranog polja u odnosu na polja po kojima se grupiše.
- Min - pronalazi minimalnu vrednost izabranog polja.
- Max - pronalazi maksimalnu vrednost izabranog polja.
- Count - izračunava broj različitih vrednosti izabranog polja u odnosu na polja koja po kojima se grupiše.
- StDev - izračunava standardnu devijaciju izabranog polja.
- Var - izračunava varijansu izabranog polja.
- First - pronalazi vrednost prvog sloga koji se prikazuje u upitu.
- Last - pronalazi vrednost poslednjeg sloga koji se prikazuje u upitu.
- Expression - koristi se kod pisanja jednostavnih izraza. U polje gde se unose nazivi polja ne unosi se naziv polja iz tabele već se piše izraz korišćenjem ugrađenih funkcija.
- Where - definiše uslov za izabrano polje i to polje se ne prikazuje.

Sve upite tipa Totals nije moguće ažurirati.

Takođe, moguće je podesiti broj slogova koji će se prikazati. Podrazumevajuća opcija je da se prikažu svi slogovi koji zadovaljavaju kriterijume postavljene u upitu, ali je moguće prikazati prvih 5, 25, 100 ili 5%, 25% ili bilo koji broj koji vi unesete. Podešavanje se vrši unosom u Combo Box Top Values na toolbar-u ili definišite u svojstvima upita u polju Top Values.

Na kraju, kad smo definisali upit, pomoću opcije Datasheet View možemo videti rezultat našeg upita. Ukoliko želite da izvršite jedan od Action tipova upita (Make Table, Update, Append, Delete), potrebno je izabrati opciju Run menija Query ili izbrati odgovarajuće dugme na toolbar-u.

SQL Editor

Pored QBE u Access-u je moguće direktno pisanje SQL upita. Prelazak na SQL editor se vrši izborom opcijom SQL View iz menija View ili klikom na odgovarajuće dugme na toolbar-u. Ukoliko ste dobri poznavaoci SQL jezika ovo i nije tako loše rešenje, ali ipak je najefikasnije pisanje upita korišćenjem QBE editora. Međutim, kod nekih tipova upita nije moguće izbeći SQL editor, kao što to su SQL specifični upiti (Union, Pass-Trough, Data Definition).

Za predhodno postavljeni QBE upt SQL upit izgleda:

```
SELECT RADNICI.PREZIME, RADNICI.RADNOM, RADNICI.PLATA
FROM RADNICI
WHERE (((RADNICI.RADNOM)="03") AND ((RADNICI.PLATA)>28000));
```

Odgovor na postavljeni upit je:

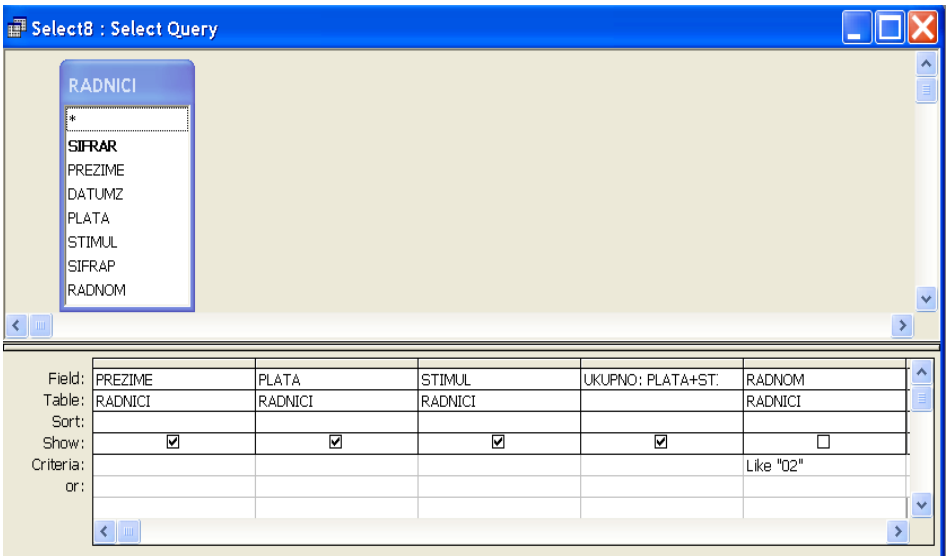
PREZIME	RADNOM	PLATA
JOVIC	03	29750
BOBIC	03	28500

Primeri za SQL i QBE upite

Izlistajmo prezime, platu, stimulaciju i sumu plate i stimulacije za sve RADNIke na radnom mestu 02.

```
SELECT PREZIME,PLATA,STIMUL,PLATA+STIMUL AS UKUPNO;
      WHERE RADNOM ='02'
```

PREZIME	PLATA	STIMUL	UKUPNO
ALAGIC	16000	3000	19000
VUKIC	12500	5000	17500
MARTIC	12500	14000	26500

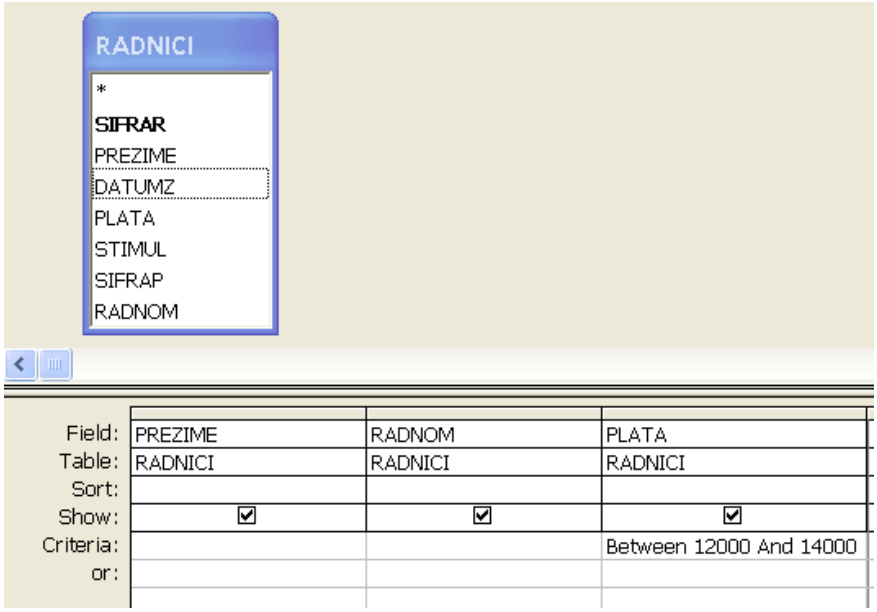


Slika 29 Prikaz SQL upita

Izlistaj zaposlene iz tabele RADNICI, čija je plata u rasponu 12000 i 14000.

```
SELECT PREZIME, PLATA;
       FROM RADNICI;
       WHERE PLATA
BETWEEN 12000 AND 14000
```

prezime	plata
STARCEVIC	8000
ALIMPIC	11000
JAKIC	9500

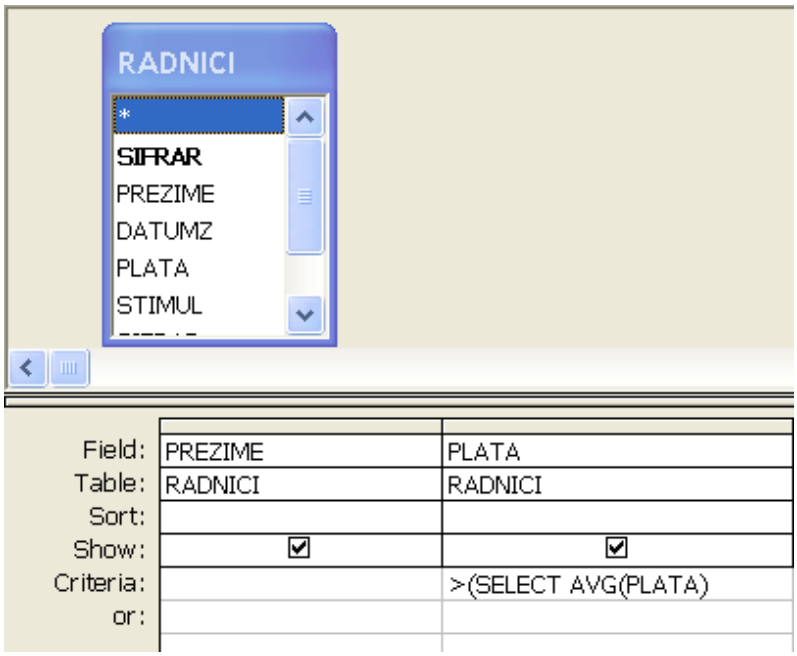


Slika 30 Prikaz SQL upita

Ako želimo da izlistamo sve Radnike koji zaradjuju više od proseka napisaćemo:

```
SELECT PREZIME,PLATA;
       FROM RADNICI;
       WHERE PLATA >;
              (SELECT AVG (PLATA);
              FROM RADNICI)
```

PREZIME	PLATA
JOVIC	29750
BOBIC	28500
CEBIC	24500
SUSIC	30000
KLJAKIC	50000
FILIPIC	30000

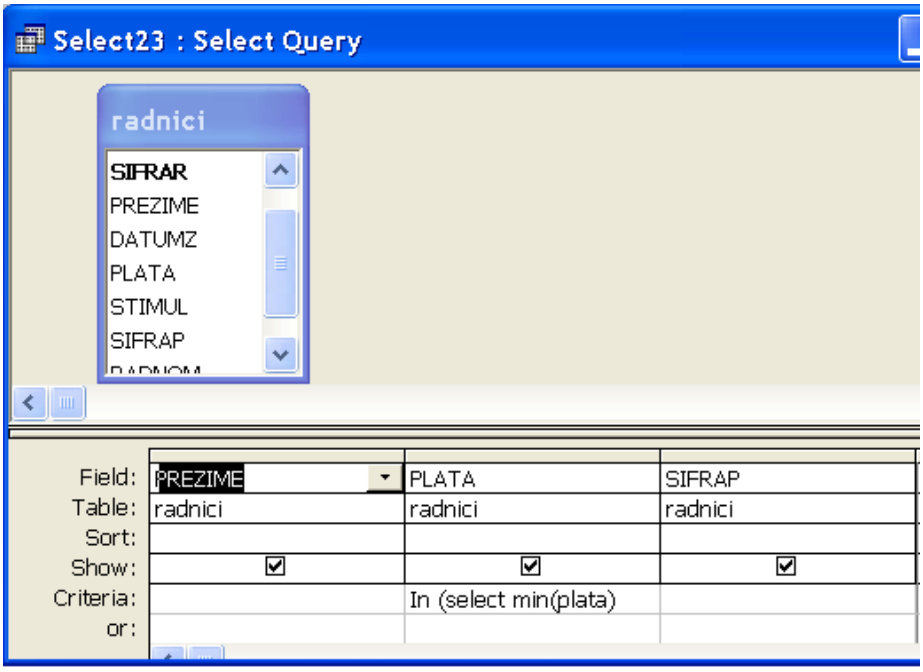


Slika 31 Prikaz SQL upita

Koji su to radnici koji u svakom odeljenju preduzeća imaju namanju platu

```
SELECT PREZIME, PLATA, SIFRAP
FROM RADNICI
WHERE PLATA In (select min(plata)
from radnici
group by sifrap));
```

prezime	plata	sifrap
STARCEVIC	8000	30
ALIMPIC	11000	40
MILIC	13000	20

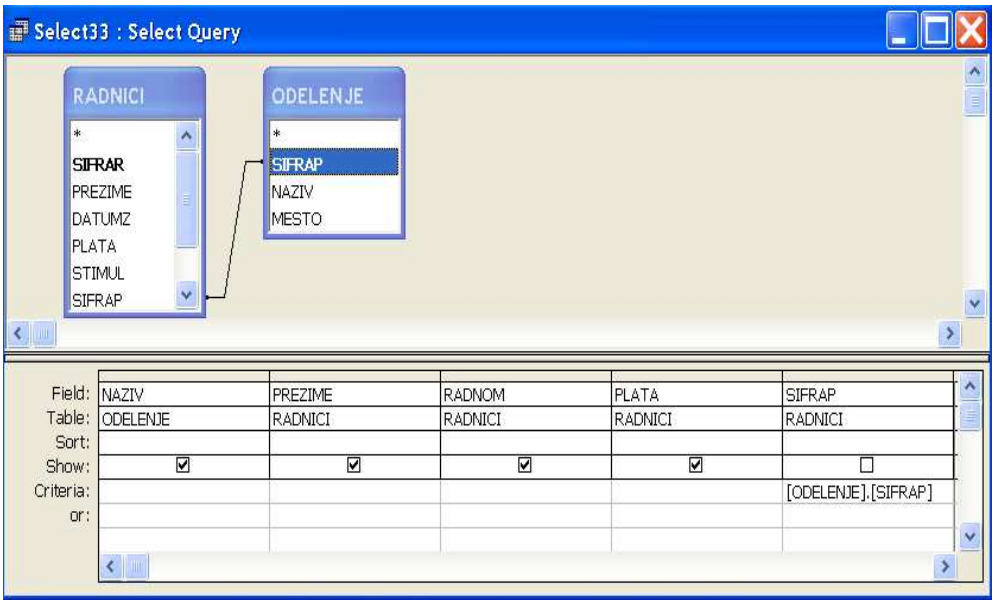


Slika 32 Prikaz SQL upita

Prikazi naziv odeljenja, prezime, platu, radno mesto radnika

```
SELECT              ODELENJE.NAZIV,
RADNICI.PREZIME,
RADNICI.RADNOM, RADNICI.PLATA
FROM RADNICI, ODELENJE
WHERE
(((RADNICI.SIFRAP)=ŠODELENJEĆ.ŠSIF
RAPĆ));
SELECT RADNICI.PREZIME
```

NAZIV	PREZIME	RADNOM	PLATA
RAZVOJ	FILIPIC	04	30000
RAZVOJ	MILIC	01	13000
PRODAJA	STARCEVIC	01	8000
PRODAJA	ALAGIC	02	16000
PRODAJA	VUKIC	02	12500
PRODAJA	MARTIC	02	12500
PRODAJA	BOBIC	03	28500
PRODAJA	SUSIC	04	30000
PRODAJA	KLJAKIC	05	50000
PRODAJA	TUBIC	03	15000
PRODAJA	JAKIC	01	9500
PROIZVODNJA	JOVIC	03	29750
PROIZVODNJA	CEBIC	03	24500
PROIZVODNJA	ALIMPIC	01	11000



Slika 33 Prikaz SQL upita

Obrasci (Forme)

Ekranske forme su osnovni tip objekata u većini SUBP i treba da omoguće korisniku predstavljanje podataka iz baze i unos podataka u bazu. Forme u sebi mogu imati veliki broj drugih objekata (kontrola). Većina SUBP, koji za osnovu imaju MS WINDOWS, podržava tzv. wizard metodologiju za kreiranje formi. Specifičnosti u izradi formi nisu predmet razmatranja ove knjige, pa se čitaoci upućuju na literaturu, u zavisnosti koji su SUBP izabrali. Ovde će biti definisane neke opšte postavke koje se moraju poštovati prilikom definisanja ekranskih formi.

Dakle, ekranske forme treba da ispune sledeće karakteristike:

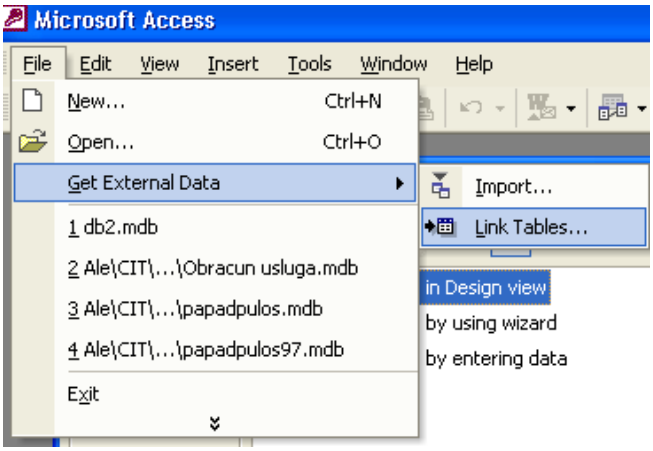
- opšte kakrakteristike:
  - svaka forma mora imati naslov;
  - formi dati samo ono što korisniku treba;
  - obezbediti simetriju i balans na ekranu;
  - u slučaju pojavljivanja više formi, naznačiti u kojoj se formi korisnik nalazi;
  - omogućiti korišćenje malih i velikih slova u tekstu;
  - definisati praznu liniju između svakog paragrafa;
  - tekst poravnavati na levu stranu;
  - biti pažljiv sa skraćenicama i akronimima;
- tabele i liste:
  - redovi i kolone u listama moraju imati imena;
  - listeređati u prepoznatljivom redosledu;
  - koristiti vertikalne kolone, jer su preglednije za čitanje;
  - tekst u kolonama poravnavati nalevo, a brojeve nadesno;
  - svaka peta linija treba da bude prazna linija;
  - ostaviti bar dva mesta prazna između kolona;
  - numerisati liste počev od 1, a ne od 0;
  - razbiti niz slova u kraće podnizove;
- naglašavanje:
  - ne preterivati sa naglašavanjem;

- treptanje i zvuke upotrebljavati samo kao alarm;
- naglašeni tekst treba da bude razumljiv;
- boje birati iz sredine duginog spektra;
- biti dosledan u prikazima i naglašavanju;

▪ unos podataka:

- pri unosu podataka iz formulara kopirati izgled formulara;
- grupisati polja po kategorijama i staviti nazive;
- ne unositi unapred poznate podatke;
- u svakom polju obezbediti memorisanje polja;
- uvesti kad je moguće podrazumevanu (default) vrednost;
- omogućiti help na nivou polja;
- omogućiti slobodno kretanje među poljima za unos podataka;
- forma je jedinica prenosa podataka;
- omogućiti korisniku da odustane od unosa.

Pre nego sto se pristupi izradi klijent strane potrebno je povezati se sa server stranom kao sto je pokazano na skledecoj slici.U okviru opcije File bira se Get External Data/ Link Tables. Povezivanjem sa server starnom nastavlja se rad na klijent strani i izradjuju se forme.



Slika 34 Linkovanje za sever stranom

Forme predstavljaju osnovni objekat u Access-u koja je namenjena za komunikaciju sa korisnikom. Kao i kod papirnih obrazaca koje popunjavate olovkom (na primer, prijava ispita, popunjavanje računa), Access forme vam omogućavaju da na identičan način sakupite podatke. Pored toga što omogućavaju da unesete podatke, pomoću forme možete pregledati podatke, pretraživati po nekom uslovu, ažurirati podatke ili štampati. Korišćenjem Microsoft Access-a možete dizajnirati formu tako da bude jednostavna za unos i pregled podataka, ali i da odgovara odgovarajućem papirnom obrascu.

Kreiranje forme

Prilikom kreiranja nove forme redosled početnih radnji je identičan kao i kod kreiranja novih tabela i upita. Prvo je potrebno da se u Database Window izabere kartica Form. Zatim, pritisnite dugme New i otvoriće vam se dijalog prozor za izbor načina kreiranja formi i Combo box u kome možete da izaberete tabelu ili upit na kome će se bazirati forma. Drugim rečima, vi sada pravite obrazac za unošenje, brisanje, ažuriranje, pregled i pretraživanje podataka iz tabele ili više tabela. U dijalog prozoru su ponuđeni sledeći načini kreiranja formi:

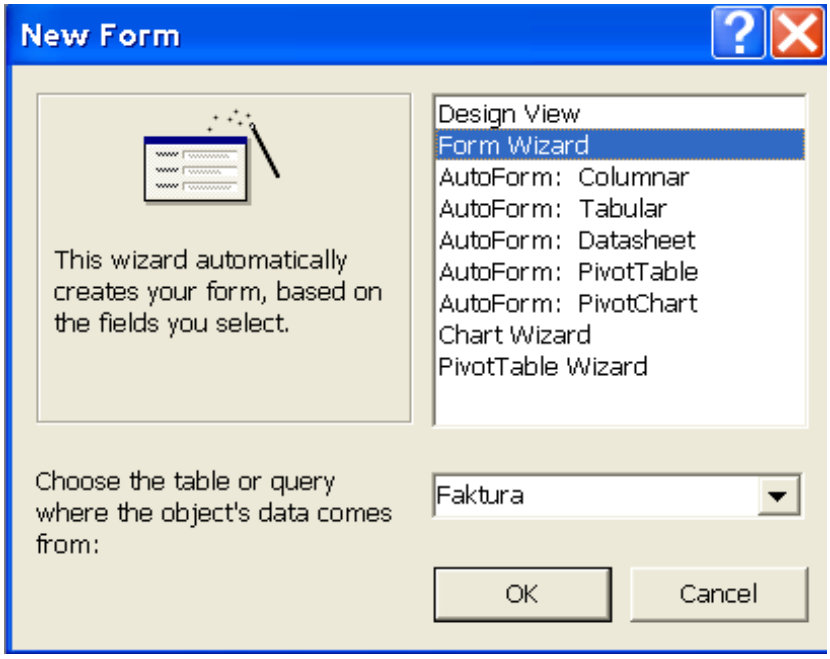
Design View - kreira se prazna forma. Za nju nije vezana ni jedna tabela ili upit i nije kreirana ni jedna kontrola. Ova opcija se preporučuje iskusnijim korisnicima ili kod kreiranja dijalog formi sa vrlo malo kontrola.

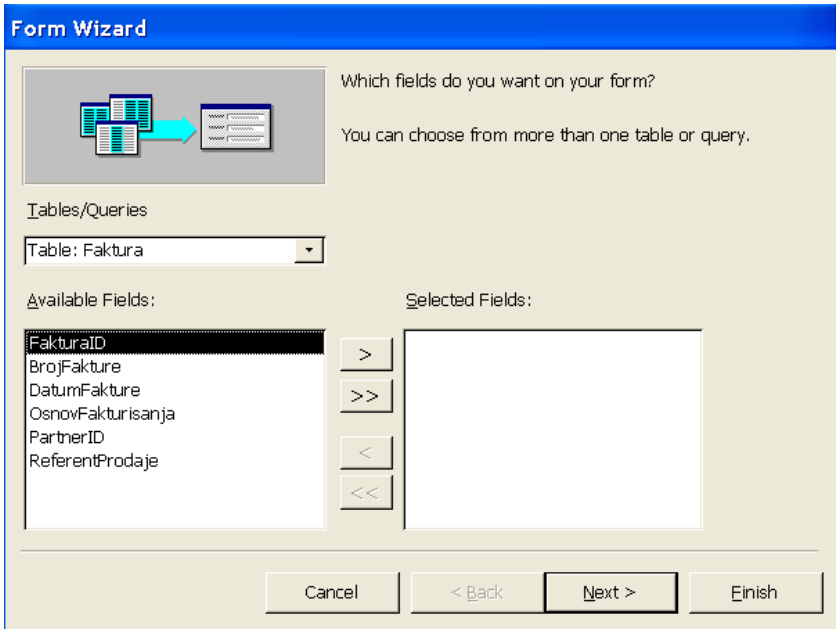
Form Wizard - "čarobnjak" koji omogućava postepeno kreiranje forme. U sledećem koraku vršite izbor polja, koja će se prikazivati na formi iz tabele izabrane u prvom koraku. Takođe, Access omogućava kreiranje forme koja se bazira na dve tabele, tzv. podforma (subform). Na taj način možete da odaberete i drugu tabelu na kojoj će se bazirati podforma. Zatim se bira tip forme:

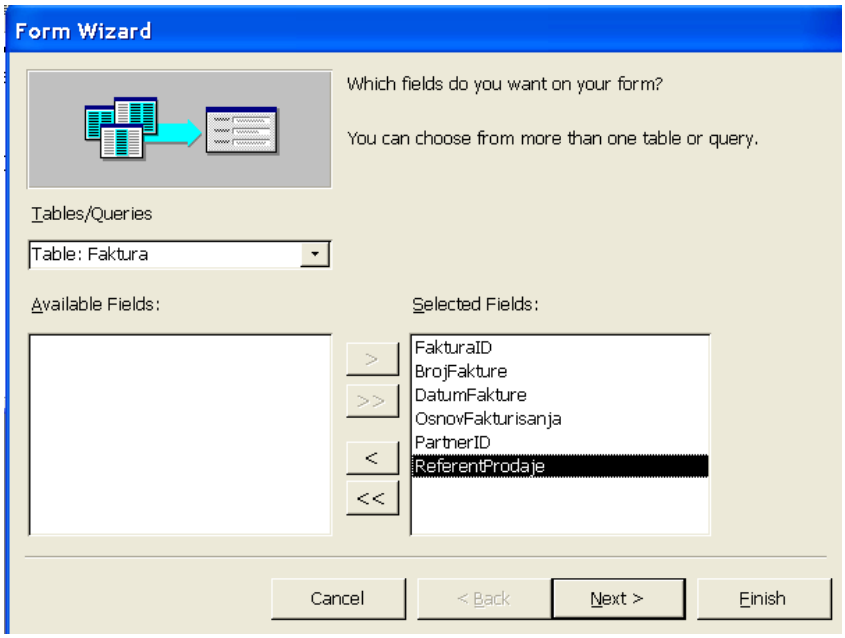
- Auto Form:Columnar - "čarobnjak" automatski pravi "kolonski" tip formi(polja na formi su raspoređena po kolonama).
- Auto Form:Tabular - "čarobnjak" automatski pravi "tabelarni" tip formi (prikaz više zapisa).
- Datasheet (tzv. grid prikaz), Justifed (raspoređuje polja ravnomerno po formi, maksimalno iskorišćavajući prostor).
- Chart Wizard - kreiranje formi koji pored klasičnih kontrola sadrži i grafikon. Grafikon je OLE objekat kreiran u aplikaciji Microsoft Graph, koja se isporučuje sa Access-om ili nekim programom iz Microsoft Office familije. Grafikon se kreira na osnovu podataka iz tabela ili upita koji se nalaze u vašoj aplikaciji.
- PivotTable Wizard - "čarobnjak" kreira formu za prikaz tabela iz Excel-a koje služe za unakrsno interaktivno izračunavanje.

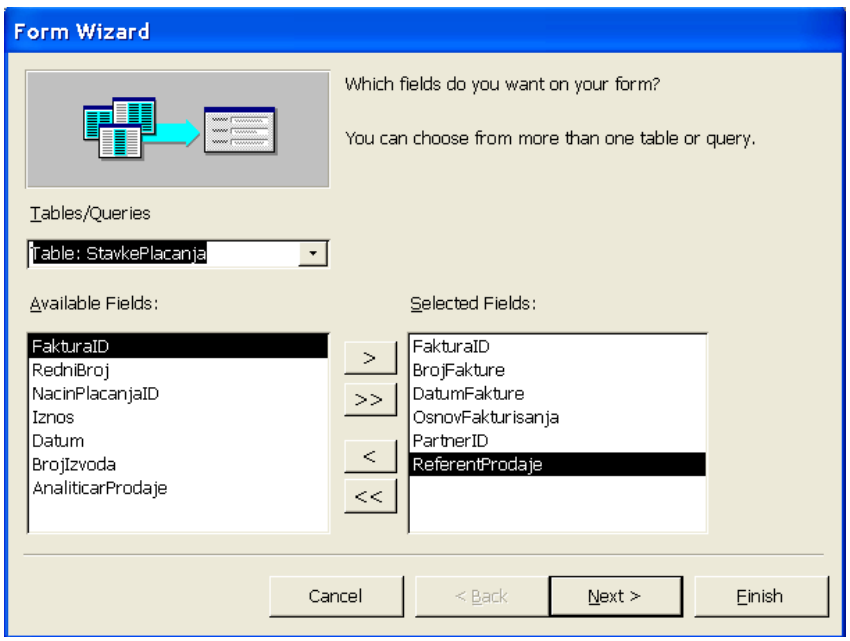
Kad izaberete tip forme, u narednom koraku birate stil forme (izgled pozadine, boju i tip fonta i slično). Na kraju, upisujete naziv forme i taj naziv "čarobnjak" koristi i za definisanje naslova forme. Pritisnite dugme Finish i Access je kreirao kostur forme. Na vama sada ostaje da kreirate

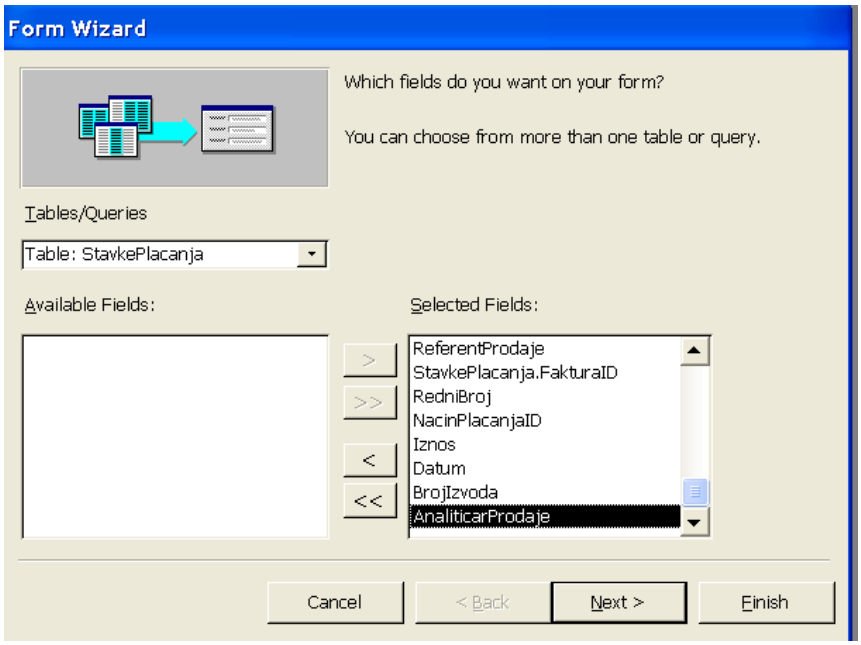
dodatne kontrolne funkcije.

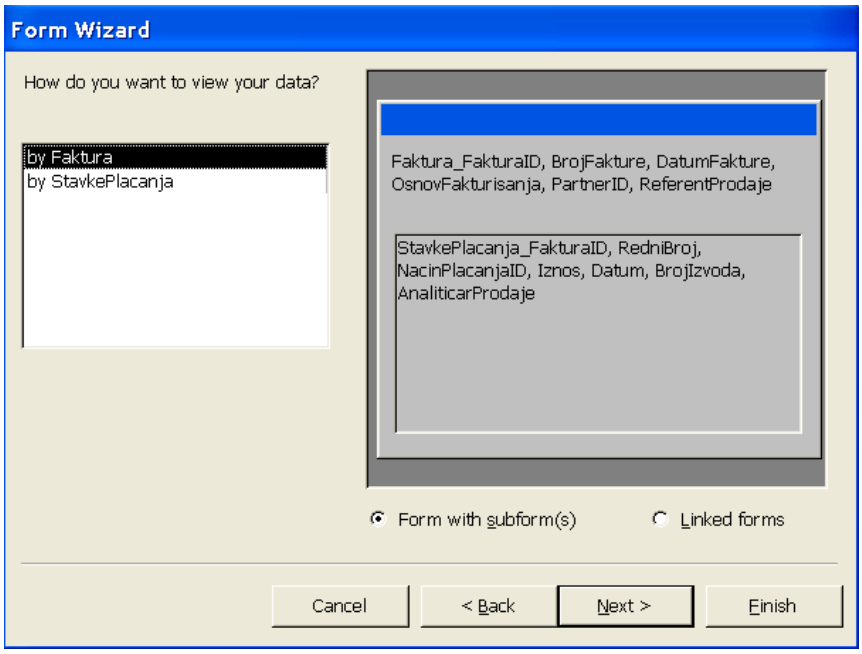


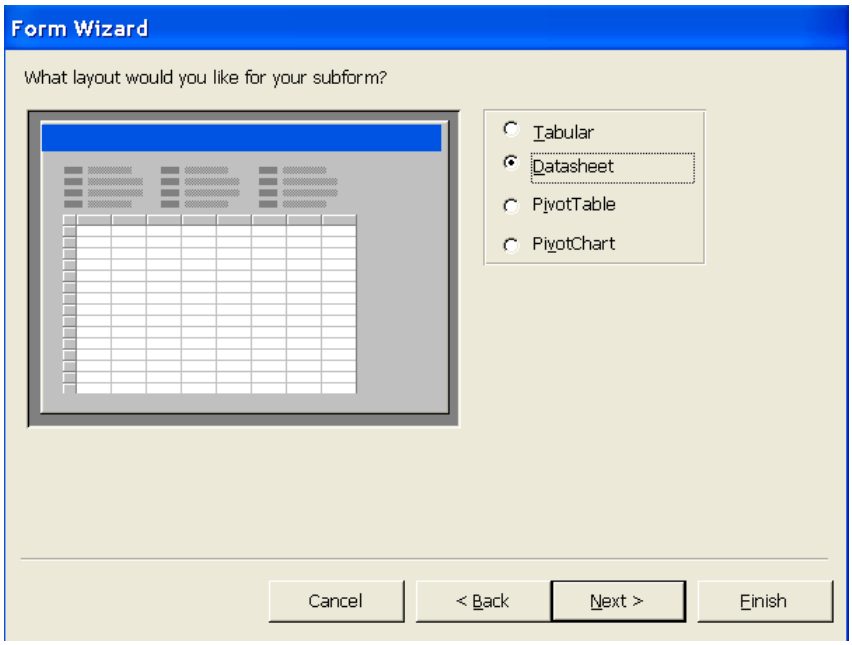


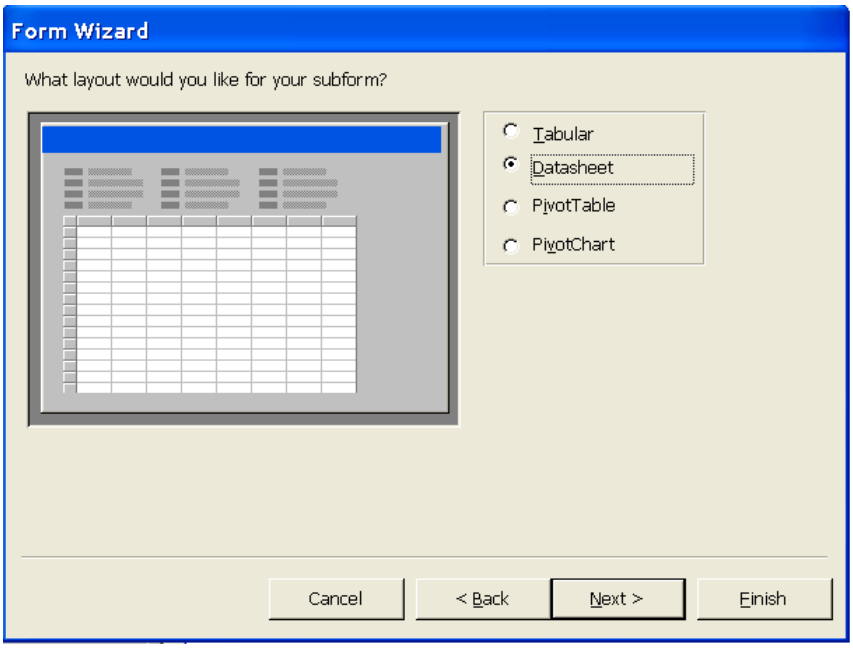

Form Wizard
Which fields do you want on your form?
You can choose from more than one table or query.
Tables/Queries
Table: Faktura
Available Fields:
Selected Fields:
FakturaID
BrojFakture
DatumFakture
OsnovFakturisanja
PartnerID
ReferentProdaje
>
>>
<
<<
Cancel
< Back
Next >
Finish

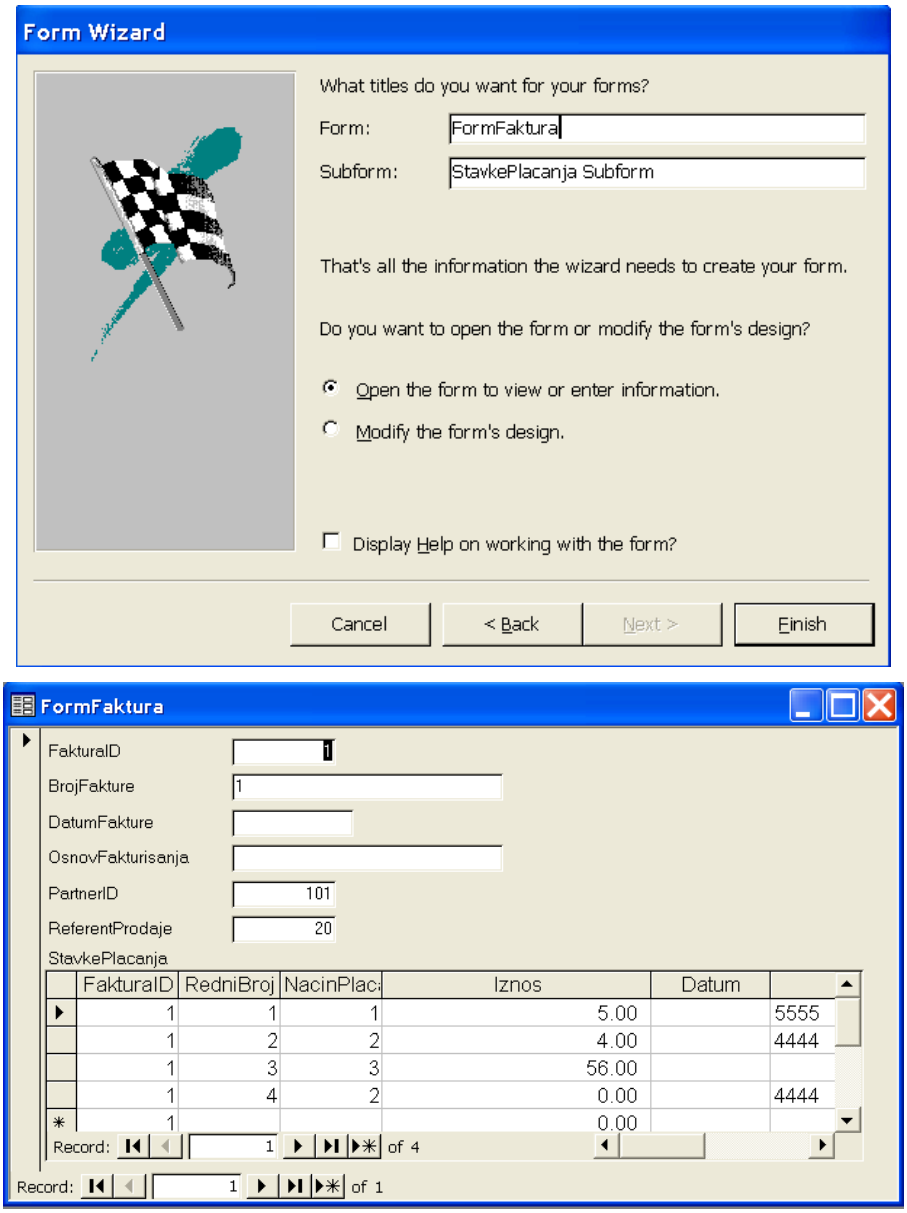

Form Wizard
Which fields do you want on your form?
You can choose from more than one table or query.
Tables/Queries
Table: StavkePlacanja
Available Fields:
Selected Fields:
FakturaID
RedniBroj
NacinPlacanjaID
Iznos
Datum
BrojIzvoda
AnaliticarProdaje
FakturaID
BrojFakture
DatumFakture
OsnovFakturisanja
PartnerID
ReferentProdaje
>
>>
<
<<
Cancel
< Back
Next >
Finish


Form Wizard
Which fields do you want on your form?
You can choose from more than one table or query.
Tables/Queries
Table: StavkePlacanja
Available Fields:
Selected Fields:
ReferentProdaje
StavkePlacanja.FakturaID
RedniBroj
NacinPlacanjaID
Iznos
Datum
BrojIzvoda
AnaliticarProdaje
>
>>
<
<<
Cancel
< Back
Next >
Finish


Form Wizard
How do you want to view your data?
by Faktura
by StavkePlacanja
Faktura_FakturaID, BrojFakture, DatumFakture, OsnovFakturisanja, PartnerID, ReferentProdaje
StavkePlacanja_FakturaID, RedniBroj, NacinPlacanjaID, Iznos, Datum, BrojIzvoda, AnaliticarProdaje
Form with subform(s)
Linked forms
Cancel
< Back
Next >
Finish


Form Wizard
What layout would you like for your subform?
Tabular
Datasheet
PivotTable
PivotChart
Cancel
< Back
Next >
Finish


Form Wizard
What layout would you like for your subform?
Tabular
Datasheet
PivotTable
PivotChart
Cancel
< Back
Next >
Finish



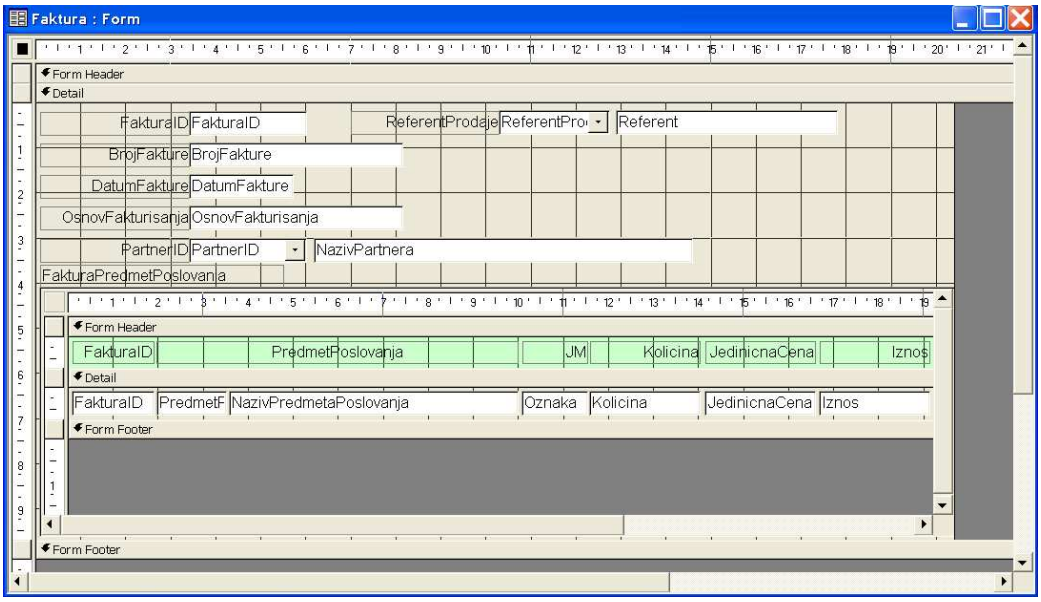
Slika 35 Prikaz generisanja forme koriscenjem carobnjaka

Dizajniranje forme

Obično forma koja je kreirana pomoću "čarobnjaka" ne zadovolja sve potrebe krajnjih korisnika, pa ju je potrebno još malo doraditi. Ako se nalazite u Database Window-u, izaberite formu koju želite da uređujete i pritisnite dugme Design View. Ako ste u tzv. Form View-u modu onda se u dizajn mod (slika 7) prelazi pritiskom na odgovarajuće dugme na toolbar-u ili izborom opcije Design View menija View. Dizajniranje forme u ovom modu rada je estetsko i funkcionalno doterivanje forme.

Forma je sastavljena od kontrola razmeštenih u definisanom prostoru za formu. Prostor za formu je podeljen u tri sekcije:

- Form Header - zaglavlje forme; obično se koristi za definisanje naslova forme,
- Detail - centralni deo forme u kome se nalazi većina kontrola na formi
- Form Footer - donji deo forme, slična zaglavlju forme; obično se koristi za prikaz zbirnih rezultata.



Slika 36 Dizajniranje forme

Već smo pomenuli da je forma objekat pa samim tim ona ima i određena svoja svojstva.

Slika 37 Prikaz svojstva forme

Svojstva forme su:

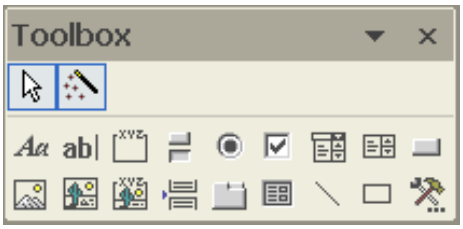
- Record Source - definisanje tabele ili upita nad kojom je definisana forma.
- Filter - postavljanje uslova po kome će se filtrirati podaci na formi (na primer, ŠIMEĆ = 'Ivan')
- Order by - definisanje polja po kojima će se vršiti sortiranje
- Allow Filters - omogućavanje korišćenja filtera nad formom
- Default View - definisanje osnovnog tipa prikaza; Single Form - prikaz jednog zapisa, Continuos Form - prikaz više zapisa odjednom, Datasheet View - grid prikaz.
- Allow Edits - dozvoljavanje ažuriranje podataka u formi
- Allow Deletions - dozvoljavanje brisanje zapisa
- Allow Additions - dozvoljavanje dodavanje novih zapisa
- Data Entry - ukoliko je postavljen na Yes prilikom otvaranja forme vrši se pozicionira na blanko zapis, u suprotnom pozicionira se na prvi zapis u tabeli nad kojom je definisana forma.
- RecordSet Type - postavljanje tipa RecordSet-a; Dynaset - moguće je ažuriranje svih kontrola, izuzev u slučajevima kada se vrši ažuriranje polja nad kojim postoji relaciona veza jedan-više. To polje nije moguće ažurirati u slučaju da nova vrednost polja ne postoji u tabeli "roditelja". Dynaset (Inconsistent Updates) - moguće ažuriranje svih kontrola bez ikakvih ograničenja. Snapshot - nije moguće ažuriranje ni jedne kontrole.
- Record Locks - zaključavanje zapisa prilikom ažuriranja zapisa ukoliko aplikaciju koristite pod mrežom; No Locks - ne zaključava zapis, All Records - zaključava sve zapise, Edited Record - zaključava samo onaj zapis koji se ažurira.
- Scroll Bars - definisanje tipa "pokretne traku" koja će se prikazivati.
- Record Selectors -prikaz trake za selektovanje celog zapisa.
- Navigation Buttons - prikaz dugmadi za kretanje po zapisima (nalazi se u donjem desnom uglu prozora forme).
- Dividing Lines - prikaz linija za razgraničavanje sekcija forme.
- Auto Resize - vrši automatsko podešavanje veličine prozora forme da se prikaže kompletan zapis (Yes).
- Auto Center - automatsko centriranje forme u odnosu na prozor aplikacije.
- Pop Up - (Yes) forma se otvara u Pop Up modu, uvek je prikazana na "vrhu" svih Access prozora bez obzira da li je fokusirana ili ne.

- Modal - (Yes) nije mogući preći na drugu formu ili bilo koji prozor u Access-u, a da se ne zatvori trenutna forma.
- Border Style - definisanje ivice i stila prozora (Title bar, Control menu, Minimize i Maximize buttons, Close button).
- Width - definisanje širine forme.
- Picture - ime datoteke koja je namenjena za pozadinu forme.
- Picture Type - Embeded - slika je ugrađena u aplikaciju, Linked - povezana sa originalnom slikom
- Picture Size Mode - izbor načina prilagođavanja veličine slike pozadine.
- Picture Aligment - izbor načina poravnavanja slike pozadine.
- Picture Tiling - (Yes) nad slikom pozadine se primenjuje "tiling" efekat
- Cycle - definisanje prelaska sa poslednje kotrole na formi. All Records - prelazi na sledeći zapis i pozicionira se na prvu kontrolu na formi, Current Record - prelazi na prvu kontrolu na formi ne prelazeći na sledeći zapis, Current Page - prelazi na prvu kontrolu na trenutnoj vidljivoj strani forme.
- Menu Bar - izbor korisničkog menija koji će se prikazivati kad je forma otvorena.
- Toolbar - izbor korisničkog toolbar-a koji će se prikazivati kad je forma otvorena.
- ShortCut Menu - određivanje da li će se prikazivati korisnički meni na desni klik miša.
- ShortCut Menu Bar - izbor korisničkog "shortcut" menija koji će se prikazivati kad je forma otvorena.
- GridX, GridY - definiše se horizontalno i vertikalnu rastojanje mreže za poravnavanje kontrola na formi u dizaj modu.
- LayoutPrint - (No) prilikom štampe koristi fontove sa forme.
- Fast Laser Printing - (Yes) zamenjuje linije i kvadrate sa odgovarajućim tekstualnim karakterima što mnogo ubrzava štampanje na laserskom štampaču.
- Help File - definisanje "puta" do help datoteke koju je korisnik kreirao za trenutnu formu.
- Help Contex Id - specifiranje ID teme u korisničkoj help datoteci.
- Palette Source - izbor palete boja.
- Tag - dodatne informacije o formi.
- Has Module - (Yes) definisanje da li nad formom postoji modul (programski kod). Ukoliko je

setovano na No perfomanse forme se znatno poboljšavaju.

Kontrole u formi

Na sledećoj slici prikazan je toolbar za kreiranje kontrola na formi. Prikaz ovog toolbar-a se omogućava izborom opcije Toolbox menija View ili pritiskom na dugme Toolbox na toolbar-u.



Slika 38 Prikaz Toolbox

Access nudi sledeće predefinisane kontrole:

- Label - statički tekst, namenjen pre svega za definisanje naslova obrasca i naziva drugih kontrola na formi. Promena vrednosti ove kontrole u run modu je jedino moguće uraditi programski (na primer, Label1.Caption = "Naziv proizvoda").
- TextBox - kontrola koja omogućava unos, ažuriranje i pregled podataka iz polja tabele ili upita nad kojim je definisana forma.
- Option Group - grupiše više kontrola tipa OptionButton, ToggleButton, CheckBox i sinhronizuje njihov rad. Sinhronizacija se odvija na taj način da je dozvoljeno da samo jedna kontrola bude "uključena". Svaka od kontrola koje su "grupisane" imaju svoju vrednost, a vrednost Option Group kontrole je identična vrednosti "uključene" kontrole. Prilikom kreiranja ove kontrole korisno je koristiti "čarobnjaka".
- OptionButton, ToggleButton, CheckBox - su kontrole koje poseduju dva stanja (Yes/No) i obično se koriste nad poljima Boolean tipa. Razlike između njih su vizuelne prirode.
- Combo Box - kontrola slična Text box-u. Ona omogućava izbor elemenata liste koja je napravljena na osnovu vrednosti nekih polja iz tabele ili upita. Pored toga, elemente liste je moguće ručno uneti. Prilikom kreiranja ove kontrole korisno je koristiti "čarobnjaka".
- List Box - po unutrašnjoj strukturi i nameni slična je Combo Box kontroli, ali mnogo više zauzima prostora na formi. Koristi se uglavnom kod izbora više elemenata odjednom. Prilikom kreiranja ove kontrole korisno je koristiti "čarobnjaka".
- CommandButton - je tip kontrole za koju je vezana funkcija napisana u Visual Basic-u ili makrou. Access nudi veliki izbor korisnih varijanti ove kontrole, pa se pri kreiranju ove kontrole preporučuje korišćenje njegovog "čarobnjaka".
- Image - omogućava unošenje slike koja služi kao pozadina na formi.

- ObjectFrame - sadrži crtež, grafički prikaz ili neki drugi OLE objekat koji nije u okviru baze podataka već se nalazi van Access aplikacije.
- BoundObjectFrame - sadrži crtež, grafički prikaz ili neki drugi OLE objekat koji se nalazi u okviru baze podataka.
- PageBreak - naglašava deljenje centralne sekcije forme na stranice.
- Subform - je kontrola koja u sebi sadrži vezu sa drugim formama prikazanim u definisanom okviru ove kontrole. Veza između polja glavne forme (forma koju trenutno kreiramo) i podforme ostvaruje se preko veznih polja sa glavne forme i veznih polja sa podforme. Ova kontrola se uglavnom koristi za prikazivanje većeg broja zapisa neke tabele koji odgovaraju sadržaju izbornog sloga u glavnoj tabeli.
- Line - kontrola za prikaz linije na formi.
- Rectangle - kontrola za prikaz pravougaonika na formi.

Pored ovih predefinisanih kontrola moguće je dodavanje korisničkih kontrola.

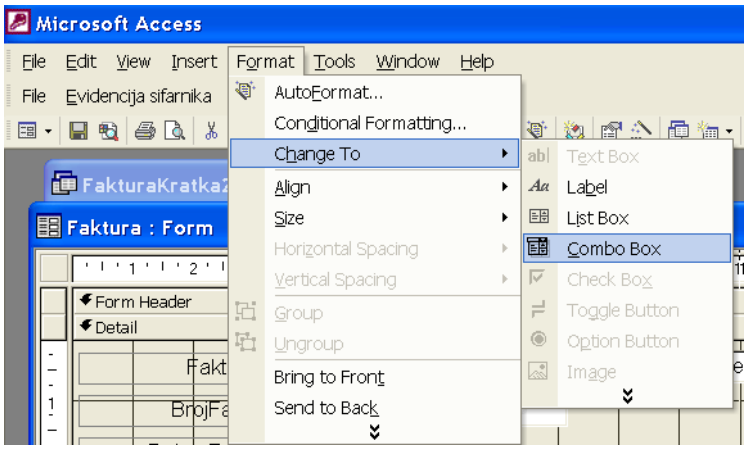
Sve ove kontrole imaju svoja svojstva koja se uglavnom poklapaju sa svojstvima forme, ali postoji izvesnih razlika.

Redosled kretanja kroz kontrole se automatski podešava opcijom TabsOrder menija Edit. Takođe, ovaj redosled je moguće "ručno" podesiti izmenom svojstva kontrole TabIndex.

Definisanje Combo Box-a

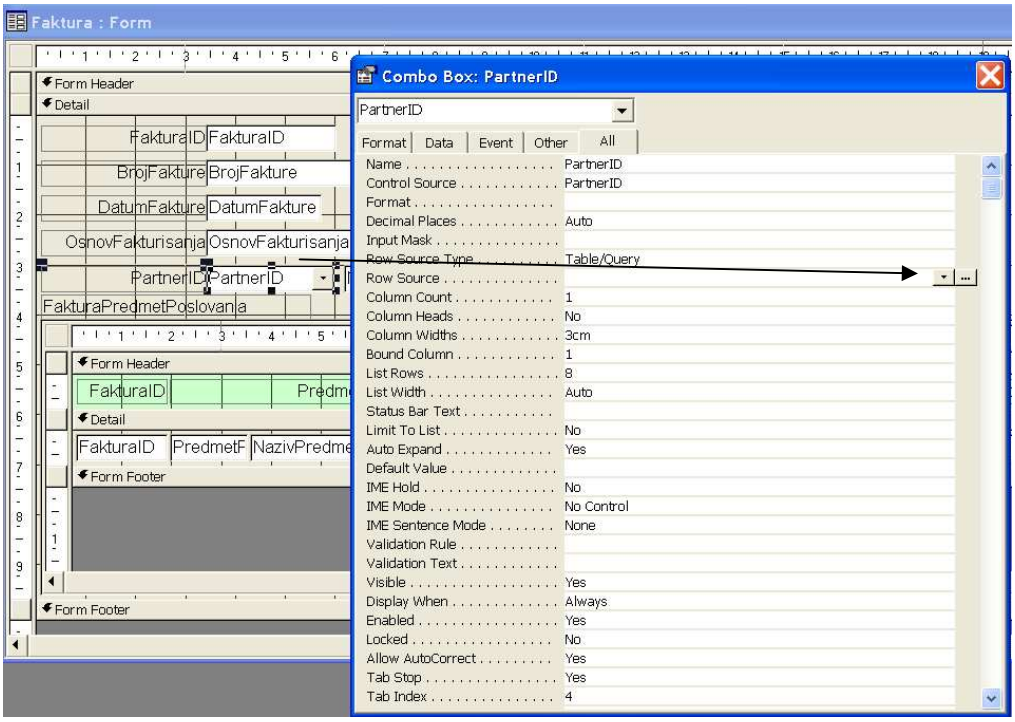
Na sledećim slikama prikazaće se postupak formiranja Combo Box-a.

1. Korak Definisanje Combo Box-a kao sto je prikazano na sledećoj slici



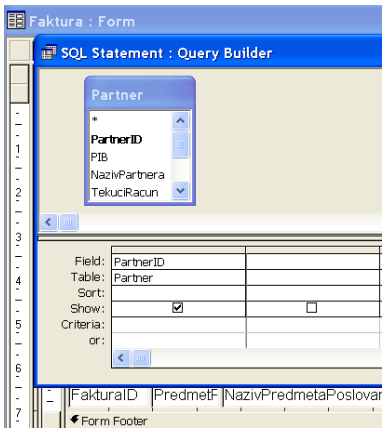
Slika 39 Prikaz definisanja Combo Box-a

2 Korak Selektovanje Combo Boxa-a i definisanje osobina u okviru RowSorce.



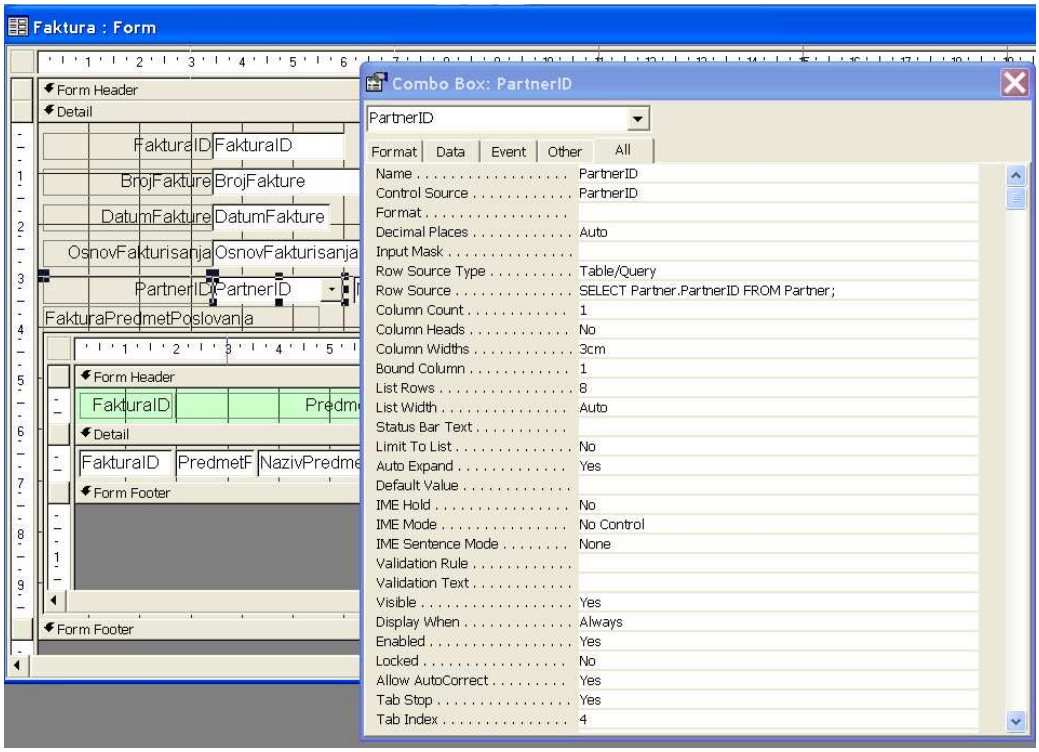
Slika 40 Selektovanje Combo Boxa-a i definisanje osobina u okviru RowSorce

3 Korak Pritiskom na tri tackice iz predhodne slike dobija se Query Builder izabere se kolona koja treba da bude u Combo Boxu.



Slika 41 Izabor kolone koja treba da bude u Combo Boxu

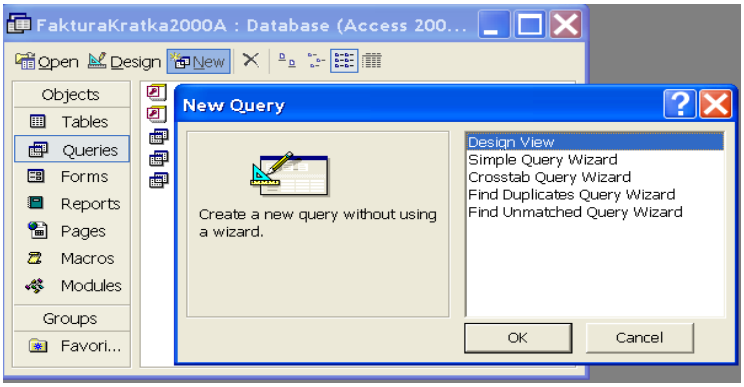
4. Korak Vidi se definisana select naredba

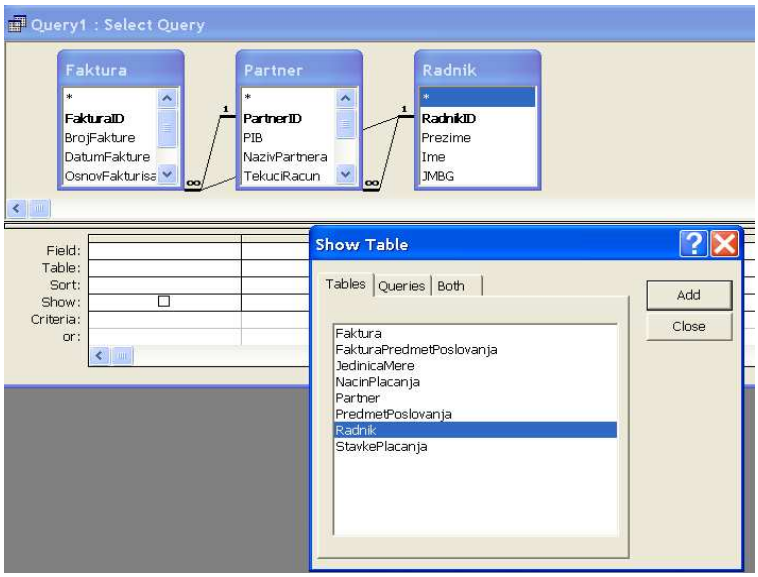


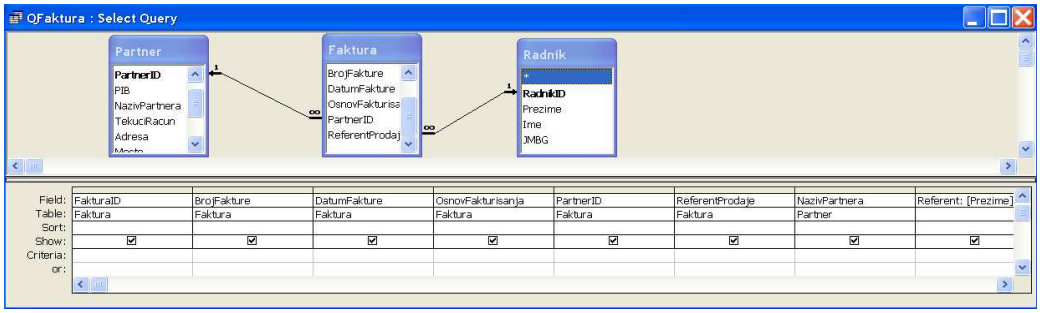
Slika 42 Prikaz Select naredbe

Definisanje Text Box-a

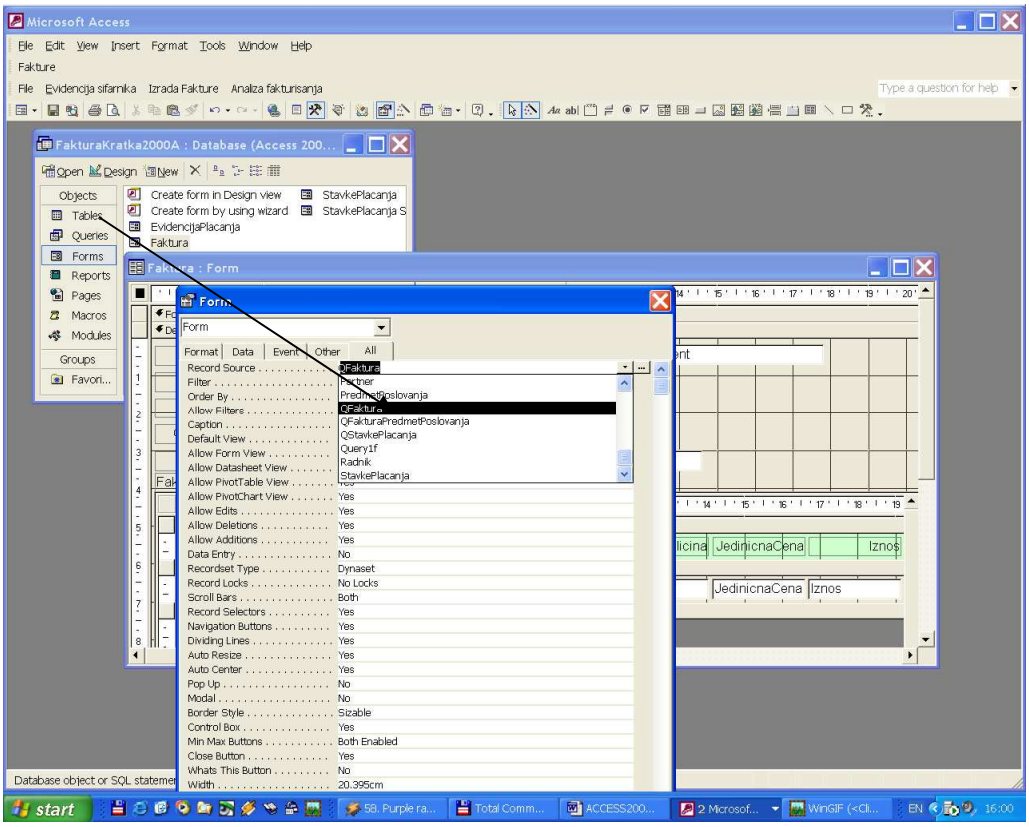
1. Korak je definisanje upita





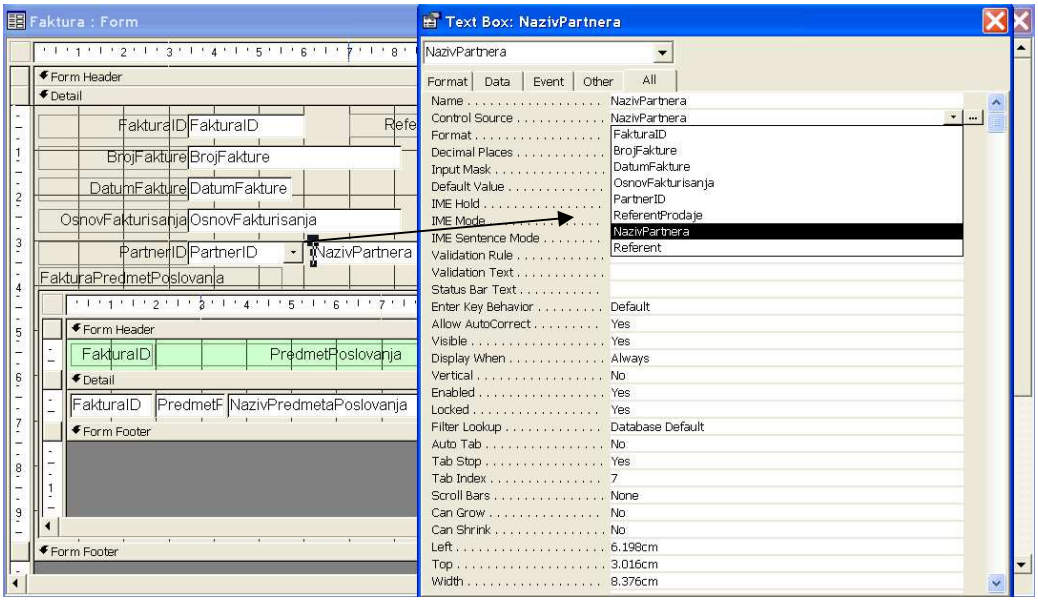


2. Korak je povezivanje upita i forme



Slika 43 povezivanje upita i forme

3. Korak Izbor opcije Tex Box i definisaće se Control Source u upitu.

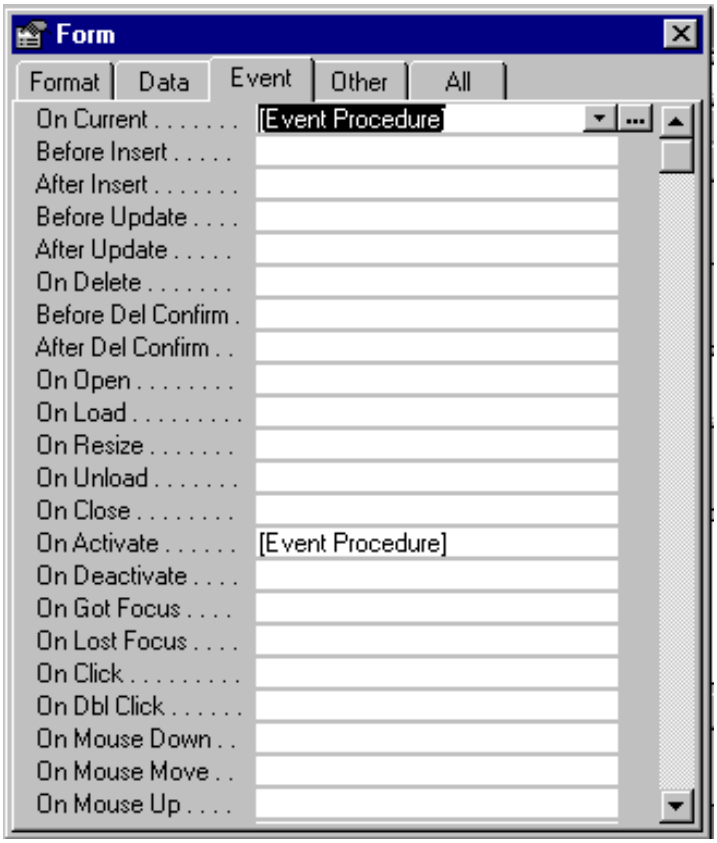


Slika 44 Izbor opcije Tex Box

Event model

Dosad nisu objašnjena sva svojstva. U Access-u ima nekoliko predefinisanih svojstava koja prate dešavanja u radu aplikacije i akcije samog korisnika i nazivaju se događaji (events). Princip rada sa događajima je da vi vezujete neku akciju koja treba da se izvrši kada se izabrani događaj desi. Akcije se obično rade u Access Visual Basic-u, mada ih je moguće uraditi i korišćenjem makroa. Pored VBA koda i makroa, akcije je moguće napisati korišćenjem izraza i ugrađenih funkcija.

Prlikom rada sa događajima veoma je važno poznavati sve događaje u Access-u, a i njihovo značenje.



Slika 45 Prikaz Event modela

U narednom delu prikazan je spisak događaja na formama, kao i objašnjenja kada se dešavaju:

- Current - kada se pomerite na sledeći zapis,
- BeforeInsert - kad se izvrše prve promene na novom zapisu ,
- AfterInsert - pošto se promene na novom zapisu snime,
- BeforeUpdate - neposredno pre nego što se izvršene izmene na zapisu snime,
- AfterUpdate - neposredno posle snimanja izvršenih izmena na zapisu snime,
- Delete - neposredno prije brisanja zapisa; pri brisanju više selektovanih zapisa, ovaj događaj se dešava za svaki selektovani zapis,
- BeforeDelConfirm - pošto se zapis izbriše, ali pre pojavljivanja sistemskog dijaloga za potvrdu brisanja,

- AfterDelConfirm - pošto je zapis obrisan ili po otkazivanju brisanja,
- Open - kada se forma otvori, ali kontrole i podaci nisu dostupne,
- Load - pošto se forma "napuni" kontroloma i RecordSource otvori,
- Resize - prilikom promene veličine prozora, dešava se i prilikom otvaranja forme,
- Unload - kada se na formi inicira zatvaranje, ali pre nego se forma stvarno zatvori,
- Close - kada se forma zatvori (u momentu iščezavanja),
- Activate - kada forma prima fokus,
- Deactivate - kada forma gubi fokus,
- GotFocus - pošto forma dobije fokus (dešava se samo kada forma ne sadržava kontrole koje mogu da prime fokus),
- LostFocus - pošto forma izgubi fokus (dešava se samo kada forma ne sadržava kontrole koje mogu da prime fokus),
- Click - kada na formi pritisnite Record Selector ili na prostor u formi koji se pojavljuje kada je forma veća od veličine njenih sekcija,
- DblClick - kada na formi dva puta pritisnite Record Selector ili na prostor u formi koji se pojavljuje kada je forma veća od veličine njenih sekcija,
- MouseDown - kada pritisnite bilo koji taster na mišu na Record Selector ili na prostor u formi koji se pojavljuje kada je forma veća od veličine njenih sekcija,
- MouseMove - kada pomerate kursor miša preko Record Selector ili preko prostora u formi koji se pojavljuje kada je forma veća od veličine njenih sekcija,
- MouseUp - kada otpustite bilo koji taster na mišu na Record Selector ili na prostor u formi koji se pojavljuje kada je forma veća od veličine njenih sekcija,
- KeyDown - kada pritisnite taster na tastaturi bilo gde na formi, pod uslovom da je KeyPreview postavljen na True,
- KeyUp - kada otpustite taster na tastaturi bilo gde na formi, pod uslovom da je KeyPreview postavljen na True,
- KeyPress - kada pritisnite i otpustite ANSI taster na tastaturi bilo gde na formi, pod uslovom da je KeyPreview postavljen na True, ukoliko nije dešava se jedino kada je selektovan RecordSelector,
- Error - kada se nad formom dogodi run-time greška, kao što su greške validacije i greške sa tipovima podataka,

- Filter - kada izaberete opciju za unošenje filtera,
- ApplyFilter - kada dodelite filter,
- Timer - kada vremenski interval postavljen u obeležju TimerInterval istekne.

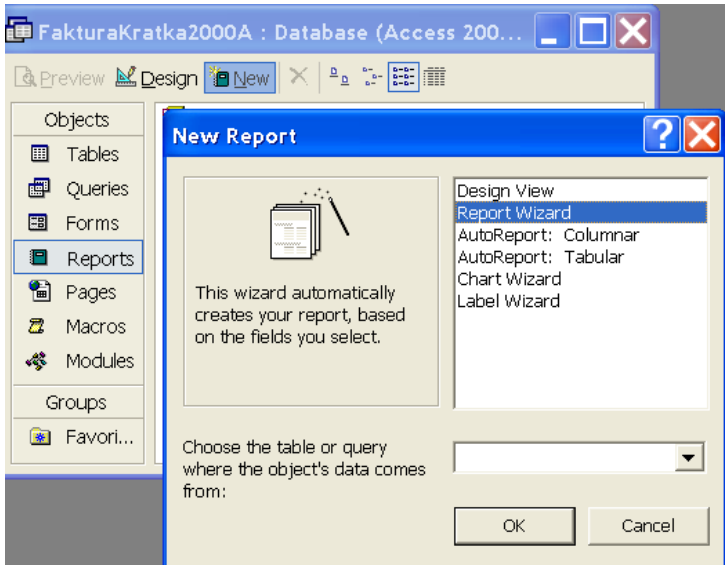
Izveštaji

Kreiranje izveštaja je pogodan način da prikažete vaše podatke kao štampani dokument. Zapitaćete se zašto sad ovo kad je moguće štampanje formi i podataka direktno iz upita i tabela. Izveštaji vam omogućavaju više kontrole da efektno prikažete podatke i mnogo veću fleksibilnost kod izračunavanja i prikazivanja zbirnih pregleda.

Izveštaji su praktično izvučene informacije iz vaše baze koje su organizovane i formatirane tako da zadovolje vaše zahteve, a ujedno ih možete upotrebiti i distribuirati. Primeri nekih izveštaja su nalepnice sa adresama, narudžbine, telefonski imenik itd.

Kreiranje izveštaja

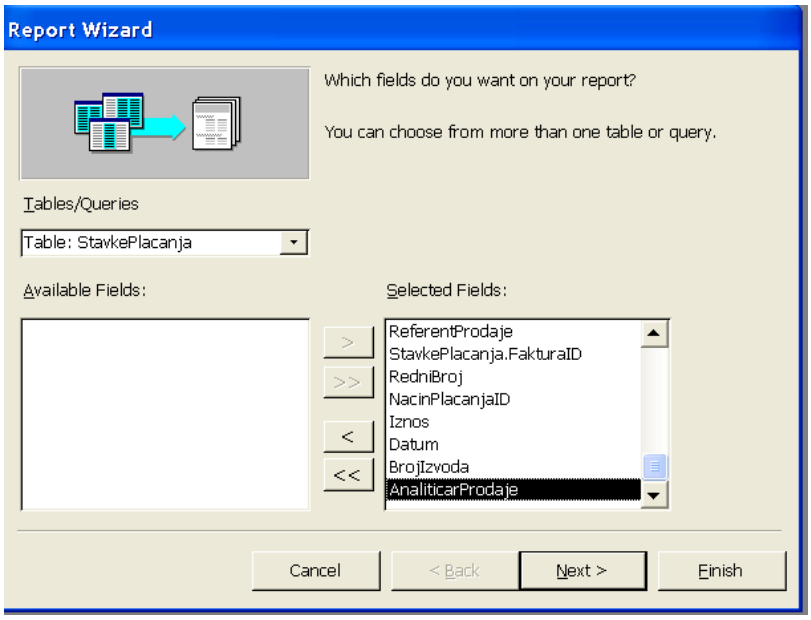
Postupak pri izradi novog izveštaja je sličan kao i kod izrade ostalih Access-ovih objekata. Prvo je neophodno odabrati u Database Window-u karticu Reports, pritisnuti dugme New nakon čega će se otvoriti dijalog za izbor načina kreiranja izveštaja.

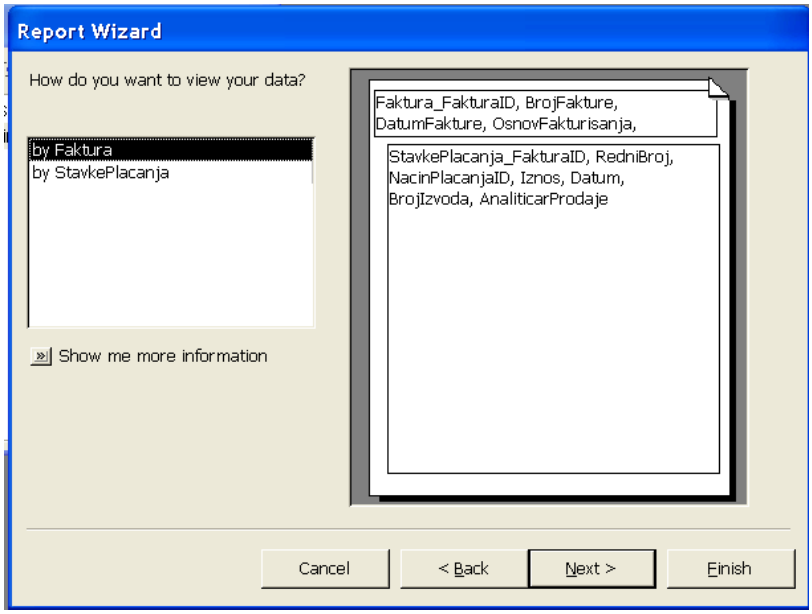


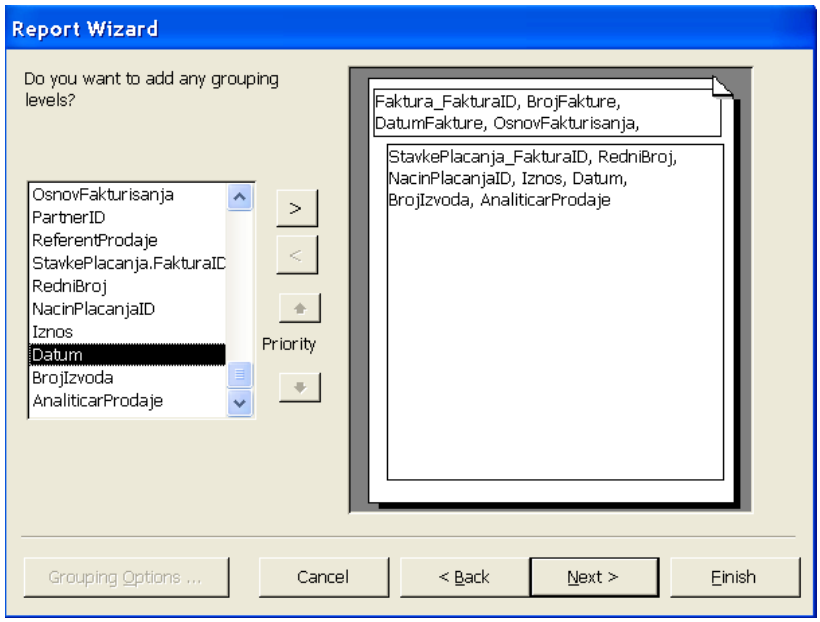
Slika 46 Prikaz kreiranja izvestaja

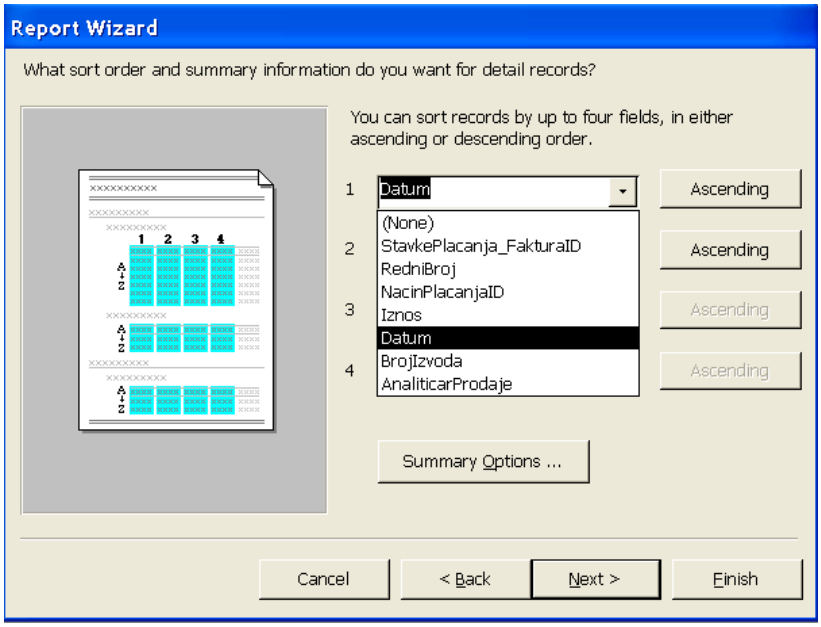
Ponuđeni načini kreiranja su:

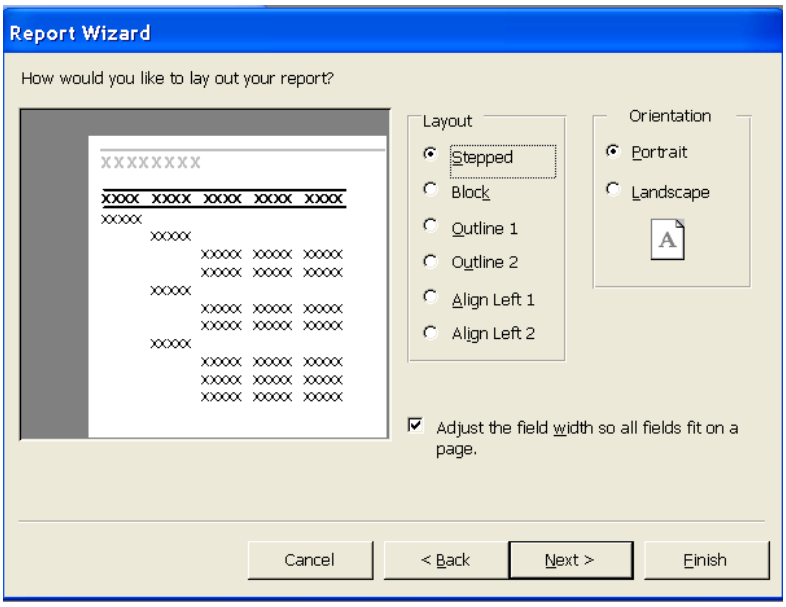
- Design View - opcija za kreiranje blanko izveštaja. Preporučuje se iskusnim korisnicima Access-a.
- Report Wizard - "čarobnjak" koju služi za kreiranje izveštaja. U prvom koraku odaberete naziv tabele ili upita na osnovu koju ćete da izradite izveštaj. Zatim, u drugom koraku, vršite izbor polja iz tabele ili upita koje želite da se prikažu na vašem izveštaju. U trećem koraku definišete polja po kojim želite da izvršite grupisanje vaših podataka. U donjem uglu prozora nalazi se dugme Grouping Options pomoću koga se otvara prozor za definisanje načina grupisanja. U sledećem koraku od preostalih polja (od polja koji nisu uključeni u grupisanju) se vrši izbor polja po kojima će se vršiti sortiranje. U Summary Options možete izabrati polja nad kojima će se vršiti sumiranje (po grupama ili ukupno), izračunavanje srednje vrednosti, minimuma i maksimuma. U narednim koracima izabirate način prikazivanja podataka, orjentaciju papira, stil izveštaja (izbor fonta, veličine fonta, boja slova, boja pozadine, itd.) i naziv izveštaja. Na kraju pritisnite dugme Finish i kostur izveštaja je gotov. Dorada ovako kreiranih izveštaja se vrši u Design View-u.
- AutoReport: Columnar - opcija za automatsko kreiranje izveštaja "kolonskog" tipa.
- AutoReport: Tabular - opcija za automatsko kreiranje "tabelarnih" izveštaja.
- Chart Wizard - opcija za kreiranje izveštaja koji su bazirani na dijagramima. Pri kreiranju ovog tipa izveštaja u prvom koraku se vrši izbor tabele ili upita i polja. U drugom koraku se vrši izbor tipa dijagrama, a zatim izbor polja po kojima će se vršiti grupisanje i sumiranje. Pri tome možete da vidite i izgled vašeg dijagrama na osnovu vrednosti iz vaših polja. Na kraju se definiše naziv izveštaja.
- Label Wizard - opcija koja omogućava kreiranje izveštaja za izradu nalepnica za pisma. U prvom koraku vršite izbor dimenzija nalepnica. Ukoliko ponuđene dimenzije ne zadovoljavaju vaše potrebe, opcija Custom vam omogućava da u editoru kreirate dimenzije koje će zadovoljiti vaše potrebe. Nakon definisanja dimenzija nalepnice, u narednim koracima birate fontove, polja koja će se prikazivati, polje po kome će se vršiti sortiranje i naziva izveštaja.

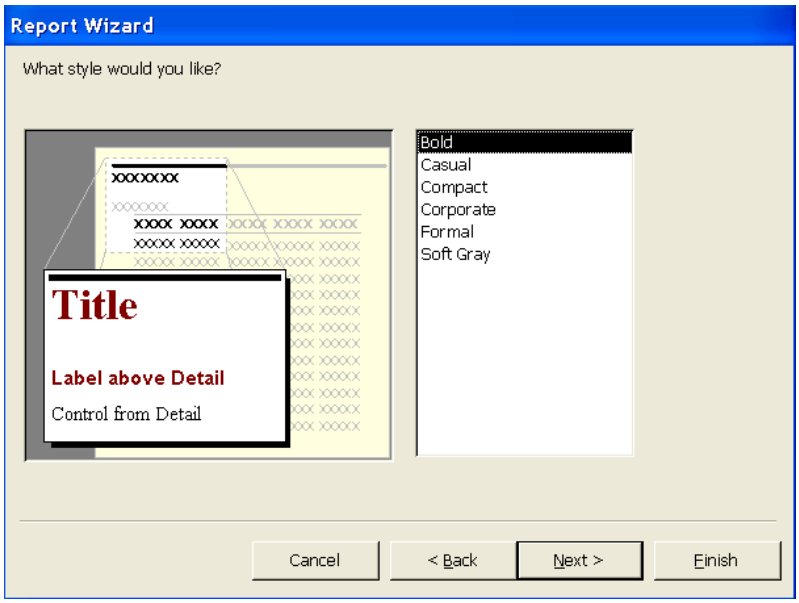

Report Wizard
Which fields do you want on your report?
You can choose from more than one table or query.
Tables/Queries
Table: StavkePlacanja
Available Fields:
Selected Fields:
ReferentProdaje
StavkePlacanja.FakturaID
RedniBroj
NacinPlacanjaID
Iznos
Datum
BrojIzvoda
AnaliticarProdaje
>
>>
<
<<
Cancel
< Back
Next >
Finish

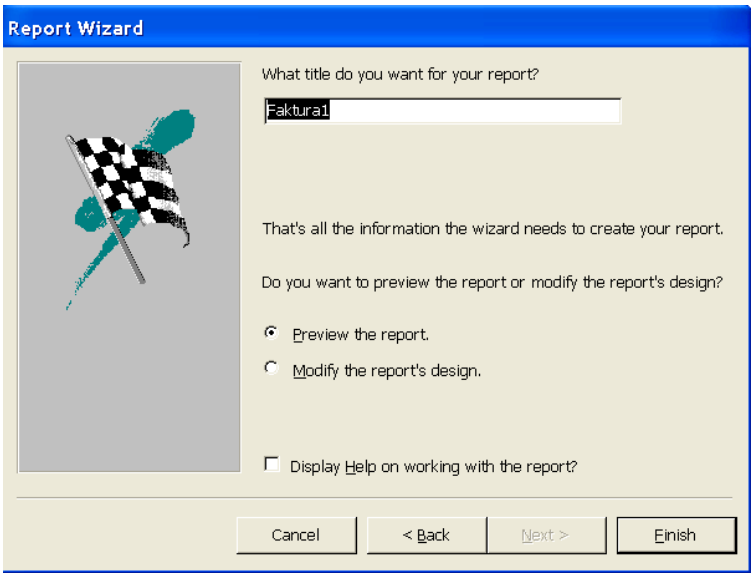

Report Wizard
How do you want to view your data?
by Faktura
by StavkePlacanja
Show me more information
Faktura_FakturaID, BrojFakture,
DatumFakture, OsnovFakturisanja,
StavkePlacanja_FakturaID, RedniBroj,
NacinPlacanjaID, Iznos, Datum,
BrojIzvoda, AnaliticarProdaje
Cancel
< Back
Next >
Finish


Report Wizard
Do you want to add any grouping levels?
OsnovFakturisanja
PartnerID
ReferentProdaje
StavkePlacanja.FakturaID
RedniBroj
NacinPlacanjaID
Iznos
Datum
BrojIzvoda
AnaliticarProdaje
>
<
Priority
Faktura_FakturaID, BrojFakture, DatumFakture, OsnovFakturisanja,
StavkePlacanja_FakturaID, RedniBroj, NacinPlacanjaID, Iznos, Datum, BrojIzvoda, AnaliticarProdaje
Grouping Options ...
Cancel
< Back
Next >
Finish


Report Wizard
What sort order and summary information do you want for detail records?
You can sort records by up to four fields, in either ascending or descending order.
1
Datum
(None)
StavkePlacanja_FakturaID
RedniBroj
NacinPlacanjaID
Iznos
Datum
BrojIzvoda
AnaliticarProdaje
2
3
4
Ascending
Summary Options ...
Cancel
< Back
Next >
Finish

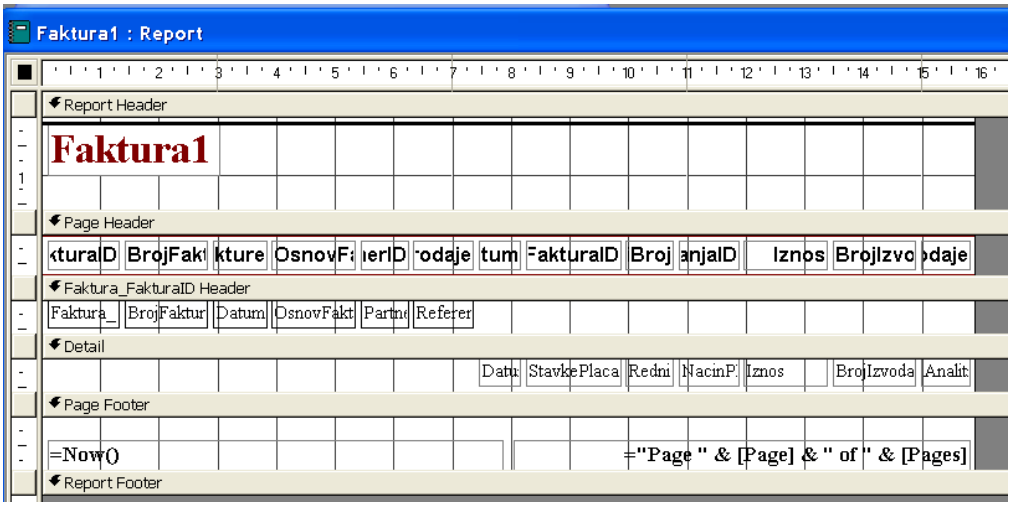

Report Wizard
How would you like to lay out your report?
Layout
Stepped
Block
Outline 1
Outline 2
Align Left 1
Align Left 2
Orientation
Portrait
Landscape
Adjust the field width so all fields fit on a page.
Cancel
< Back
Next >
Finish


Report Wizard
What style would you like?
Bold
Casual
Compact
Corporate
Formal
Soft Gray
Title
Label above Detail
Control from Detail
Cancel
< Back
Next >
Finish



Slika 47 Prikaz koriscenja carobnjaka za izvestaje

Izgled primera izveštaja u Design View-u prikazan je na sledecoj slici.



Slika 48 Izgled primera izveštaja u Design View-u

Podešavanje stranice izveštaja i štampanje

Poslednja faza dizajniranja izveštaja je izbor štampača i podešavanje parametara stranice za štampu. Aktiviranjem opcije Page Setup menija File otvara se dijalog za podešavanje stranice. Dijalog je podeljen na tri dela: Margins, Page i Columns.

U Margins delu podešavaju se margine izveštaja (Top-gornja, Bottom-donja, Right-desna, Left-leva). Opcija Print Data Only, ukoliko je izabrana, u izveštaju štampa samo podatke, bez dodatnih kontrola koje su uključene na izveštaju.

Page kartica je namenjena za podešavanje orjentacije izveštaja (Landscape i Portrait), veličine stranice izveštaja (A4, A3, Letter,...), način prihvata papira pri štampi i izbor štampača.

Column kartica služi za kontrolu veličine kolona pri štampanju višekolonskih izveštaja i labela. Opcija Grid Settings podešava broj kolona i razmak između redova. Opcijom Layout Items bira se redosled po kome će se štampati labele.

Štampanje izveštaja se vrši izborom opcije Print menija File ili pritiskom na odgovarajuće dugme na toolbar-u.

Makroi

Reč makro je u vrlo raširena i uglavnom u računarstvu označava objedinjavanje više akcija, naredbi. Makroi su sastavljeni od više linija dogovorene konstrukcije sa pozivima predefinisanih procedura u Access-u. Oni se uglavnom koriste prilikom aktiviranja nekog objekta ili događaja na objektu. Makroi su pogodni za brzo pravljenje prototipskih aplikacija, ali njihovo korišćenje u profesionalnim aplikacijama se ne preporučuje iz više razloga:

- ne mogu se "povratiti" posle grešaka
- ne mogu se koristiti za povezivanje sa DLL-ovima
- otežano je izvođenje petlji
- ne mogu se koristi u radu sa recordset-ovima
- ne mogu se koristiti pri transakcijama
- nemaju mogućnost prosleđivanja parametara
- otežano je korišćenje debuger-a
- moguće je prekinuti njihovo izvršavanje sa Ctrl+Break (i ovo nije moguće ukinuti).

Kreiranje makroa

U Database window-u izaberite karticu Macro, pritisnite dugme New i otvoriće se radni prozor za kreiranje makroa. Radni prostor je sličan kao i kod definisanja atributa (kolona) u tabelama. Gornja polovina prozora je namenjena za izbor akcija, a donja polovina za definisanje argumenata akcija. Tabela za odabir akcija u makrou sastoji se od dve kolone. Prva kolona je combo box u kome su ponuđene sve akcije koje su podržane u makroima, a u drugoj koloni je vaš opis akcije. U sklopu jednog makroa moguće je definisanje podmakroa i uslova pod kojim se izvršava neka akcija. Izborom opcije Macro Names menija View ili klikom na odgovarajuće dugme na toolbar-u prikazuje se kolona za definisanje naziva podmakroa. Pozivanje podmakroa se vrši na sledeći način macroname.submacroname. Prikaz kolone za definisanje uslova za izvršavanje akcija se vrši izborom opcije Conditions menija View ili klikom na odgovarajuće dugme na toolbar-u.

U Access-u su podržane sledeće akcije:

- AddMenu - kreira korisnički napravljen meni na osnovnom meniju forme ili izveštaja,
- ApplyFilter - koristi filter, upit ili SQL Where izraz za filtriranje podataka u tabeli, formi ili izveštaju,
- Beep - zvučni signal,
- CancelEvent - otkazuje događaj iz kojeg je pozvan makro,
- Close - zatvara prozor koji se navede u argumentima, a ako se argument ne navede, zatvara se trenutno aktivan prozor,
- CopyObject - kopira objekat iz trenutne baze u drugu bazu,
- DeleteObject - briše objekat iz baze koji je naveden u argumentu ili, ako nije navedeno ništa u argumentu, briše se aktivni objekat,
- DoMenuItem - izvršava komandu iz Access menija,
- Echo - skriva ili prikazuje rezultate makroa dok se izvršava,
- FindRecord - pronalazi prvi zapis koji zadovoljava kriterijum definisan u argumentu,
- FindNext - pronalazi sledeći zapis koji zadovoljava kriterijum definisan u argumentu,
- GoToControl - pomera fokus na kontrolu definisanu u argumentima,
- GoToPage - pomera fokus na prvu kontrolu na formi na strani nevedenoj u argumentima,
- GoToRecord - izbor zapisa u zavisnosti definisanog argumenta,
- Hourglass - menja klasičan pointer miša u peščani sat,

- Maximize - maksimizira aktivni prozor,
- Minimize - minimizira aktivni prozor,
- MoveSize - menja veličinu aktivnog prozora na veličinu navedenu u argumentima,
- MsgBox - prikaz dijalog prozora za ispis poruka; sadržaj, tip i ime ovog dijalog prozora se podešavaju u argumentima,
- OpenForm - otvara formu čiji je naziv definisan u argumentu,
- OpenModule - otvara modul u dizajn modu i na specifiranoj proceduri,
- OpenQuery - otvara upit čiji je naziv definisan u argumentu,
- OpenReport - otvara izveštaj čiji je naziv definisan u argumentu,
- OpenTable - otvara tabelu čiji je naziv definisan u argumentu,
- OutputTo - izvoz podataka iz tabele ili upita u jedan od sledećih formata: .XLS (Excel), .RTF (Rich Text Format), .TXT (DOS tekst),
- PrintOut - štampa aktivni objekat iz baze,
- Quit - napuštanje Access-a; u argumentima se definiše da li se čuvaju promene izvršene nad objektima,
- Rename - menja ime objekta navedenom u argumentima ili aktivnog objekta u ime navedeno u argumentima,
- RepaintObject - osvežava ekran,
- Requery - ponovo izvršava upit definisan nad aktivnim objektom ili kontrolom,
- Restore - vraća minimizovani ili maksimizovan prozor na prethodnu veličinu,
- RunApp - pokreće aplikaciju definisanu u argumentima,
- RunCode - izvršava Visual Basic proceduru,
- RunMacro - izvršava drugi makro,
- RunSQL - izvršava SQL izraz koji je naveden u argumentima,
- Save - čuva sve izvršene promene nad navedenim objektom ili trenutno aktivnim objektom,
- SelectObject - selektuje izabrani objekat u bazi,
- SendKeys - šalje kodove kombinacije tastera Access-u ili drugoj aplikaciji,
- Send Objects- šalje specifirani objekat uz e-mail poruku,

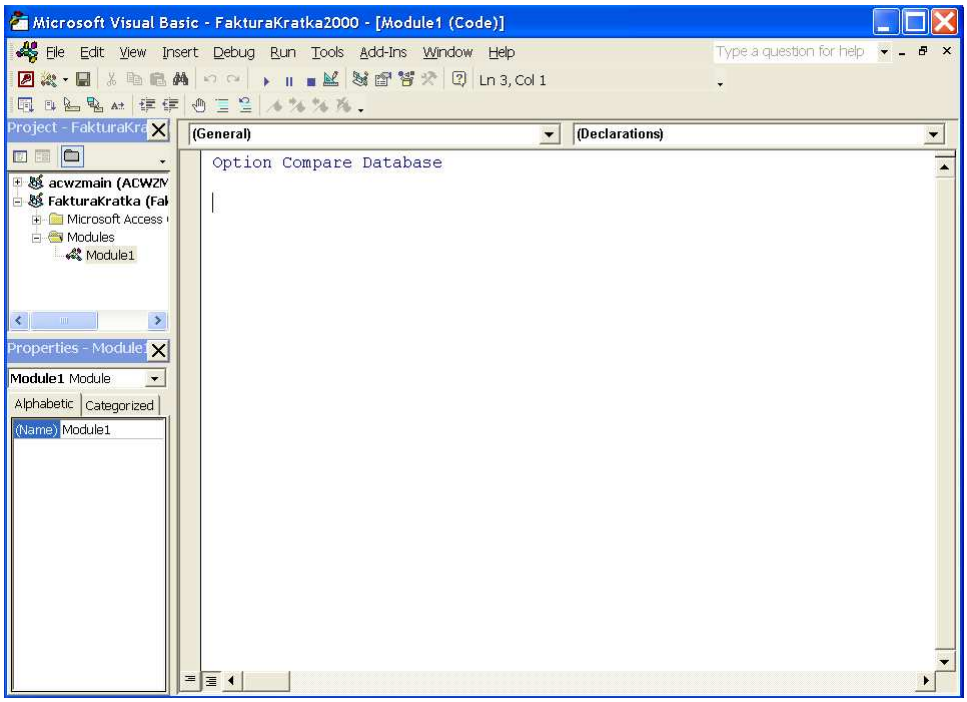
- SetMenuItem - definiše opcije osnovnog menija,
- SetValue - postavljanje vrednosti kontroli, polju ili obeležju na formi,
- SetWarnings - uključuje ili isključuje prikaz sistemskih poruka,
- ShowAllRecords - uklanja filter sa aktivne tabele, upita ili forme i prikazuje sve zapise,
- ShowToolbar - uključuje ili isključuje prikaz toolbar-a navedenog u argumentu,
- StopAllMacros - prekida izvršavanje svih makroa koji se trenutno izvršavaju,
- StopMacro - prekida izvršenje aktivnog makroa,
- TransferDatabase - uvoženje ili povezivanje tabela iz drugih baza podataka u trenutnu bazu podataka, kao i izvoženje podataka iz tekuće baze podataka u druge,
- TransferSpreadsheets - uvoženje ili povezivanje podataka iz spreadsheet datoteka u trenutnu bazu podataka, kao i izvoženje podataka iz tekuće baze podataka u jedan od spreadsheet formata,
- TransferText - uvoženje ili povezivanje podataka iz tekstualnih datoteka u trenutnu bazu podataka, kao i izvoženje podataka iz tekuće baze podataka u neki od tekstualnih tipova datoteka.

Moduli

U Access-u, svaka forma i izveštaj "nose" iza sebe sopstvene module. Objekat modul predstavlja kolekciju procedura i funkcija koje nisu napisane za neke konkretne forme ili izveštaje, već se mogu koristiti u svim formama i izveštajima .

Svaki modul se sastoji od deklaracione sekcije i korisničkih funkcija (slika 11). Editor programskog koda podseća na već postojeće editore poznatih programskih jezika, kao što je Microsoft Visual C++ ili Microsoft Visual Basic (na primer, kada se kursor nalazi iznad promenljive u debug modu, pojavljuje se poruka o trenutnoj vrednosti promenljive, zatim prilikom kucanja standardnih naredbi otvara se Combo Box u kome se može izabrati naredba bez daljnjeg kucanja, itd.).

Pisanje programskog koda procedura, funkcija i svojstava u modulima svodi se na poznavanje Visual Basic-a te se preporučuje samo dobrim poznavaocima ovog programskog jezika.



Slika 49 Prikaz modula

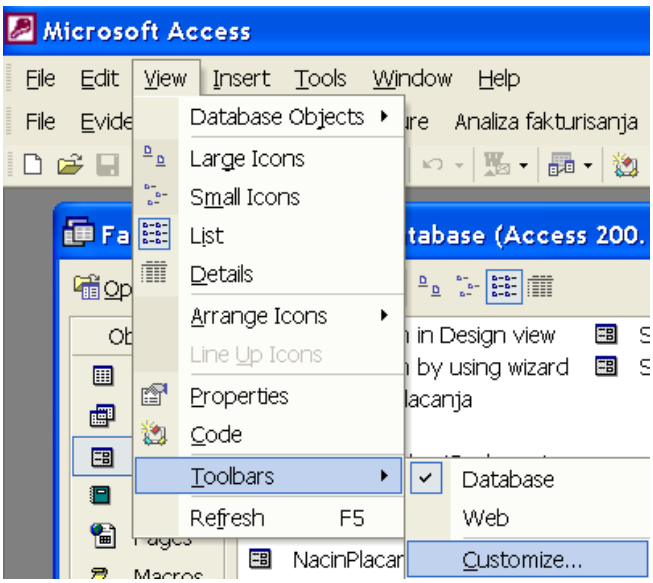
Definisanje menija

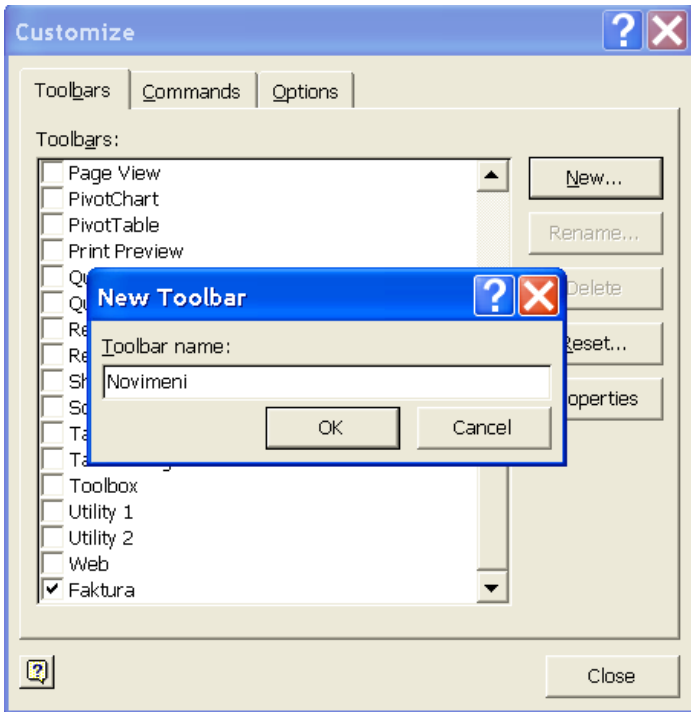
Definisani meniji treba da prate scenario odvijanja aktivnosti budućeg korisnika. Za definisanje menija moraju se koristiti odgovarajuća pravila za strukturiranje kojima se definiše mogući redosled pozivanja operacija.

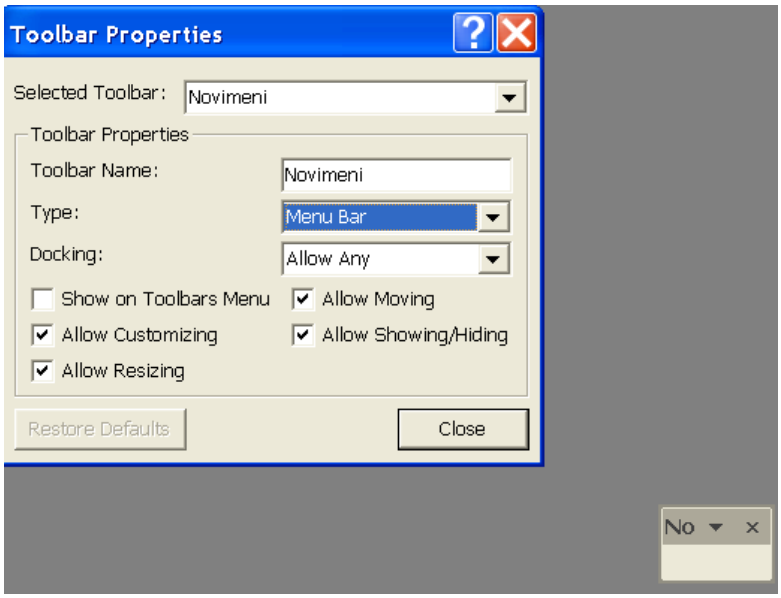
Meniji treba da se definišu na takav način:

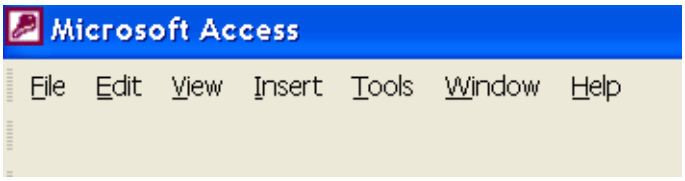
- da svaki meni ima koncizan naslov na vrhu;
- da meniji koji zauzimaju ceo ekran budu balansirani;
- da se razdvajaju liste opcija na više celina;
- da se ograniči broj izbora u meniju na jedan ekran;
- da se razmisli o selekciji menija;
- da se omogući napuštanje menija bez izbora bilo koje opcije;
- da se koriste aktivne imenice za opis opcija menija;
- da se koriste nedvosmislene ikone;
- da se izbegava često naglašavanje;

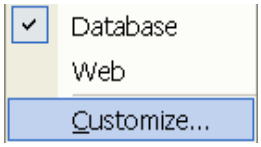
- da se omogući korišćenje i malih i velikih slova,
- da se proveri tastatura pre upotrebe;
- da se izabere jasna, opštepoznata, kratka reč za komande;
- da se omogući korisniku da upotrebi više komandi u jednoj liniji;
- da se omogući korisniku da dobije listu komandi.

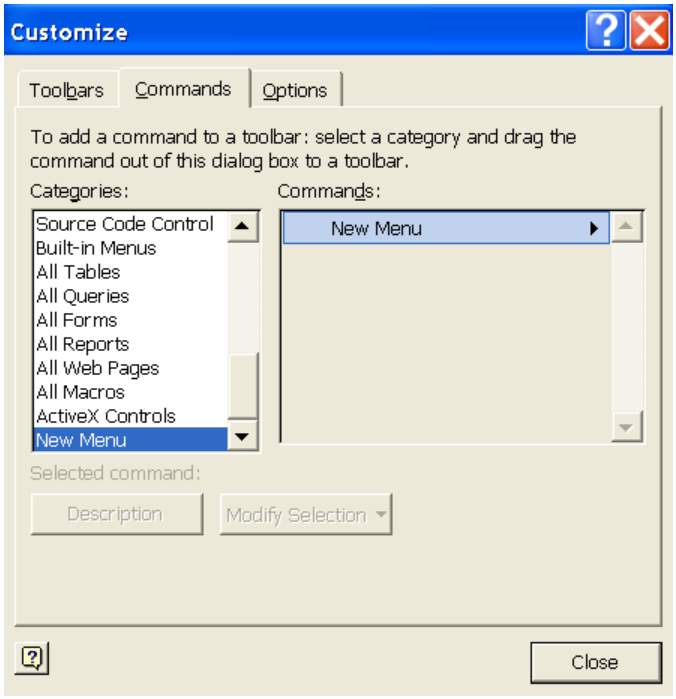


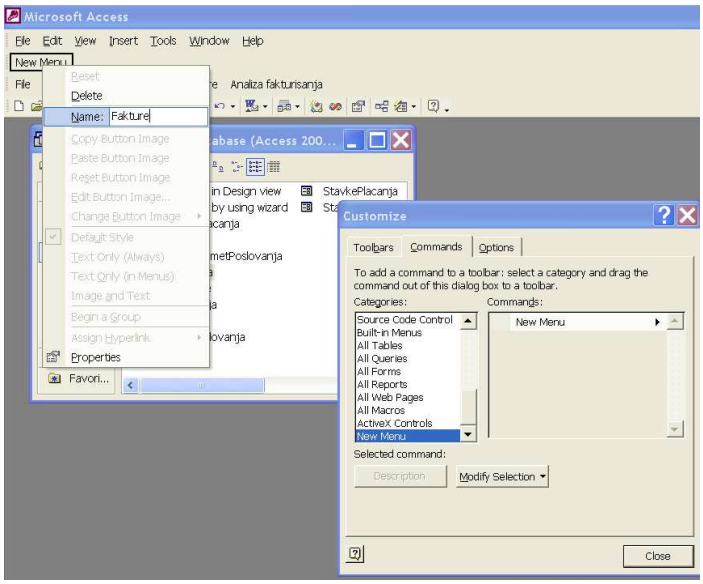

Customize
Toolbars
Commands
Options
Toolbars:
Page View
PivotChart
PivotTable
Print Preview
Toolbox
Utility 1
Utility 2
Web
Faktura
New...
Rename...
Delete
Reset...
operties
Close
New Toolbar
Toolbar name:
Novimeni
OK
Cancel

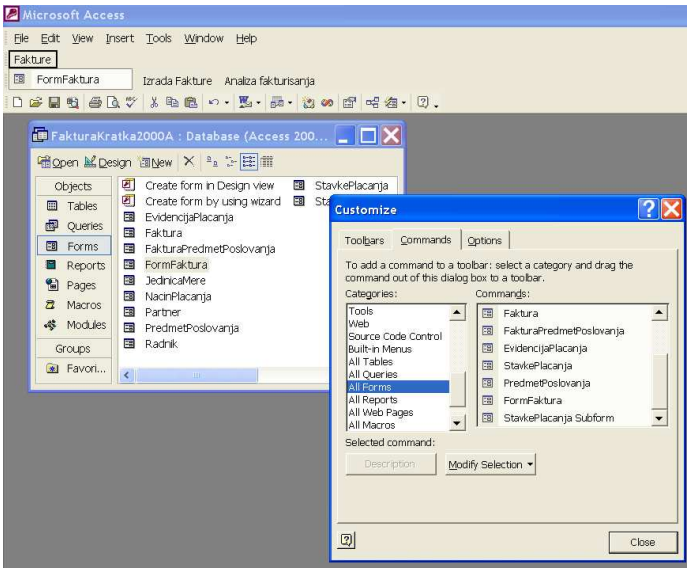

Toolbar Properties
Selected Toolbar:
Novimeni
Toolbar Properties
Toolbar Name:
Novimeni
Type:
Menu Bar
Docking:
Allow Any
Show on Toolbars Menu
Allow Moving
Allow Customizing
Allow Showing/Hiding
Allow Resizing
Restore Defaults
Close
No

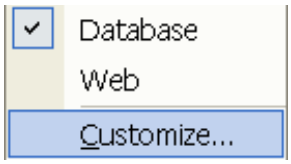

Microsoft Access
File Edit View Insert Tools Window Help

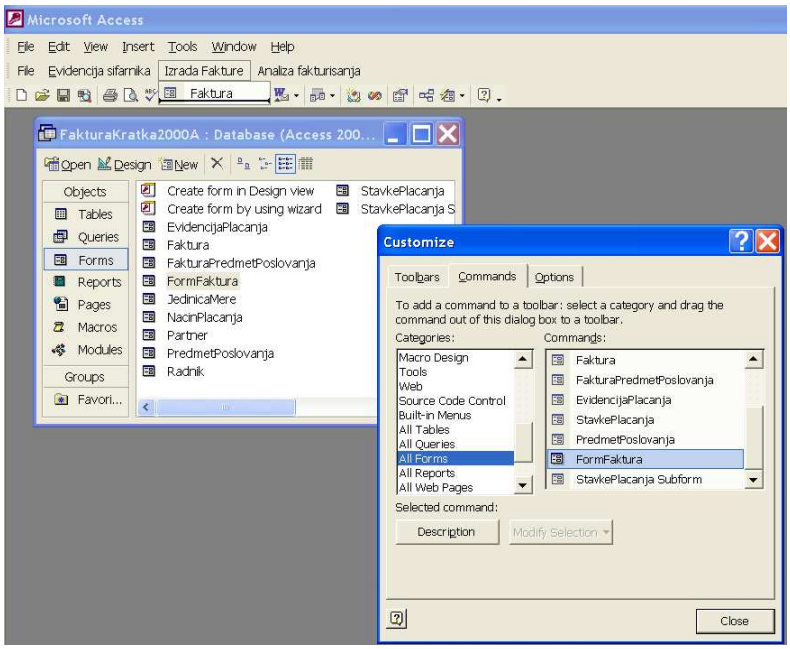

Database
Web
Customize...

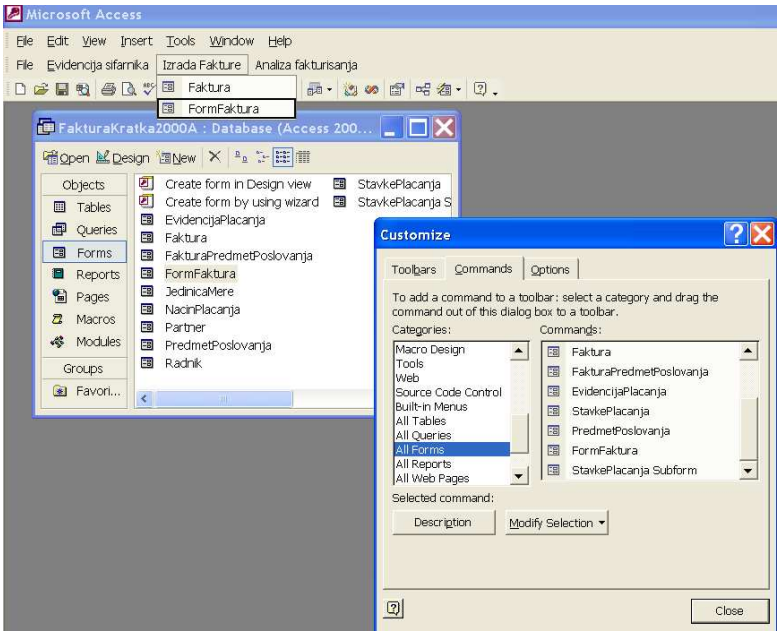

Customize
Toolbars
Commands
Options
To add a command to a toolbar: select a category and drag the command out of this dialog box to a toolbar.
Categories:
Source Code Control
Built-in Menus
All Tables
All Queries
All Forms
All Reports
All Web Pages
All Macros
ActiveX Controls
New Menu
Commands:
New Menu
Selected command:
Description
Modify Selection
Close


Microsoft Access
File Edit View Insert Tools Window Help
New Menu
Reset
Delete
Name: Fakture
Copy Button Image
Paste Button Image
Reset Button Image
Edit Button Image...
Change Button Image
Default Style
Text Only (Always)
Text Only (in Menus)
Image and Text
Begin a Group
Assign Hyperlink
Properties
Analiza fakturisanja
Create form in Design view
Create form by using wizard
StavkePlacanja
Customize
Toolbars Commands Options
To add a command to a toolbar: select a category and drag the command out of this dialog box to a toolbar.
Categories:
Commands:
Source Code Control
Built-in Menus
All Tables
All Queries
All Forms
All Reports
All Web Pages
All Macros
ActiveX Controls
New Menu
Selected command:
Description
Modify Selection
Close


Microsoft Access
File Edit View Insert Tools Window Help
Fakture
FormFaktura
Izrada Fakture
Analiza fakturisanja
FakturaKratka2000A : Database (Access 200...
Open Design New
Objects
Tables
Queries
Forms
Reports
Pages
Macros
Modules
Groups
Favori...
Create form in Design view
Create form by using wizard
EvidencijaPlacanja
Faktura
FakturaPredmetPoslovanja
FormFaktura
JedinicaMere
NacinPlacanja
Partner
PredmetPoslovanja
Radnik
StavkePlacanja
Customize
Toolbars Commands Options
To add a command to a toolbar: select a category and drag the command out of this dialog box to a toolbar.
Categories:
Commands:
Tools
Web
Source Code Control
Built-in Menus
All Tables
All Queries
All Forms
All Reports
All Web Pages
All Macros
Faktura
FakturaPredmetPoslovanja
EvidencijaPlacanja
StavkePlacanja
PredmetPoslovanja
FormFaktura
StavkePlacanja Subform
Selected command:
Description
Modify Selection
Close


Database
Web
Customize...

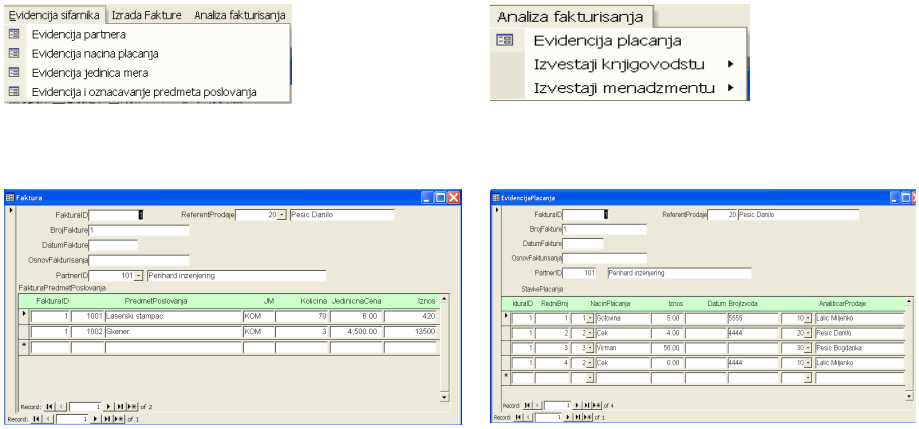

Microsoft Access
File Edit View Insert Tools Window Help
File Evidencija sifarnika Izrada Fakture Analiza fakturisanja
Faktura
FakturaKratka2000A : Database (Access 200...
Open Design New
Objects
Tables
Queries
Forms
Reports
Pages
Macros
Modules
Groups
Favori...
Create form in Design view
Create form by using wizard
EvidencijaPlacanja
Faktura
FakturaPredmetPoslovanja
FormFaktura
JedinicaMere
NacinPlacanja
Partner
PredmetPoslovanja
Radnik
StavkePlacanja
StavkePlacanja S
Customize
Toolbars Commands Options
To add a command to a toolbar: select a category and drag the command out of this dialog box to a toolbar.
Categories:
Macro Design
Tools
Web
Source Code Control
Built-in Menus
All Tables
All Queries
All Forms
All Reports
All Web Pages
Commands:
Faktura
FakturaPredmetPoslovanja
EvidencijaPlacanja
StavkePlacanja
PredmetPoslovanja
FormFaktura
StavkePlacanja Subform
Selected command:
Description
Modify Selection
Close



Slika 50 Definisanje menija

Izrada korisničke aplikacije za primer procesa IZRADA FAKTURE

Izrada korisničke aplikacije za primer procesa FAKTURISANJE prikazna je na sledećim slikama.



Slika 51. Izgled forme za proces IZRADA FAKTURE

Na slici je prikazan primer realizovane forme za proces IZRADA FAKTURE. Forma je podeljena na glavnu formu i stavke fakture. U glavnoj formi definišu se ekranska polja o fakturi, kao što su Broj fakture i Datum fakture, koja odgovaraju definisanim kolonama u okviru tabele Faktura. Ekransko polje FakturaID odgovara koloni FakturaID u tabeli Faktura koja je definisana kao primarni ključ.

U okviru glavne forme definisana su ekranska polja: Partner i Referent prodaje kao tzv. ComboBox za prenesene ključeve Partner, ReferentProdaje u tabeli Faktura. ComboBox nudi listu za izbor iz tabela: Partner i Referent prodaje, gde su PartnerID, ReferentProdajeID primarni ključevi.

6. ANALITIČKE BAZE PODATAKA

Analitičke baze podataka uslovile su nastanak skladišta podataka (Data Warehouse – DW), čija je osnovna namena izveštavanje korišćenjem transakcionih podataka, istorijskih podataka, kao i podataka iz drugih sistema. Rad sa velikim količinama podataka, tj.velikim brojem tabela, čini da brzina odgovora SQL rapidno opada.

Ovo je uslovilo nadgradnju skladišta podataka OLAP sistemima i data mining. OLAP (On Line Analytical Processing) je softverska tehnologija koja omogućava analitičarima i menadžerima brz, interaktivan i konzistentan uvid u informacije, putem širokog spektra mogućih pogleda na informacije.

Kako se za donošenje poslovnih odluka i ovo pokazalo nedovoljnim, pristupa se tzv. traganju kroz podatke (data mining – DM). DM se odnosi na automatsko otkrivanje implicitno prisutnih zakonitosti, pravilnosti i indirektnih sadržaja u velikim bazama podataka savremenih informacionih sistema.

Skladišta podataka

Skladištenje podataka ili Data Warehousing (DW) jeste proces integracije podataka u jedan repozitorijum iz kojeg krajnji korisnici mogu sprovoditi ad-hock analize podataka i praviti izveštaje. Zbog velike količine informacija, skladišta podataka imaju tendenciju da postaju ogromna, što je uslovilo potrebu za visokim performansama hardverskog i softverskog obezbeđenja.

Inicijativa za izgradnju skladišta podataka treba da potekne od rukovodstva, mora da zadovolji specifične potrebe i izgrađuje se kroz saradnju korisnika i projektanata baze podataka.

Glavni posao leži u analizi izvornih podataka i procesa, tj. moraju se poznavati procesi za transakcionu obradu podataka, sa ciljem zapisivanja poslovnih pravila bez grešaka. CASE alati su nezamenjivo sredstvo koje se koristi u izvođenju ovih poslova.

Postoje dve definicije skladišta podataka. Po prvoj definiciji (W. H. Inmon, "Building the Data Warehouse", 1992.), skladište podataka je baza podataka za procese podrške odlučivanju u kojoj su podaci:

- subjektno orjentisani – odslikavaju poslovne procese,
- integrisani – baza podataka konsoliduje podatke iz različitih sistema koji koriste razne vrste kodovanja, mernih jedinica itd. i obezbeđuje konzistentnost podataka,
- vremenski zavisni – svi podaci su u vezi sa nekim vremenskim trenutkom na osnovu kojeg se podaci mogu i porediti,
- nepromenljivi – podaci se, najčešće, pridodaju već postojećim umesto da ih zamenjuju.

Druga definicija je dobijena na osnovu izjava korisnika skladišta podataka i predstavlja odličan okvir za planiranje. Prema toj definiciji, skladište podataka je informaciona baza podataka dizajnirana za podršku jedne ili više klasa analitičkih zadataka, kao što su nadgledanje i izveštavanje, analiza i dijagnoza i simulacija i planiranje.

Zadaci nadgledanja i izveštavanja zasnivaju se na prikupljanju podataka, a ne na otkrivanju i analizi informacija. Procesi analize i dijagnoze se zasnivaju na otkrivanju informacija. Simulacija i planiranje su najsloženiji zadaci, koji zahtevaju mogućnost izmene sadržaja i strukture skladišta podataka. Čim se utvrdi koji od ovih zadataka treba da se implementira u skladištu podataka, može se odabrati pogodna tehnologija koja podržava taj zadatak.

Dakle, "skladište podataka" je analitička baza podataka u kojoj su omogućeni složeni, unapred nepredviđeni (ad-hoc) pristupi velikom broju različitih podataka. Neophodno je razviti IS koji će omogućiti brz i efikasan način zadovoljavanja složenih upravljačkih ad-hoc informacionih zahteva.

Do sada realizovani koncepti "skladišta podataka" fizički su razdvojeni od baze "živih" podataka transakcionog nivoa. U bazi podataka "skladišta podataka" nalaze se podaci koji se periodično repliciraju iz transakcione baze podataka, u skladu sa potrebama sistema za podršku odlučivanju. Osim toga, u ovoj bazi se nalaze i podaci prikupljeni iz eksternih baza podataka, kao i složeni tipovi podataka (grafika, video, World Wide Web itd.).

Za skladišta podataka ili Data Warehous preporučuju se posebne hardverske platforme.

Zašto skladište podataka?

Skladište podataka je, posle Interneta, jedan od najvažnijih industrijskih trendova zbog potrebe da proizvođači budu konkurentniji i bliži kupcu, jer je u opticaju ključna poruka: "Neprekidna evolucija omogućava menadžmentu da postavlja teža i zahtevnija pitanja".

Warehousing koncept je skladištenje agregiranih, ekstrahovanih i filtriranih podataka u meta baze, koje omogućavaju slojevit, multidimenzionalni pristup podacima, kakav je potreban za donošenje odluka najvišeg strateškog nivoa.

Koncept je postavljen veoma fleksibilno i omogućuje naknadno korišćenje različitih alata i modela i posebno se mora naglasiti da transakciona baza nije uslov za primenu koncepta. Koncept se primenjuje nad operativnim podacima, pa ako su oni transakcionog tipa – i nad njima.

Osnovni cilj skladištenja podataka je prikupljanje i distribucija informacija kroz preduzeće, tj. korišćenje bilo koje informacije, sa bilo kog mesta, u bilo koje vreme, tačnije – ostvarenje principa "Biti uvek na usluzi korisniku informacija".

Cilj skladištenja podataka nije da se podaci samo skladište, već je cilj da menadžeri mogu sami da vrše analize. Čak i u svetu uspešnih menadžera postoje ljudi koji nisu tehnički obrazovani, a imaju potrebu za informacijama i ne znaju da programiraju.

Donosioci odluka su pod velikim pritiskom jer moraju da zasnivaju svoje analize na osnovu tekućih činjenica koje se dobijaju iz raznih poslovnih situacija. Te činjenice se čuvaju u on-line transakcionim (OLTP) sistemima i nije im lako pristupiti.

Ovde će biti reči o tome zašto su sistemi koji su dizajnirani za OLTP (On Line Transaction Processing) neodgovarajući za izvršavanje upita za podršku odlučivanju. Takođe, dosadašnji OLTP sistemi će biti upoređeni sa novijim sistemima, koji su optimizovani za poslovnu analizu i koji se nazivaju skladišta podataka (data warehouse).

Ono što krajnjem korisniku treba jeste sledeće:

- da može da postavi bilo koje poslovno pitanje,
- da bilo koji podatak iz preduzeća koristi za analizu,
- mogućnost neograničenog izveštavanja.

Donosiocima poslovnih odluka su potrebni odgovori na pitanja koji direktno utiču na njihovu mogućnost da budu kompetentni na današnjem brzo promenljivom tržištu. Njima su potrebni jasni odgovori na koliko god teška pitanja, i to u što kraćem vremenskom periodu.

Sirovi i neobrađeni podaci, koji su potrebni za poslovnu analizu, nalaze se na različitim lokacijama i u različitim su formatima (npr. hijerarhijske baze podataka, skupovi podataka, datoteke itd.). Takođe, činjenice se prikupljaju i čuvaju u sistemima koji su predviđeni za automatizaciju operacija koje se svakodnevno izvode. To su tzv. OLTP sistemi koji vrše ubacivanje ili ažuriranje podataka u bazama podataka impresivnim brzinama. Ipak, sve te činjenice su van domašaja donosilaca poslovnih odluka.

Skladište podataka je arhitektura za organizovanje informacionih sistema. Sastoji se od skupa programa koji vrše ekstrakciju podataka iz transakcionih sistema, baze podataka u koju se

smeštaju podaci i sistema koji obezbeđuju podatke korisnicima.

U kom pravcu treba ići?

Ako se prihvati ad-hock upit kao širi pojam i kao početni prilaz razvoju skladišta podataka, onda treba posmatrati sledeće elemente:

- Pitanje mogućnosti jednostavnog izdvajanja podataka je problematičano jer je vezano za definisanje tzv. rečenica definisanih u okviru SELECT naredbe, koja zahteva poznavanje relacione baze podataka, što opet uslovljava razvoj meta podataka tj. podataka o podacima.
- Laka izrada izveštaja je moguća jer na raspolaganju stoje mnogi grafički alati za pravljenje izveštaja i grafova.
- Definisanje elemenata OLAP-a za interaktivnu analizu podataka podržavaju, do određenog nivoa, i relacione baze podataka, tj. omogućuju multidimenzionu analizu relacionih podataka. Ova analiza omogućava korisniku da upravlja podacima i menja im perspektivu. Naravno, treba imati u vidu da je težište ovog pristupa prikazivanje relacionih podataka u formatu koji ima smisla za svakodnevne poslovne odluke.
- Pretvaranje podataka u znanje vezano je za razvoj Data mininga, čemu u uslovima ad-hock upita treba da posluže iskustvo i znanje korisnika koji će da definišu pravce svojih pogleda.

Ono što želimo da imamo vezano je za pristup jedinstvenom integralnom izvoru podataka, podržan snažnim analitičkim serverima i snažnim i jednostavnim alatima za podršku odlučivanju, sa mogućnošću samostalnog pravljenja izveštaja i analiza.

Ovo treba da omogući da menadžment donosi poslovne odluke na osnovu jedinstvene i konzistentne slike svih raspoloživih podataka. Dakle, potrebno je postojeće podatke pretvoriti u informacije, a zatim u znanje.

Korišćenjem skladišta podataka, kompanija se više približava kupcima ne bi li bolje razumela njihove zahteve i potrebe. Takođe, kompanije na osnovu prethodnih iskustava pokušavaju da predvide buduće događaje i žele da odrede kupce koji će im obezbediti najveće prihode.

Postoji trend u arhitekturi skladišta podataka koji se sastoji u kreiranju veoma velikih dimenzionih tabela sa višestrukim atributima. Najbolji primer je dimenzija KUPAC. Problem sa ovakvim dimenzijama je što postoji veliki broj atributa, kao što su Pol, Starosna grupa, Etnička grupa, Nivo obrazovanja, Verska pripadnost itd., čijim se odabiranjem može selektovati veliki broj redova podataka. Naprimer, ako se odaberu kupci muškog pola, otprilike će se selektovati polovina tabele. Za tradicionalne sisteme za upravljanje bazama podataka ovo može predstavljati problem.

Jedan od najvećih problema koji se javlja kod tradicionalnih SUBP jesu strategije indeksiranja (tehnologije pristupanja podacima). Kod relacionih SUBP koriste se B-stabla za indeksiranje. Ona su dizajnirana za selekciju relativno malog broja kolona (do 5%).

Warehousing koncept poseduje sve karakteristike potrebne za najviši nivo strateških odluka, imajući u vidu mogućnost pristupa analitičkim bazama podataka i pritom mogućnosti za kreiranje tabela i grafika u cilju izrade i slanja izveštaja korišćenjem Internet servisa (E-mail, telefon, fax i dr.). Da bi se ovakav rad skladišta podataka ostvario koristi se multidimenzioni model, koji reflektuje način na koji korisnik misli o svojim poslovnim podacima i pritom pravi kondenzovane izveštaje, koji prikazuju tekst, slike i multimediju kao dodatak izveštajima i grafikonima.

Warehousing pristup omogućava brzu manipulaciju, agregiranje i lokalne proračune za analize trendova, koristeći viši upravljački nivo kao podlogu za strateško odlučivanje. Strateške odluke zahtevaju predviđanje, statistike, simultane funkcije i analizu vremenskih serija. Korisnici skladišta podataka mogu da postavljaju raznovrsna analitička pitanja na bazi poređenja u vremenu, nalaženja relativnih vrednosti i kreiranja šta-ako scenarija. Mora se istaći da korisnik može da radi i sa osnovnim podacima opciono – spuštanjem ka relacionoj bazi uz minimum programiranja.

U daljem tekstu će biti reči o fundamentalnim idejama koje obezbeđuju uspešnost skladištenja podataka:

- transakciono i analitičko procesiranje,
- "primitivni" podaci nasuprot izvedenim podacima,
- serije vremenskih podataka,
- administracija podataka.

Transakciono i analitičko procesiranje

Kad god se spomene transakciono procesiranje, misli se na sisteme koji izvršavaju svakodnevne poslove neke kompanije. To su OLTP (on line transaction processing) sistemi koji se ažuriraju kontinualno tokom dana. Naprimer, ako neko kupi štampač iz jedne prodavnice računarske opreme, transakcioni sistem će odmah smanjiti ukupni broj štampača za jedan i to će svaki korisnik sistema moći da zna. U slučaju da se dopremi nova količina štampača u prodavnice, transakcioni sistem će to odmah evidentirati. Ako se desi da se broj štampača smanji na neki određeni broj, transakcioni sistem može automatski da generiše zahtev nabavljačima za novim štampačima.

Analitički sistemi su sistemi koji obezbeđuju informacije koje se koriste za analizu problema ili situacija. Ona se primarno vrši korišćenjem poređenja ili analiziranjem šablona i trendova. Naprimer, analitički sistem bi mogao da prikaže kako se određena vrsta štampača prodaje u

različitim delovima zemlje. Takođe, mogao bi i da prikaže kako se jedna vrsta proizvoda prodaje sada u odnosu na period kada se proizvod prvi put pojavio na tržištu. Za analitičke sisteme razvijaju se analitičke baze podataka.

Analitičke baze podataka ne sadrže ažurirane podatke, već čuvaju informacije iz određenog trenutka vremena. Takvi podaci su od izuzetnog značaja za poređenja i analizu trendova. Naprimer, moguće je utvrditi da je prodaja u jednom mesecu znatno opala samo ako u sistemu postoje podaci o prodaji u prethodnim mesecima, tako da se može vršiti poređenje. Ovakvo poređenje je skoro nemoguće izvesti u OLTP sistemima, jer se u takvim sistemima podaci neprestano menjaju.

Informacije iz određenog trenutka vremena često se nazivaju i snimcima podataka. Naprimer, ako analitička baza podataka sadrži snimke podataka o prodaji koji se uzimaju svako veče, onda će se pri analizi podataka sutradan znati da su to podaci koji su stari jedan dan. Tada dnevna, nedeljna i godišnja poređenja imaju smisla jer će uvek na raspolaganju biti konzistentne vrednosti sa kojima se može vršiti poređenje.

Analiziranje šablona podataka i trendova zahteva postojanje velikog broja istorijskih podataka. Naprimer, podatak da žene sve više kupuju određeni proizvod može uticati na to da kompanija promeni marketinšku kampanju.

Ideja da su podaci u transakcionim sistemima promenljivi, a da su podaci analitičkih sistema nepromenljivi, direktno utiče na različitu funkcionalnost ovih sistema. Analitičke baze podataka su, najčešće, projektovane samo za čitanje. Tada korisnici mogu samo da pregledaju podatke ili eventualno da vrše neke manipulacije nad njima, ali nisu u stanju da ih menjaju.

Druga karakteristika koja razdvaja transakcione sisteme od analitičkih jeste dizajn baze podataka. Transakcioni sistemi su dizajnirani tako da preuzimaju podatke, vrše izmene nad postojećim podacima, daju izveštaje, održavaju integritet podataka i upravljaju transakcijama što je brže moguće. Analitički sistemi nisu predviđeni da obavljaju ove poslove. Oni se dizajniraju za veliki broj podataka namenjenih samo za čitanje, obezbeđujući informacije koje se koriste za donošenje odluka. Skladište podataka je analitička baza podataka namenjena samo za čitanje i koristi se kao osnova sistema za podršku odlučivanju.

Tradicionalni OLTP sistemi, koji su predviđeni za automatizaciju operacija koje se izvode svakog dana, veoma su brzi pri smeštanju podataka u bazu podataka, ali nisu prihvatljivi kada se koriste za izvršavanje analiza. Dobijanje činjenica traje veoma dugo (npr. izrada jednog tipičnog ad-hoc izveštaja može potrajati danima), a kad donosilac odluka i dobije izveštaj i ručno ga uporedi sa drugim izveštajima, onda je već kasno – poslovna dinamika se promenila.

Jasno je da se OLTP sistemi ne mogu koristiti za čuvanje činjenica i istorijskih poslovnih podataka koji se koriste u poslovnim analizama. Oni su veoma brzi, tačni i efikasni za unos podataka u baze podataka, ali ne mogu da obezbede brze odgovore na ad-hoc upite. Takođe, podaci koji se čuvaju u OLTP bazama podataka su nekonzistentni i neprestano promenljivi. Često postoje dupli zapisi transakcija koji bi samo zbunili donosioca odluka pri analizi.

Nedostatak istorijskih podataka u OLTP sistemima čini ih neprihvatljivim za analizu trendova. Čak i kad se dobiju podaci iz OLTP sistema, oni su i dalje sirovi i prilično nerazumljivi. Prema tome, OLTP podaci su daleko od poslovnih činjenica koje su neophodne donosiocima poslovnih odluka.

U OLTP sistemima više se zadaju zahtevi nego što se postavljaju pitanja (na primer: ažuriraj podatke o rezervacijama za putnički avion). Povremeni upiti su ograničeni na lociranje određenog zapisa u informacionom sistemu i njegovo pripremanje za ažuriranje ili izvršavanje jednostavnih agregacija. Donosioci odluka postavljaju pitanja koja su potpuno suprotna od onih koja se postavljaju u OLTP sistemima. Takvi upiti idu od poslovnih potreba za analizom podataka pa sve do donošenja odluka i preporuka. Ovakva pitanja su veoma složena i tipično zahvataju dimenzije koje u OLTP sistemima i ne postoje, kao što su vremenski periodi, regije sveta, vrste proizvoda itd. Tako, naprimer:

- Koji se proizvod najbolje prodaje u srednjoj Evropi i u kom je to odnosu sa demografskim podacima?
- Da li je unapređenje prodaje prošlog meseca bilo bolje nego prošle godine?

Na ova pitanja ne može se dobiti odgovor u OLTP sistemima. OLTP sistemi su dizajnirani tako da prikupljaju informacije i da se ažuriraju veoma brzo.

Sistemi za podršku odlučivanju se po svojim karakteristikama razlikuju od transakcionih sistema. Transakcioni sistemi pristupaju i vrše ažuriranje zapisa podataka jednog poslovnog objekta ili događaja (naprimer, jedan račun, jedna narudžbenica i slično). Ovakve transakcije su, najčešće, unapred definisane i zahtevaju da baza podataka obezbeđuje brzi pristup zapisima podataka. Korisnici sistema za podršku odlučivanju su, najčešće, menadžeri koji razmišljaju o budućim događajima. Njihovi upiti mogu zahtevati prikupljanje velikog broja podataka da bi se izvršila potrebna analiza. Skladište podataka se koristi da se spoje dobre karakteristike i transakcionih sistema i sistema za podršku odlučivanju.

U tabeli 2 date su neke od razlika između transakcionih sistema i skladišta podataka.

Tabela 2

	Transakcioni sistemi	Skladište podataka
Sadržaj podataka	tekuće vrednosti	arhivski podaci, sumarni podaci, proračunati podaci
Struktura podataka	složena i pogodna za operaciona proračunavanja	jednostavna i pogodna za poslovnu analizu
Verovatnoća pristupanja	velika	srednja do mala
Vreme odziva	reda sekunde	reda minuta

Namena	automatizacija svakodnevnih operacija	nalaženje i analiza informacija
Model podataka	normalizovan	dimenzionalan
Pristup	SQL	SQL i alati za poslovnu analizu
Tip podataka	podaci koji upravljaju poslovima	informacije za poslovnu analizu
Stanje podataka	dinamično (promenljivi podaci)	statično (istorijski i opisni podaci)

"Primitivni" i izvedeni podaci

Element "primitivnog" podatka opisuje individualni objekat ili događaj, a element izvedenog podatka opisuje više različitih objekata ili događaja. Element "primitivnog" podatka obično ne može da se koristi u računskim operacijama, dok se elementi izvedenih podataka računaju na osnovu primitivnih ili ostalih izvedenih elemenata. Naprimer, Cena može biti "primitivni" element ako opisuje jedan objekat u datom trenutku. S druge strane, Prosečna cena može biti izvedena za dati proizvod za celu godinu.

Često se pravi greška kada se misli da su menadžerima potrebni samo sumarni podaci te skladište podataka može biti popunjeno samo izvedenim elementima. Jedan od razloga za uvođenje i "primitivnih" podataka u skladište podataka jeste mogućnost vršenja operacija drill down/up, pomoću kojih korisnik može pristupati različitim nivoima detalja. Drugi razlog za uvođenje detaljisanih podataka u skladište podataka jeste što sumarni podaci ograničavaju mogućnost korisnika da kreira različite sumarne podatke. Naprimer, ako baza podataka ima mogućnost da kreira sumarne podatke o prodaji samo po različitim zemljama, onda korisnik ne može da utvrdi da li je prodaja bolja na početku meseca ili na kraju. Treći razlog za korišćenje i "primitivnih" podataka u skladištu podataka jeste taj što je mnogo teže modelovati sve moguće izvedene podatke nego sve moguće "primitivne" podatke. Pri tome količina izvedenih podataka može biti mnogo veća od količine "primitivnih" podataka.

Serije vremenskih podataka

Za razliku od transakcionih sistema, sistemi za podršku odlučivanju uzimaju u obzir trendove, umesto da se zasnivaju na jednom trenutku u vremenu. Posledica ovakvog razmišljanja je da svaki element podatka mora nositi zajedno sa sobom i trenutak vremena na koji se odnosi. Procesiranje mesečnih i godišnjih podataka je relativno jednostavno. Problem predstavljaju nedeljni i dnevni podaci, s obzirom da za svaku godinu postoji različit broj nedelja i dana. Zato je

dizajniranje programa za analizu dnevnih i nedeljnih podataka teži posao.

Administracija podataka

Korisnost sistema se može odrediti na osnovu dostupnosti i kvaliteta podataka koje obezbeđuje. Većina postojećih aplikacija je namenjena za specifične zahteve korisnika i mogu se koristiti samo za te potrebe. Različite aplikacije koje su razvijene u različitim vremenskim trenucima i koje su namenjene za različite potrebe korisnika dovode do pojave nekonzistentnosti i redundantnosti podataka. Elementi podataka sa istim nazivom mogu biti različito definisani. Jedan isti element podatka u dva različita sistema može biti sačuvan pod različitim nazivom. Prema tome, da bi se pravilno vodila administracija podataka, mora se odrediti tim ljudi koji će voditi računa o kvalitetu podataka.

Datamart

Datamart je subjektno orjentisani poslovni pogled na skladište podataka. On sadrži značajno manje podataka od "skladišta podataka" i predstavlja objekt analitičkog procesiranja od strane korisnika. U okviru "skladišta podataka" datamartovi se koriste za tzv. informatička ostrva vezana za finansije, proizvodnju i dr. Na ovaj način se želi da pojedini segmenti preduzeća donose bolje odluke.

Datamartovi su subjektno orijenisane multidimenzione baze podataka sa životnim ciklusom od tri godine. Mnogi datamartovi su podskup velikih skladišta podataka.

Datamartovi su multidimenzionalni i omogućuju korisnicima više kriterijuma za upoređivanje, korišćenjem ad-hoc upita. Mnogi upiti nad operacionim bazama podataka se preprogramiraju ili konzerviraju. Konzervirani (cannet) upit je onaj upit koji je postavljen da bi pokrenuo korisnički upit koji će obezbediti podatke u već ranije određenom formatu.

Datamart mora biti sposoban da podržava n načina upita sa mrežom indeksa. Operator može da koristi OLAP alate i napravi izveštaj od informacija iz jedne tabele u datamartu koristeći bilo koju kolonu kao selekcioni kriterijum. Takođe, može da poveže podatke iz dve ili više tabela u datamartu, spajajući objekte preko prenesenih ključeva.

Datamart služi kao osnova za OLAP u sistemima za podršku odlučivanju.

Osnovna pitanja koja se postavljaju prilikom izgradnje Datamart arhitekture za podršku odlučivanju su:

- Šta želite da dobijete od datamarta?
- Na koji način želite da informacija bude prezentovana?

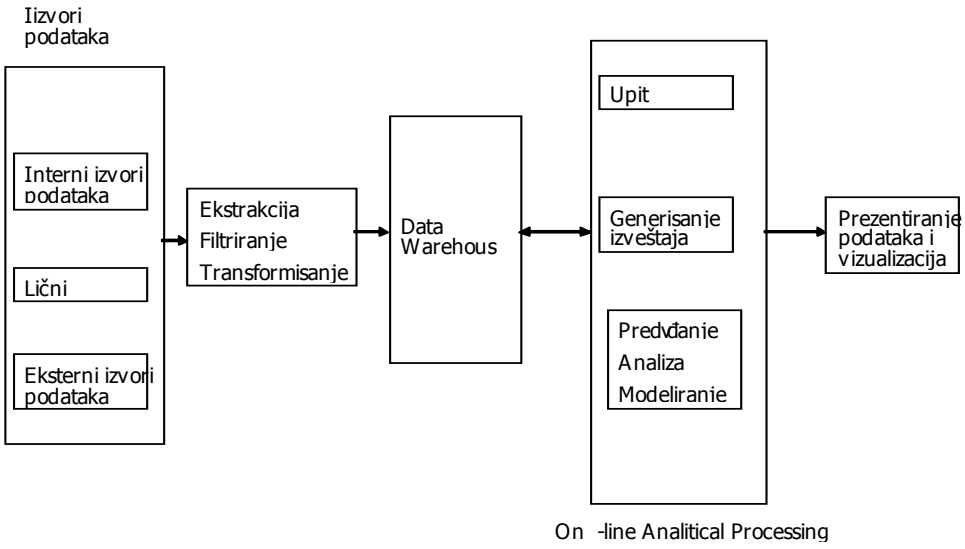
- Koji nivo zbirnih informacija se želi?
- Koje tabele se obično povezuju (join) u OLAP procesiranju upita?

Kako Datamart može biti deo integralnog "skladišta podataka", to je veoma bitan odnos ove dve vrste "skladišta podataka", kao i način njihovog formiranja iz transakcionoh baza podataka.

Može se zaključiti da koncept "skladišta podataka", koga podržava savremena informaciona tehnologija, u potpunosti omogućuje razvoj efikasnih sistema za podršku odlučivanju.

Razvoj skladišta podataka

Skladište podataka predstavlja specifičnu bazu podataka, namenjenu podršci odlučivanju u određenoj organizaciji. Za razliku od transakcionih sistema (OLTP sistemi), koji su orjentisani poslovnim procesima, skladišta podataka su subjektno orjentisana, što znači da su fokusirana na subjekte u poslovnim procesima, kao što su kupci, zaposleni i dobavljači. Integrisanost podataka u skladištima podataka obezbeđuje da se podaci predstavljaju u konzistentnim formatima korišćenjem konvencija pri zadavanju imena i ograničenja nad domenima, atributima i merama. Podaci u skladištima podataka su vremenski zavisni, što znači da je svaki podatak koji se nalazi u skladištu podataka u vezi sa nekim vremenskim trenutkom. Na kraju, podaci u skladištima podataka su nepromenljivi, tj. čim se neki podatak upiše u skladište podataka, moguće mu je samo pristupati. Na sledećoj slici su prikazani svi elementi potrebni za razvoj skladišta podataka.



Slika 52. Data warehous i OLAP

Pri izgradnji skladišta podataka najbitniji su sami podaci, a ne poslovni procesi i funkcije, kao što je to slučaj sa transakcionim sistemima. Baze podataka namenjene sistemima za podršku odlučivanju mogu biti veoma velike (terabajtne), pri čemu neke tabele mogu sadržati i gigabajt podataka. Zato se veličina baze podataka mora uzeti u obzir pri planiranju skladišta podataka.

Za razvoj skladišta podataka potrebno je:

- izvršiti analizu izvora podataka,
- pripremiti podatake,
- izgraditi skladište podataka.

Analiza izvora podataka

Osnovni izvori podataka za koncept skladišta podataka su operativni (transakcioni), tzv. OLTP (On-Line Transaction Processing) podaci, kao i spoljne informacije nastale kao istorija poslovanja, ili industrijski i demografski podaci uzeti iz velikih javnih baza podataka. Analiza izvornih podataka se smatra ključnim elementom i oduzima 80% vremena, jer je potrebno definisati odgovarajuća pravila za preuzimanje podataka iz izvornih podataka. Znanja vezana za ovu oblast su najčešće u glavama onih koji treba da koriste skladište podataka. Ovde do izražaja neosporno dolaze i metode vođenja intervjua, kao i korišćenje CASE alata, naročito prilikom definisanja poslovnih pravila. Na osnovu iskustva autora, postojeća dokumentacija najčešće ne

daje dovoljno podataka za ekstrakciju znanja. Korišćenjem CASE alata, kao što je ranije pokazano, definišu se procesi i struktura podataka koja je potrebna, a koja se nalazi u OLTP i u spoljnim izvorima informacija.

Analiza izvora podataka prolazi kroz sledeće faze:

- prikupljanje zahteva,
- planiranje skladišta podataka,
- izbor tehnike analize podataka.

Prikupljanje zahteva

U ovoj fazi razvoja skladišta podataka, razmatraju se poslovne potrebe i zahtevi budućih korisnika sistema. Postoji mnogo metoda za prikupljanje poslovnih zahteva. U opštem slučaju, ove metode mogu biti smeštene u dve kategorije: prikupljanje izvornih zahteva i prikupljanje korisničkih zahteva.

Prikupljanje izvornih (source-driven) zahteva

Prikupljanje izvornih zahteva, kao što i samo ime kaže, jeste metoda bazirana na definisanju zahteva korišćenjem izvornih podataka u produkciono-operativnim sistemima. Ovo se radi analiziranjem ER-modela izvornih podataka.

Glavna prednost ovakvog pristupa je što od početka znate da možete da podržite sve podatke, jer ste već ograničili sami sebe samo na one podatke koji su na raspolaganju. Druga dobit je u tome što možete da svedete na minimum vreme potrebno korisniku u ranim fazama (stanjima) projekta.

Naravno, postoje i nedostaci ovakvog pristupa. Umanjivanjem korisnikovog učešća, povećava se rizik od promašaja ispunjenja zahteva korisnika. U zavisnosti od količine izvornih podataka koju imate i kvaliteta ER-modela za njih, ovaj pristup može oduzeti dosta vremena. Možda je najvažnije to da neki od ključnih korisničkih zahteva u datom momentu nisu dostupni. Bez mogućnosti za identifikovanjem ovakvih zahteva, ne postoji mogućnost da ispitate šta je uključeno, ili šta se pojavljuje u eksternim podacima. Eksterni podaci su takva vrsta podataka koja postoji van organizacije.

Ovaj pristup omogućava da korisnici imaju uvida u to šta vi posedujete. Uvereni smo da postoje bar dva slučaja gde je ovakav pristup aprioran. Prvo, u relaciji sa dimenzionim modelovanjem, može biti upotrebljen da daje prilično jasne glavne dimenzije od interesa jedne organizacije. Ako se uslovno planira skladište podataka na nivou organizacije, ovo može umanjiti gomilanje dimenzija kroz odvojeni razvoj datamartova. Drugo, analiziranjem veza između izvornih podataka mogu se identifikovati područja na koja će se koncentrisati napori razvoja skladišta podataka.

Prikupljanje korisničkih (User-Driven) zahteva

Prikupljanje korisničkih zahteva je metoda koja se bazira na definisanju zahteva istraživanjem funkcija kojima korisnik teži, odnosno koje korisnik izvršava. Ovo se obično postiže kroz seriju sastanaka i/ili intervjua sa korisnikom.

Glavna prednost ovog pristupa je što se koncentriše na ono što je potrebno, a ne na ono što je dostupno. Uopšteno, ovaj pristup ima manje područje posmatranja nego izvorišno upravljani pristup. Zbog toga ovaj pristup proizvodi upotrebljivo skladište podataka u kraćem vremenskom periodu. Sa druge strane, izuzeci moraju biti što čvršće kontrolisani. Korisnici moraju dobro razumeti da je moguće da se neki od podataka koji su im potrebni jednostavno ne mogu učiniti dostupnim. Ovo je važno zbog toga što se ne sme ograničiti ono što korisnici zahtevaju. Konkretno razmišljanje treba da otpočne tek kada se definišu potrebni zahtevi za skladištenje podataka. Ako je korisnik previše usko fokusiran, moguće je propustiti korisne podatke koji su dostupni u produkcionim sistemima.

Prikupljanje zahteva podrazumeva davanje odgovora na sledeća pitanja:

- Ko (ljudi, grupe, organizacije) je zainteresovan kao korisnik?
- Šta (koju funkciju) korisnik pokušava da analizira?
- Zašto su korisniku potrebni podaci?
- Kada (za koje vreme) podaci treba da budu učitani?
- Gde (geografski, organizaciono) se proces pojavljuje?
- Do koje mere je potrebna analiza performansi, ili stanja funckija procesa?
- Kako korisnik vodi posao?
- Koji su atributi potrebni korisniku?
- Koje su poslovne hijerarhije?
- Koje podatke korisnici trenutno koriste?
- Koji su podaci potrebni korisnicima?
- Koji je nivo detalja potreban korisnicima?

Da bi se dobili odgovori na ova i slična pitanja, korisnici se moraju intervjuisati. Najčešće se prvo intervjuišu ključni ljudi u organizaciji, kao što su analitičari, menadžeri i izvršioci, jer oni imaju najveće potrebe za podacima.

Veoma je bitno da se utvrdi protok informacija u i iz svakog odeljenja. Zato se moraju prikupiti podaci o tome koji izveštaji i dokumentacija pristižu u odeljenje, kako se koriste, ko ih koristi,

koliko često pristižu itd. Često se dešava da podaci u odeljenje pristižu na papiru i da se zatim ponovo unose ručno u tabele da bi se mogli vršiti dodatni proračuni. Jedan od razloga uvođenja skladišta podataka jeste upravo da se izbegnu takve situacije kad god je moguće. Takođe, bitni su i podaci o tome koje izveštaje odeljenje kreira, koliko često, kome ih upućuje, koliko je vremena potrebno da se napravi izveštaj, koliko ljudi radi na njemu i slično.

Sve prikupljene podatke potrebno je dokumentovati jer broj intervjuisanih korisnika može biti veliki, te se može desiti da se neki zahtevi i potrebe korisnika zaborave. Dobijene podatke treba organizovati u nekoliko sekcija, kao što su:

- podaci o analizi (podaci o svim vrstama analiza koje se trenutno koriste) i
- zahtevi vezani za podatke (opis svih polja podataka koja se koriste, nivo detalja, izvori).

Tako organizovane podatke treba proslediti svim učesnicima intervjua da bi se čulo i njihovo mišljenje i da bi se izvršile potrebne korekcije.

Planiranje skladišta podataka

Planiranje skladišta podataka obuhvata mnoge zadatke koji se javljaju i pri razvoju bilo kojeg projekta.

Planiranje skladišta podataka sastoji se od sledećih zadataka:

- definisanje obima projekta,
- kreiranje projektnog plana,
- definisanje tehničkih uslova,
- definisanje resursa, zadataka i vremenskih rokova.

Kreiranje projektnog plana i definisanje realnih vremenskih rokova može biti težak posao. Veoma je bitno da se pre početka razvoja projekta razmotri arhitektura i infrastruktura skladišta podataka.

Tehnička infrastruktura je usko povezana sa arhitekturom. To su razne tehnologije, platforme, baze podataka i ostale komponente koje podržavaju izabranu arhitekturu skladišta podataka. Tehnička infrastruktura uključuje i izbor instalacije baze podataka, podešavanje mrežnog okruženja, kao i izbor i instalaciju alata za rad sa bazom podataka. Potrebno je naglasiti i da jedna arhitektura može imati više različitih infrastruktura, u zavisnosti od okruženja kompanije. Preporučuje se da identifikacije arhitekture skladišta podataka i infrastrukture budu odvojeni projekti koji se moraju završiti pre započinjanja razvoja samog skladišta podataka.

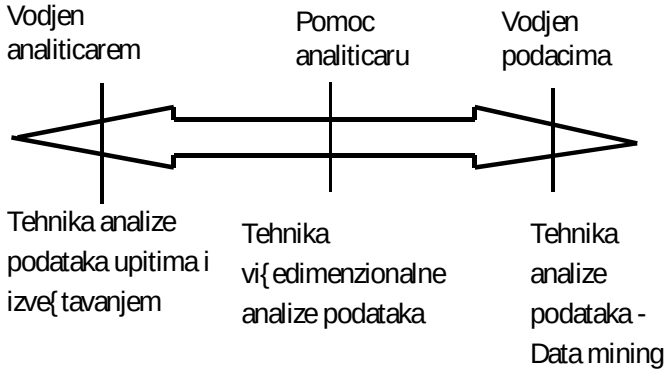
Potrebno je napomenuti da se moraju uzeti u obzir i neki tehnički uslovi. Oni se moraju uključiti u plan projekta, s obzirom da zahtevaju određene resurse i vreme. Pri tome se misli na:

- planiranje kapaciteta,
- strategije arhiviranja,
- procedure pomoću kojih će korisnici pristupati arhiviranim podacima,
- strategije osvežavanja i ažuriranja podataka,
- vremensko planiranje poslova.

Vrlo često se dešava da se pre razvoja samog skladišta podataka izradi probni projekat kako bi se stekla iskustva i prikazao značaj skladišta podataka samim korisnicima. Interaktivni pristup podacima pokazaće korisnicima koje prednosti donosi skladište podataka. Takođe, projektanti će steći bolji uvid u potrebe korisnika, bolje će proceniti vremenske rokove, resurse, potrebnu arhitekturu i infrastrukturu.

Izbor tehnike analize podataka

Skladište podataka se gradi da bi se obezbedio lako pristupačan izvor podataka visokog kvaliteta. Obično postoji potreba da se vrše analize i donose odluke kroz korišćenje tog izvora podataka. Postoji nekoliko tehnika analize podataka koje su danas u širokoj upotrebi. To su upiti i izveštaji, višedimenzionalne analize i data mining. One se koriste za formulisanje i prikazivanje rezultata upita, analizu sadržaja podataka njihovim posmatranjem iz različitih perspektiva i otkrivanje šablona i klasterisanih atributa u podacima koji će omogućiti dublji pogled u sadržaj podataka.



Slika 53. Izbor tehnike analize podataka

Tehnike analize podataka mogu uticati na tip odabranog modela podataka i njegov sadržaj. Naprimer, ako je namera da se obezbedi jednostavna mogućnost upita i izveštaja, model podataka koji struktuira podatke na normalizovani način verovatno će obezbediti najbrži i nalakši pristup podacima. Mogućnost upita i izveštavanja se primarno sastoji od biranja povezanih

elemenata podataka, eventualnog njihovog sumiranja i grupisanja u neku kategoriju i prezentovanja rezultata. Izvršavanje ovog tipa mogućnosti uglavnom može da dovede do korišćenja direktnijeg skeniranja tabela. Za ovu vrstu mogućnosti, ER model sa normalizovanom i/ili denormalizovanom strukturom podataka je najprikladniji.

Ako je cilj obezbediti višedimenzionalnu analizu podataka, prikladniji bi bio dimenzionalni model podataka. Ova vrsta analaize zahteva da model podataka podržava strukturu koja omogućava brz i lak pristup podacima na osnovu bilo kakvih numeričkih kombinacija dimenzija analize. Naprimer, možete hteti da znate koliko je određenih proizvoda prodato određenog dana, u određenoj prodavnici i u određenom rasponu cena. Onda za dalju analizu možete hteti da znate koliko prodavnica je prodalo određeni proizvod, u određenom rasponu cena, određenog dana. Ova dva pitanja zahtevaju slične informacije, ali jedna posmatrane iz ugla proizvoda, a druga iz ugla prodavnice.

Višedimenzionalna analiza zahteva model podataka koji će omogućiti da se podaci lako i brzo mogu pogledati iz bilo koje moguće perspektive ili dimenzije. Pošto se koristi više dimenzija, model mora da obezbedi način da se podacima brzo pristupa. Ako se koriste visoko normalizovane strukture podataka, biće potrebno mnogo grupisanja između tabela koje sadrže različite dimenzije podataka i mogu značajno uticati na performanse. U ovom slučaju bi dimenzionalni model podataka bio prikladniji.

Razumevanje podataka i njihova upotreba utiču na izbor modela podataka. Jasno je i da se, u većini implementacija, može koristiti više tipova modela podataka da bi se najbolje zadovoljili različiti zahtevi skladišta podataka.

Tehnika analize podataka upitima i izveštavanjem

Analiza upitima i izveštajima je proces postavljanja pitanja na koje se traži odgovor, izdvajanje podataka od značaja iz skladišta podataka, njihova transformacija u odgovarajući kontekst i prikazivanje u čitljivom formatu. Ovim procesom upravlja analitičar, koji mora postavljati pitanja da bi dobio odgovor. Primetićete da je ovo pomalo različito od, naprimer, data mininga koji je vođen podacima.

Tradicionalno, upiti su radili sa dve dimenzije ili dva faktora u trenutku vremena. Naprimer, neko može pitati: "Koja količina nekog proizvoda je prodata ove nedelje?" Upiti koji slede ovom pitanju će biti urađeni da odrede koja količina proizvoda je prodata u određenoj prodavnici. Definicija upita je proces uzimanja poslovnih pitanja ili hipoteza i njihovo prevođenje u format upita koji može koristiti određeni alat za podršku pri odlučivanju. Kada se upit izvrši, alat generiše odgovarajuće komande za dobijanje traženih podataka, koji se smeštaju u skup odgovora. Analitičar podataka zatim obavlja potrebne kalkulacije i manipulacije na skupu odgovora da bi dobio željene rezultate. Ovi rezultati se zatim formatiraju da bi odgovarali obrascu prikaza ili izveštaja koji je odabran da krajnjem korisniku olakša razumevanje. Ovaj obrazac se može sastojati od kombinacije teksta, grafike, videa i audia. Na kraju, izveštaj se dostavlja krajnjem korisniku na željenom izlaznom medijumu koji može biti papir, monitor, ili se može predstaviti

zvukom.

Krajnji korisnici su prvenstveno zainteresovani za obradu numeričkih podataka koje koriste za analizu ponašanja poslovnih procesa. Oni takođe mogu da računaju ili istražuju kvalitativne mere, kao što su stepen zadovoljstva korisnika, kašnjenje u poslovnim procesima ili pogrešne isporuke. Oni takođe mogu analizirati efekte poslovnih transakcija ili događaja, analizirati trendove ili vršiti ekstrapolaciju njihovih predviđanja za budućnost. Često će prikazani podaci uzrokovati da korisnik formuliše drugi upit da bi razjasnio skup odgovora ili prikupio detanjnije informacije. Proces se nastavlja dok se ne dobiju željeni rezultati.

Tehnika višedimenzionalne analize podataka

Višedimezionalna analiza je način da se prošire mogućnosti upita i izveštaja. Ovo znači da se umesto izvršavanja višestrukih upita podaci struktuiraju da bi se omogućio brz i lak pristup odgovorima na pitanja koja se tipično postavljaju. Naprimer, podaci su struktuirani tako da sadrže odgovore na pitanje: "Koja količina svakog proizvoda je prodata određenog dana, od strane određenog prodavca u određenoj prodavnici?" Svaki deo ovog upita se naziva dimenzija. Računanjem odgovora unapred za svaki podupit u okviru većeg konteksta, mnogo odgovora može biti uvek dostupno pošto se rezultati ne računaju ponovo za svaki upit, već im se lako pristupa i lako se prikazuju.

Naprimer, ako imate rezultate gornjeg upita, automatski imate odgovore na bilo koji od podupita. Ovo znači da ćemo već znati odgovor na podupit: "Koju količinu određenog proizvoda je prodao određeni prodavac?" Imati podatke kategorizovane po ovim različitim faktorima ili dimenzijama čini da poslovno orijentisani korisnici lakše razumeju podatke. Dimenzije mogu imati individualne entitete ili hijerarhiju entiteta, kao što su region, prodavnica i odeljenje.

Višedimanzionalne analize omogućavaju korisnicima da sagledaju veliki broj međuzavisnih faktora koji učestvuju u poslovnom problemu i da pregledaju podatke u složenim vezama. Krajnji korisnici su zainteresovani u istraživanju podataka na različitim nivoima detaljnosti, koji se dinamički određuju. Složene veze mogu biti analizirane kroz iterativni proces koji sadrži probijanje na niže nivoe detaljnosti ili dizanje na više nivoe sumarizacije i agregacije. Kao kod upita i izveštavanja, višedimenzionalne anlize se nastavljaju dokle god se vrše probijanja dole i vraćanja gore.

Tehnika analize podataka – Data mining

Data mining je relativno nova tehnika anlize podataka. Veoma je različita od upita i izveštaja, kao i od višedimenzionalnih analiza, po tome što koristi tehniku otkrivanja. Ovo znači da ne pitate određeno pitanje već koristite određene algoritme koji analiziraju podatke i izveštavaju šta su otkrili. Za razliku od upita, izveštaja i višedimenzionalnih anliza, gde je korisnik morao da kreira i izvršava upite zasnovane na hipotezama, data mining traži odgovore na pitanja koja ne moraju biti prethodno postavljana. Otkrivanje može imati formu pronalaženja značaja u vezama između određenih elemenata podataka, klasterisanja određenih elemenata podataka ili neki drugi

obrazac u korišćenju određenih skupova elemenata podataka. Nakon iznalaženja ovih obrazaca, algoritmi mogu da iz njih izvedu pravila. Ova pravila tada mogu biti korišćena da se generiše model koji ima željeno ponašanje, identifikuje veze među podacima, otkriva obrasce i grupiše klastere zapisa sa sličnim atributima.

Data mining se najtipičnije koristi za statističke analize podataka i otkrivanje znanja. Statističke analize podataka detektuju neuobičajene obrasce u podacima i primenjuju statističke i matematičke tehnike modelovanja da bi objasnile obrasce. Modeli se zatim koriste za progniziranje i predviđanje. Vrste statističkih analiza podataka sadrže linearne i nelinearne analize, regresivne analize, viševarijantne analize, analize u vremenu. Otkrivanje znanja izdvaja implicitne, prethodno poznate informacije iz podataka. Ovo često rezultuje u razotkrivanju nepoznatih poslovnih činjenica.

Data mining je *vođen podacima*. Postoji visok nivo složenosti u uskladištenim podacima i međusobnim vezama podataka u skladištu podataka koje je teško otkriti bez data mininga. Data mining nudi nove poglede na posao koji se ne mogu ostvariti sa upitima i izveštajima ili višedimenzionalnom analizom. Data mining može pomoći da ostvarimo nove poglede na posao dajući nam odgovore na pitanja koja nikad nismo mislili da postavimo.

Priprema podataka

U procesu razvoja skladišta podataka priprema podataka je jedna od najbitnijih aktivnosti. Dalji proces razvoja skladišta podataka biće uspešan samo ako je ova aktivnost uspešno završena.

Priprema podataka se vrši na osnovu ranije određenog izvora podataka, pravila za preuzimanje tih podataka, procedure pripreme i zahteva korisnika. Priprema se vrši određenim ekstrakciono-transformacionim alatima kroz sledeće korake:

- ekstrakcija i čišćenje podataka,
- transformacija podataka.

Rezultat ovih aktivnosti treba da budu podaci koji će nam omogućiti generisanje meta podataka, na osnovu kojih se može pristupiti dizajnu skladišta podataka.

Ekstrakcija i čišćenje podataka

Ova faza se sastoji od sledećih zadataka:

- razvoj procedura za ekstrakciju podataka,
- razvoj procedura za čišćenje podataka.

Razvoj procedura za ekstrakciju podataka

Podaci koji će se koristiti u skladištu podataka moraju se ekstrahovati iz transakcionih sistema (baza podataka u okviru nekog sistema) koji sadrže te podatke. Podaci se inicijalno ekstrahuju u procesu kreiranja skladišta podataka, a kasnije se na osnovu određnih procedura vrši dodavanje novih podataka u skladište podataka. Ekstrakcija podataka je vrlo jednostavna operacija, ako se potrebni podaci nalaze u jednoj relacionoj bazi, ali može da bude i veoma kompleksna operacija, ako su podaci smešteni u višestrukim heterogenim transakcionim sistemima. Cilj procesa ekstrakcije podataka je da sve potrebne podatke, u pogodnom i konzistentnom formatu, pripremi za učitavanje u skladište podataka.

Pre procesa ekstrakcije trebalo bi proveriti da li u bazi podataka iz koje vršimo ekstrakciju nema logičkih grešaka. Ovakve greške bi pre ekstrakcije trebalo ukloniti korišćenjem procedura za proveru grešaka.

Postoji mogućnost da se ne može utvrditi eventualno postojanje logičkih grešaka. To se dešava u situacijama kada se ekstrakcija vrši iz više izvora podataka. Prilikom ekstrakcije iz više izvora podataka može se javiti i problem nekonzistentnosti podataka usled različitog označavanja istih pojmova (nazivi država se mogu skraćeno označavati sa tri ili sa dva simbola).

Jedan od alata koji nam omogućava efikasnu ekstrakciju podataka je DTS (Data Transformation Services) koji je deo Microsoft SQL Server 2000 sistema za upravljanje bazama podataka.

Procedure za ekstrakciju podataka treba da izvršavaju sledeće aktivnosti:

- *Čitanje podataka iz prethodnih sistema ili prelaznih šema.* Prva prepreka pri ekstrakciji podataka je čitanje podataka koji su smešteni u starijim sistemima. U slučaju da su ti podaci dobro dokumentovani, onda je ovaj korak jednostavan jer se tada lako može utvrditi značenje svakog polja podataka. Međutim, vrlo često se dešava da se podaci mogu pročitati, ali se ne može utvrditi njihovo značenje. Najteža varijanta je da se podaci uopšte i ne mogu čitati. U tom slučaju jedino se može osloniti na izveštaje aplikacija koje koriste te podatke.
- *Utvrđivanje podataka koji se menjaju.* Veoma je bitno da se za vreme čitanja podataka iz starijih sistema utvrde oni podaci koji se menjaju jer se na taj način može smanjiti količina podataka koju treba prenositi u skladište podataka.
- *Kombinovanje različitih izvora za svaki ključ zapisa.* U najvećem broju slučajeva stariji sistemi se sastoje od većeg broja različitih datoteka. Zato se za dimenzione tabele često mora sprovoditi čitav proces denormalizacije tako da se podaci iz odvojenih datoteka mogu spojiti u jedan zapis.

Razvoj procedura za čišćenje podataka

Zbog problema koji se prilikom ekstrakcije podataka javljaju, podaci dobijeni ekstrakcijom se moraju "čistiti". Čišćenje podataka podrazumeva: proveru postojanja logičkih grešaka,

"poboljšanje" podataka i eliminisanje ostalih grešaka.

Provera logičkih grešaka uključuje:

- proveru vrednosti atributa,
- proveru atributa u kontekstu ostalih podataka u redu,
- proveru atributa u kontekstu redova druge tabele koja je povezana,
- proveru veza između redova iste ili povezanih tabela (provera prenesenih ključeva).

"Poboljšanje" podataka je proces čišćenja kojim se teži da podaci dobiju puno značenje. Primer za ovo su podaci o imenima i adresama. Često su ti podaci (npr. za jednog kupca) smešteni na više mesta u bazi i vremenom postaju nesihronizovani. Ovim procesom se teži da se takve situacije razreše.

Eliminisanje ostalih grešaka je proces u kome se odlučuje o sudbini podataka koji su nepotpuni ili nemaju veliko značenje. Ovi podaci se mogu odbaciti, privremeno smestiti i popraviti ili smestiti u skladište podataka sa tim svojim nesavršenostima.

Transformacija podataka

U ovoj fazi potrebno je definisati izvore podataka i tipove transformacija koje treba izvršiti nad podacima i ostvariti mapiranje podataka iz izvorišta u odredišta.

Pre početka procesa transformacije podataka, tim stručnjaka koji radi na projektu dizajniranja skladišta podataka definiše fizički model podataka za skladište podataka i generiše šeme. Taj tim stručnjaka se sastoji od poslovnih i tehničkih ljudi koji definišu strukturu skladišta podataka, analiziraju izvorne podatke, određuju način mapiranja podataka, prikupljaju ili kreiraju spoljne podatke, određuju logiku transformacije podataka i planiraju i generišu procedure transformacije podataka. Takođe, oni su odgovorni i za kvalitet dobijenih podataka. Pri tome biraju alate za migraciju, transformaciju i "čišćenje" podataka.

Faza mapiranja i transformacije podataka sastoji se od sledećih zadataka:

- kreiranje plana transformacije podataka,
- razvoj procedura za transformaciju podataka,
- razvoj procedura za učitavanje podataka,
- testiranje procedura,
- generisanje meta podataka.

U daljem tekstu detaljno će se obrazložiti svaka od gore definisanih faza.

Kreiranje plana transformacije podataka

Potrebno je da svi shvate zahtev za transformacijom podataka, kao i način na koji se ona izvodi. Planom je potrebno odrediti najbolji put migracije izvornih podataka do skladišta podataka. Pri tome se analiziraju raspoloživi resursi, količina izvornih podataka, različite izvorne šeme, različiti načini pristupanja podacima, struktura skladišta podataka i potreban broj agregacija. Planom se dokumentuju sve izvorne platforme, metode pristupa i programski jezik koji je potreban za ekstrakciju podataka.

Obično se izvorni podaci prvo smeštaju u prelazne šeme. Prelazne šeme su zajednički interfejs za sve izvorne sisteme. One se ne podudaraju u potpunosti ni sa izvornim ni sa odredišnim šemama. Koriste se da bi se poboljšali procesi "čišćenja" i transformacije podataka.

Nakon kreiranja plana transformacije podataka, prelazi se na analizu izvora podataka. Potrebno je odrediti koji će se podaci mapirati u odredišni sistem i koja je to logika potrebna da bi se izvršila migracija podataka.

Razvoj procedura za transformaciju podataka

Pod transformacijom podataka se podrazumeva proces kojim se usklađuju različiti načini prikazivanja podataka različitih sistema u jedinstveni oblik. Naprimer, neki sistemi mogu označavati pol ljudi sa 1 za muški pol i 2 za ženski pol. Ako se u skladištu podataka ovo označavanje vrši sa M i Z, onda mora postojati proces koji će transformisati 1 u M i 2 u Z. Proces transformacije podataka obuhvata i:

- rešavanje nekonzistentnih formata podataka izvornih sistema, kao što su ASCII i EBCDIC, različiti spelinzi, interpukcija itd.,
- obeležavanje nepoželjnih polja podataka koji nisu od značaja za analizu, kao što su oznake verzije i slično,
- prevođenje kriptovanih kodova u tekst koji je sastavljen od razumljivih reči,
- označavanje normalnih, nenormalnih, nemogućih činjenica i činjenica koje su van dozvoljenih granica.

Transformacija podataka je kritičan korak u razvoju skladišta podataka. U okviru procesa transformacije vrši se poslednja priprema podataka pre učitavanja. Proces transformacije može da se uradi i neposredno pre učitavanja podataka u skladište, korišćenjem DTS alata. Tipična transformacija podataka uključuje:

- prevođenje polja sa više imena u jedno polje,
- razbijanje,
- polja sa datumom u posebna polja za godinu, mesec i dan,

- prevođenje polja sa jednom reprezentacijom u drugu (npr. sa 1 i 0 u DA i NE),
- kreiranje i dodavanje ključeva za tabele dimenzija.

Razvoj procedura za učitavanje podataka

Procedure za učitavanje podataka treba da izvršavaju sledeće aktivnosti:

- Kreiranje formata podataka. Za sve podatke iz starijih sistema moraju se obezbediti formati pogodni za smeštanje u skladište podataka.
- Prenošenje podataka iz starijih sistema u skladište podataka. Nakon kombinovanja izvora podataka i kreiranja odgovarajućih formata podataka, prelazi se na smeštanje podataka u skladište podataka. Pri ovome, metode pristupa i transformacije podataka često moraju da vrše raspakivanje podataka, njihovo poređenje, kombinovanje i transformaciju u oblik pogodan za skladište podataka.
- Kreiranje agregacija. Kreiranje agregacija je postupak sortiranja podataka po određenim atributima na osnovu kojih se, zatim, vrši sumiranje. Tako sumirani podaci se smeštaju u skladište podataka.
- Kreiranje ključeva za agregacione zapise. Svi zapisi u tabelama, a samim tim i agregacije, moraju imati ključeve. Ovaj korak se razlikuje od prethodnog jer su ključevi za agregacione zapise u potpunosti veštački i ne smeju biti identični primarnim ključevima tabele činjenica. Prema tome, stručni tim mora dizajnirati aplikaciju koja će generisati takve ključeve.
- Obrada neučitanih podataka. Pri procesu smeštanja podataka u skladište podataka često se dešava da se neki podaci ipak ne učitaju, najčešće zbog referencijalnog integriteta. Takvi podaci se moraju obraditi u posebnoj aplikaciji, koja će obezbeđivati referencijalni integritet podataka.
- Indeksiranje podataka. Po završenom procesu smeštanja podataka u skladište podataka, svi indeksi se moraju ažurirati.

Testiranje procedura

Da bi se utvrdila ispravnost rada procedura za ekstrakciju i učitavanje podataka, mora se izvršiti njihovo testiranje. To se, najčešće, ostvaruje proverom kvaliteta podataka, tako što se zadaju upiti nad skladištem podataka koji prebrojavaju podatke ili ih prikazuju u vidu grafikona sa kojih se može utvrditi da li su podaci u rasponu koji je očekivan.

Po završenoj transformaciji, postoje svi uslovi da se pristupi generisanju meta podataka.

Izrada meta baze podataka

Upravljanje složenim distribuiranim IS zahteva i razvoj meta baze podataka, odnosno rečnika podataka. Meta baza podataka je baza podataka o bazi podataka.

Meta baza podataka čuva sve podatke o podacima mapirajući izvorni i ciljni sistem i uspostavlja vezu između podataka sa izvora i cilja. Oni čuvaju informacije o transakcionim podacima, definiciju podataka u ciljnoj bazi i transformaciono-integracionu logiku.

Tek po postavci meta baze podataka može se krenuti dalje u izdvajanje podataka iz transakcione baze podataka, pa potom sumiranje, sortiranje i organizovanje pre punjenja DW.

Izgradnja skladišta podataka

Izgradnja "skladišta" podataka nije samo prosto kopiranje podataka i prepuštanje korisnika alatima za podršku odlučivaju već pretpostavlja i restrukturiranje podataka denormalizacijom tabela, čišćenjem podataka od redundansi i nelogičnosti i dodavanjem novih polja i ključeva radi zadovoljenja korisnikovih potreba za sortiranjem, kombinovanjem i sumiranjem podataka.

Da bi se izvršili složeni upiti, skladištenje često uključuje i preračunavanje sumarnih podataka, kao i predefinisanje pogleda u bazi. U skladišta se uključuju i podaci iz eksternih izvora, kao i trendovi, prognoze i procene, na osnovu kojih se izvršavaju simulacije čiji rezultati predstavljaju dragocenu podršku za donošenje strateških odluka.

Prvi korak je da se izvrši identifikacija dimenzija i atributa. Identifikacija dimenzija i atributa podseća na klasično projektovanje upotrebom ER modela i zove se dimenziono modeliranje.

Dimenziono modeliranje je tehnika logičkog dizajna čiji je cilj prezentacija podataka u obliku koji obezbeđuje visoke performanse sistema radi vršenja analize podataka.

U dimenzionom modeliranju, strukture podataka su tako organizovane da opisuju mere i dimenzije. *Mere* su numerički podaci smešteni u centralnoj, takozvanoj tabeli činjenica (fakt tabela). *Dimenzije* su standardni poslovni parametri koji definišu svaku transakciju. Dimenzije se smeštaju tabele neposredno, ili preko druge tabele dimenzije, povezane sa tabelom činjenica.

Osnovu za izradu dimenzionog modela predstavljaju *meta podaci*, na osnovu kojih se vrši definisanje hijerarhija, elemenata i atributa, normalizacija i denormalizacija i definisanje agregacija.

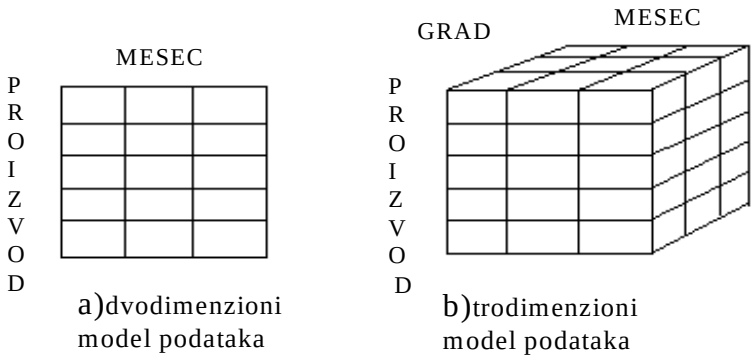
Svaka dimenziona tabela ima svoj primarni ključ, a svi oni učestvuju u stvaranju primarnog ključa tabele činjenica. Ovakvi modeli se nazivaju šemama zvezde. Tabele činjenica sadrže podatke koji su, najčešće, numeričkog tipa i mogu sadržati veliki broj zapisa. Dimenzione tabele sadrže opisne tekstualne informacije. Atributi dimenzionih tabela se koriste kao najčešća ograničenja pri zadavanju upita.

Dimenzioni modeli su standardnog oblika te se mogu predvideti interfejsi koji će biti od koristi korisnicima skladišta podataka. Dimenzioni modeli se jednostavno proširuju dodavanjem novih dimenzija i njihovih atributa i pri tome se nijedan alat za izveštavanje ili upite ne mora menjati. Sve je više pomoćnih programa i alata koji upravljaju i rade sa agregacijama i na taj način još

više poboljšavaju performanse sistema.

Izgradnja skladišta podataka je iterativni postupak. Čim se određena količina podataka smesti u skladište podataka, korisnici mogu da im pristupaju i da zaključe koje su im koristi od toga. Nakon toga, oni mogu da zadaju nove zahteve zbog kojih će se morati uneti neke izmene u modelu. Fizička arhitektura dimenzionog modela je šema zvezde, o čemu će kasnije više biti reči.

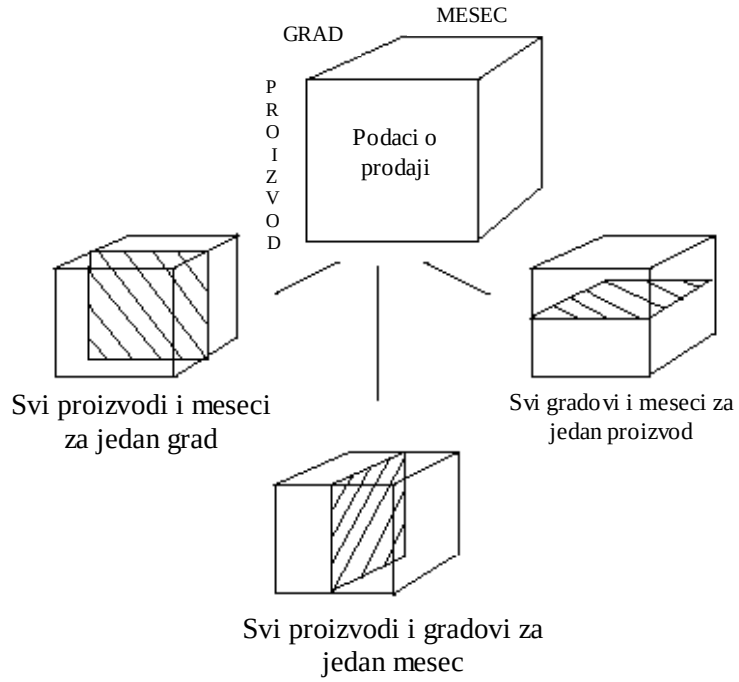
Na sledećoj slici upoređeni su načini prikazivanja podataka u dvodimenzionom i trodimenzionom modelu podataka.



Slika 54. Primeri dvodimenzionih i trodimenzionih modela podataka

Kao što se vidi na prethodnoj slici, u slučaju pod a) podaci o prodaji za svaku oblast se nalaze u različitim tabelama, dok su u slučaju pod b) svi podaci smešteni u trodimenzioni niz.

Iako su podaci sačuvani samo jednom i to na jednom mestu, svaki korisnik može dobiti različite poglede na jedne iste podatke. Jedan od primera dat je na sledećoj slici.

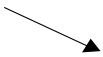


Slika 55. Različiti pogledi na iste podatke

Dimenzije se često mogu organizovati u hijerarhije, kao što je DAN –> NEDELJA –> MESEC. One omogućavaju da korisnik posmatra podatke sa manje ili više detalja.

Sve dimenzione tabele su denormalizovane, što znači da se isti podaci čuvaju na više mesta da bi se obezbedila jednostavnost i poboljšale performanse. Primer normalizovane i denormalizovane reprezentacije podataka dat je na sledećoj slici.

Sifra_Proizvoda	Ime_Proizvoda	Boja_Proizvoda	Sifra_Imena
101	N1	B1	XYZ
102	N2	B2	XYZ
103	N3	B3	ABC
104	N4	B4	ABC



Sifra_Imena	Ime
XYZ	M. Markovic
ABC	P. Petrovic

a) normalizovana reprezentacija

Sifra_Proizvoda	Ime_Proizvoda	Boja_Proizvoda	Sifra_Imena	Ime
101	N1	B1	XYZ	M. Markovic
102	N2	B2	XYZ	M. Markovic
103	N3	B3	ABC	P. Petrovic
104	N4	B4	ABC	P. Petrovic

b) denormalizovana reprezentacija

Slika 56. Različiti načini reprezentacije podataka

Na osnovu gore rečenog, izgradnja skladišta podataka se sastoji od sledećih zadataka:

- denormalizacija podataka,
- definisanje hijerarhija,
- kreiranje agregacija,
- kreiranje fizičkog modela,
- generisanje baze podataka,
- učitavanje podataka.

Denormalizacija podataka

U zavisnosti od predstave dimenzija na modelu, govorimo o normalizovanom ili denormalizovanom modelu. Kod denormalizovanog modela dimenzije su organizovane u šemu zvezde, a kod normalizovaog u šemu snežne pahuljice.

Postoje situacije u kojima šema zvezde nije pogodna za skladištenje podataka. Osnovni razlozi za to su:

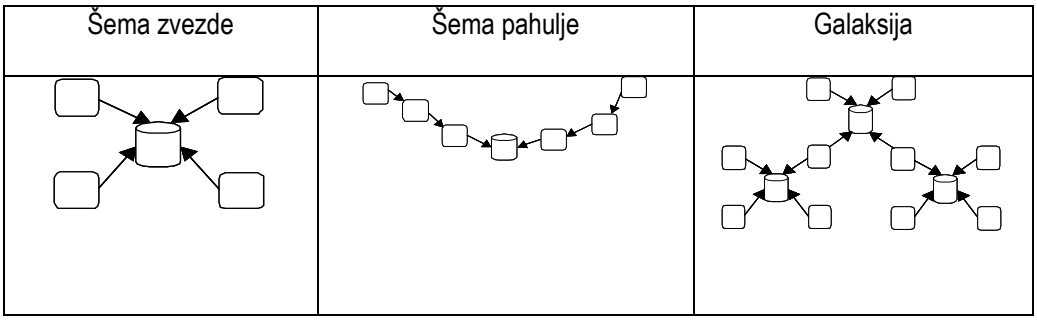
- denormalizovana šema zvezde može zahtevati previše memorijskog kapaciteta,

- veoma velike dimenzione tabele mogu uticati na pad performansi sistema.

Ovi problemi se mogu rešiti normalizacijom dimenzija. Time se šema zvezde prevodi u šemu pahulje. Glavni nedostatak šeme pahulje je njena složenost u odnosu na šemu zvezde, čime se otežava održavanje skladišta podataka. Zato je potrebno vršiti normalizaciju samo onih dimenzija koje sadrže mnogo redova podataka i koje imaju mnogo atributa. Najčešće se postižu najbolji rezultati ako se izvrši normalizacija samo par dimenzija, a da se ostale ostave onakve kakve su i bile. Na taj način se dolazi do delimične šeme pahulje.

Osnovna karakteristika šeme pahulje jeste da se ne vrši denormalizacija dimenzionih tabela, čime se poboljšavaju performanse sistema. Neke dimenzione tabele mogu sadržati veliki broj podataka, pri čemu se često dešava pojava redundantnosti, te se normalizacijom može znatno smanjiti broj podataka. Takođe, šema zvezde obezbeđuje najbolje performasne kada se radi sa agregacionim podacima. Nedostatak šeme pahulje je što se moraju kreirati dodatne veze, koje pri procesiranju upita mogu pogoršati performanse sistema. Takođe, održavanje šeme pahulje je relativno složeno s obzirom da u bazi podataka postoji veći broj tabela i da meta podaci više nisu jednostavni. Jedino se uporednim testovima može utvrditi da li je bolje koristiti šemu zvezde ili šemu pahulje.

Šema galaksije predstavlja kolekciju šema zvezda, tj. ako se ne može kreirati model koji bi imao samo jednu činjeničnu tabelu, tada je potrebno povezati dve šeme zvezde da bi se zadovoljile potrebe korisnika.



Slika 57. Šeme zvezde, pahulje i galaksije

Pri dizajniranju baze podataka najčešće se koristi šema zvezde. Ona se sastoji od relativno malog broja tabela sa dobro definisanim vezama. Šema zvezde polako postaje standard za izradu skladišta podataka zbog svojih prednosti u odnosu na ostale relacione strukture:

- obezbeđuje kraće vreme odziva na upit jer se smanjuje broj fizičkih veza između tabela,
- model je jednostavan i lako se mogu vršiti modifikacije,
- pojednostavljuje razumevanje i navigaciju meta podataka,

- održavanje je relativno jednostavno,
- proširuje skup alata koji se mogu koristiti za rad sa podacima.

Fizička arhitektura dimenzionog modela opisana je pomoću šeme zvezde definisane sa dve vrste tabela – dimenzione tabele (dimension table) i tabele činjenica (fact table)

Tabela činjenica sadrži kvantitativne podatke o poslovima, tj. podatke koje korisnici analiziraju. Ovi podaci su najčešće numeričkog tipa i mogu se sastojati i od nekoliko miliona redova i kolona.

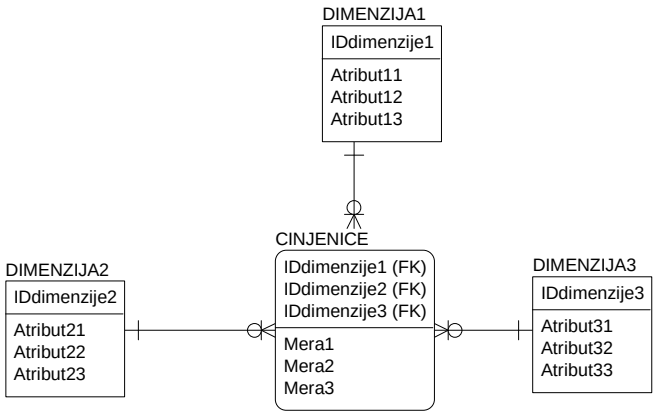
Dimenzione tabele su znatno manje i sadrže podatke koji opisuju dati posao, tj. one podatke po kojima se vrši analiziranje. Ti podaci se nazivaju *atributi*.

Osnovne prednosti šeme zvezde su što omogućava definisanje složenih višedimenzionih podataka u vidu jednostavnog modela, smanjuje broj fizičkih veza koje se moraju procesirati pri zadavanju upita, čime se postiže poboljšanje performansi sistema i omogućava proširenje skladišta podataka uz relativno jednostavno održavanje. Velika mana šeme zvezde je što se povećava redundantnost podataka.

Osnovna karakteristika šeme zvezde jeste da su dimenzione tabele denormalizovane. Denormalizacija je pristup gde se podaci u bazi podataka ponavljaju zbog pojednostavljenja dizajna i karakteristika. Denormalizacija je proces kombinovanja tabela da bi se poboljšale performanse sistema. Ovim postupkom se smanjuje broj potrebnih veza koje se moraju procesirati zadavanjem upita. Time se direktno utiče na poboljšanje performansi sistema, jer što je manji broj veza, to sistem brže nalazi tražene podatke.

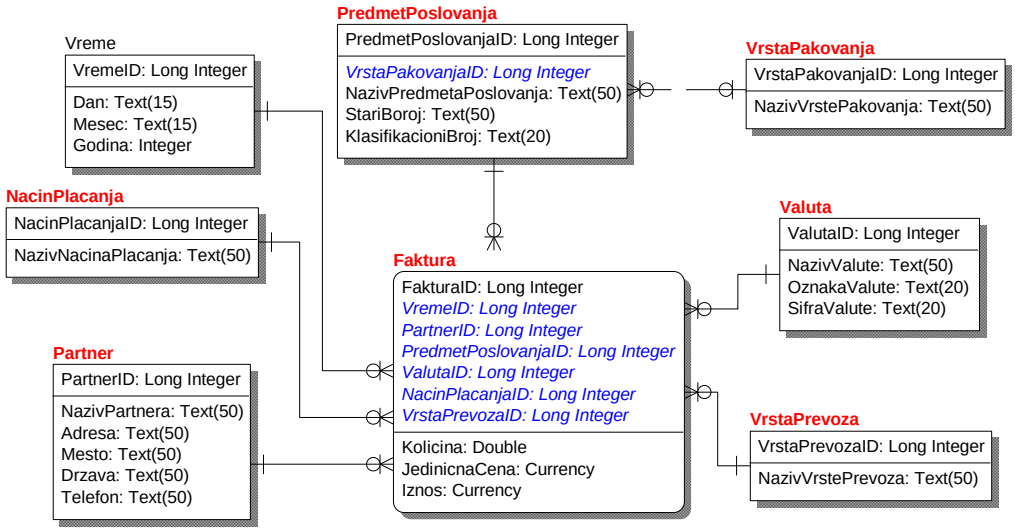
Prema tome, dimenzioni atributi mogu biti smešteni više puta u dimenzione tabele, u zavisnosti od toga koji nivo dimenzione hijerarhije atribut opisuje.

Svaka tabela mora sadržati primarni ključ koji predstavlja kolonu ili grupu kolona u tabeli čiji sadržaj jedinstveno identifikuje zapise. Na sledećoj slici dat je izgled jednostavne šeme zvezde.



Slika 58. Jednostavna šema zvezde

Na slici se vidi da je primarni ključ tabele činjenica sastavljen od tri spoljna ključa. Spoljni ključ je kolona jedne tabele, čija je vrednost definisana kao primarni ključ druge tabele. Na sledećoj slici prikazana je šema zvezde na primeru EDIFACT fakture.



Slika 59. Šema zvezde na primeru EDIFACT fakture

Dimenzione tabele mogu, takođe, sadržati i spoljne ključeve, koji referenciraju primarne ključeve drugih dimenzionih tabela. Takve tabele se nazivaju sekundarne dimenzione tabele (outrigger tables). One ne mogu biti u direktnoj vezi sa činjeničnim tabelama. Na slici dat je primer upotrebe

sekundarnih dimenzionih tabela – VrstaPakovanja, koje definišu kodove korišćene u tabeli PredmetPoslovanja.

Definisanje hijerarhija

Dimenzione tabele memorišu sledeće elemente:

- traženje hijerarhijskih relacija u svakoj dimenziji,
- definisanje opisnih atributa svake dimenzije.

Dimenzije veoma često mogu biti organizovane u hijerarhiji. Svaki hijerarhijski nivo se nastavlja sa nekim drugim hijerarhiskim nivoom. Naprimer, unutar vremenske dimenzije, dani se nastavljaju na nedelje, koji se nastavljaju na kvartale. Za dimenziju proizvoda vezuje se proizvodna grupa, koja se nastavlja na proizvodne vrste.

Ovakva povezivanja mogu biti veoma složena, kao naprimer nastavljanje nedelja na mesece. Pošto se meseci ne mogu jednako podeliti na nedelje, to se i nedelje ne mogu nastaviti na mesece, međutim i nedelje i meseci se mogu nastaviti sa kvartalima.

Dimenzioni elementi su specijalna kategorija podataka, koja predstavlja određeni nivo u dimenzionoj hijerarhiji. Za svaki hijerarhijski nivo postoji po jedan dimenzioni element. Naprimer, kod dimenzije proizvod, mogu postojati tri dimenziona elementa: prozvod, grupa i vrsta proizvoda. U ovom modelu možemo reći da dimenzioni element "proizvod" predstavlja najniži hijerarhijski nivo u dimenziji proizvod, dok vrsta proizvoda predstavlja najviši nivo.

Posmatranje podataka iz različitih, ali blisko povezanih perspektiva omogućava da korisnik analizira podatke na različitim nivoima detalja. Postupak prelaska sa nivoa sa manjim brojem detalja na nivo sa većim brojem detalja naziva se spuštanje u dubinu (drill down) i predstavlja zahtev korisnika da mu se prikaže više detalja.

Postupak prelaska sa nivoa sa većim brojem detalja na nivo sa manjim brojem detalja, na tzv. sumarne podatke, naziva se dizanje naviše (drill up). Naprimer, upit bi mogao prezentovati prodaju u odnosu na neke regione. Pošto pronađemo vrh prodaje u nekom regionu, spuštamo se naniže da bi smo saznali kako se prodaja odvija po opštinama. Dizanje naviše je suprotno od spuštanja nadole i zahteva zbirni pogled na podatke. Dobro dizajnirana šema zvezde mora obezbediti postojanje različitih nivoa detalja, tj. hijerarhija.

Naprimer, geografski podaci vezani za prodaju mogli bi se organizovati u sledeću hijerarhiju:

SVET –> KONTINENT –> DRŽAVA –> OBLAST –> GRAD

Pored operacija drill down i drill up, postoji i operacija drill across, koja se koristi za povezivanje dve ili više činjeničnih tabela na istom nivou hijerarhije.

Kreiranje agregacija

Agregacija je proces skupljanja činjeničnih podataka po unapred definisanim atributima. Naprimer, moguće je kreirati sumarne podatke o prodaji po regionu i oblasti skupljajući ih iz svake prodavnice, tj. najnižeg nivoa detalja. Agregacijama se sumiraju detalji podataka i smeštaju u posebne tabele. Ove tabele se koriste od strane aplikacija da bi se eliminisala potreba da se ponovo vrše neki proračuni koji bi se inače morali sprovesti ako ove tabele ne bi postojale.

Glavni razlozi kreiranja agregacija su da se poboljšaju performanse upita, tj. da se smanji vreme odziva na upit, kao i da se smanji broj resursa potrebnih za izvršenje upita. Pri kreiranju agregacija mora se voditi računa o tome koje bi zaista trebalo da postoje. Nije dobra praksa da se kreiraju agregacije koje obrađuju podatke nekoliko sati, a da se koriste jednom godišnje. S druge strane, veoma je dobro kreirati agregaciju koju upotrebljavaju skoro svi korisnici i to vrlo često.

Tipično skladište podataka sadrži podatke atomskog nivoa. Sve mere se smeštaju u tabele činjenica tako da se kasnije mogu koristiti za potrebe analiziranja. Međutim, preuzimanje podataka atomskog nivoa iz skladišta podataka ne obezbeđuje optimalne performanse. Tabele činjenica mogu biti vrlo velike te izvođenje operacija nad podacima atomskog nivoa smeštenih u njima može vremenski trajati dugo. Međutim, najveći broj upita zadatih nad skladištem podataka odnosi se na sumiranje (agregaciju) podataka. Naprimer, tipični korisnik će često postaviti zahtev da mu se prikaže ukupna prodaja za ceo mesec. Ovaj zahtev bi se u bazi podataka interpretirao kao potreba da se saberu svi podaci vezani za prodaju i koji postoje za svaki dan tog meseca. Ako bi, naprimer, tokom jednog dana postojalo 1000 transakcija u svakoj od 1000 prodavnica, onda bi ovaj upit morao da procesira 30 miliona redova da bi se dobio odgovor. Ovakav upit bi znatno trošio sve raspoložive resurse. Za podatke kojima se češće pristupa poželjno je izvršiti sumiranje. Time se omogućava da se već postojeći sumarni podaci mogu odmah koristiti, čime se znatno smanjuje vreme odziva na upit koji treba da procesira te sumarne podatke. Naprimer, ako bi postojala tabela u kojoj bi se čuvali sumarni podaci o prodaji za svaku od 1000 prodavnica, onda bi upit o ukupnoj prodaji za ceo mesec morao da procesira 1000 redova. Prema tome, postojanje tabele sa sumarnim podacima, u ovom primeru, smanjuje potrebu procesiranja 30 miliona redova podataka na hiljadu.

S obzirom da mnogi upiti koje postavlja korisnik mogu zahtevati agregaciju stotina hiljada redova, vršenje agregacija unapred može značajno da smanji vreme odziva na upit. Upotrebom agregacija smanjuje se vreme odziva na upit, ali se istovremeno i povećava sama baza podataka.

Prema tome, može se zaključiti da je kreiranje unapred definisanih agregacija neophodno da bi se omogućio rad sa velikim brojem podataka. Dinamičke agregacije, tj. agregacije koje vrši korisnik za vreme rada sa skladištem podataka su najčešće dugotrajne, te su za potrebe odlučivanja neprihvatljive.

Agregacije zasnovane na SQL naredbama

Jedan od načina na koji se mogu kreirati agregacije jeste korišćenje SQL naredbi. Iako ovaj način nije najbolji po pitanju performansi sistema, on je najjednostavniji, što se najlakše pokazuje na primeru. Neka se u jednom skladištu podataka primarni ključ tabele činjenica sastoji od spoljnih ključeva Proizvod_Id, Vreme_Id i Prodavnica_Id. Da bi se odredila agregacija podataka o proizvodima po podgrupama, može se zadati sledeća SQL naredba:

```
SELECT PodgrupaID, ProdavnicaID, VremeID,
FROM fact_tabela
WHERE ProizvodID = PodgrupaID
GROUP BY PodgrupaID, ProdavnicaID, VremeID
```

Agregacije koje nisu zasnovane na SQL naredbama

U slučaju kreiranja agregacija koje nisu zasnovane na SQL naredbama, potrebno je razviti specijalizovane programe, što usložnjava procese razvoja i održavanja skladišta podataka.

Prednosti ovog načina kreiranja agregacija su:

- Može se izvršiti agregacija dimenzije jednim prolazom po podacima.
- Sama priroda procesa agregacije je takva da se može dekomponovati na više paralelnih procesa.

Ukratko, proces se sastoji u traženju redova podataka koje treba agregirati, zatim sortiranju datoteke, kreiranju podzbirova, a potom agregaciji i učitavanju tokom jednog prolaza kroz datoteku. Po nalaženju redova podataka koje treba agregirati, izvrši se sortiranje po dimenziji po kojoj se traži agregacija. Na taj način će se svi podaci istog nivoa dimenzije nalaziti jedan iza drugog. Naprimer, ako se izvrši sortiranje redova podataka po dimenziji Vreme, u tabeli će se prvo nalaziti redovi podataka koji se odnose na Dan, iza njih će biti redovi podataka koji se odnose na Nedelju itd. Zatim se na svakom mestu prelaza sa jednog nivoa dimenzije na drugi (naprimer, sa Dana na Nedelju) kreiraju podzbirovi za taj nivo dimenzije. Pri tome je moguće iskoristiti prednosti paralelnog procesiranja jer su podaci podeljeni po grupama (jedan proces može računati podzbirove vezane za nivo Dan, a drugi za nivo Nedelja). Tako dobijene podzbirove treba učitati i izvršiti agregaciju. Time je proces agregacije podataka završen.

Kreiranje fizičkog modela

U okviru kreiranja fizičkog modela baze podataka, izvodi se postupak prevođenja logičkog modela u fizički model prikazan preko dijagrama entiteti – veze koji fokusira podatke. Fizički model za potrebe našeg skladišta podataka biće orijentisan relacionim bazama podataka i koristiće se za kreiranje šeme baze podataka.

Treba naglasiti da generisanje fizičkog modela mora da ispuni zahteve vezane za strukturna

dinamička pravila integriteta, i to:

- ograničenja, kojima se definišu dozvoljena stanja baze podataka,
- operacije, koje mogu potencijalno ugroziti ograničenja,
- akcije, koje treba preduzeti ukoliko dođe do narušavanja ograničenja.

Za kreiranje fizičkog modela koristitće se CASE alat ERWin, koji omogućava ostvarivanje veze između konceptualnog (logički), dimenzionog i fizičkog modela.

Neposredno pre kreiranja modela treba izabrati sistem za upravljanje bazama podataka na kome će biti implementirana baza podataka. Koristićemo Microsoft SQL Server 2000, jer ima mnoge alate i osobine koje pojednostavljuju proces instaliranja, razvoja, upravljanja i korišćenja baza podataka. SQL Server 2000 sadrži alate koji omogućavaju vezu sa Internetom, ima integrisan sistem zaštite sa Windows NT i Windows 2000. Što je najbitnije, SQL Server 2000 sadrži alate koji olakšavaju rad sa skladištima podataka. Od alata za rad sa skladištima podataka ima alate za ekstrakciju i transformaciju podataka (DTS), za OLAP (On-line Analiytical Processing) analizu (OLAP server), a uključuje i alate za vizuelni dizajn skladišta podataka.

Na slici je prikazan model nastao prevođenjem logičkog modela u fizički model. U okviru procesa prevođenja klasa u entitete trebalo je rešiti probleme:

- multiplikativnosti,
- referencijalnog integriteta i
- kreiranja indeksa.

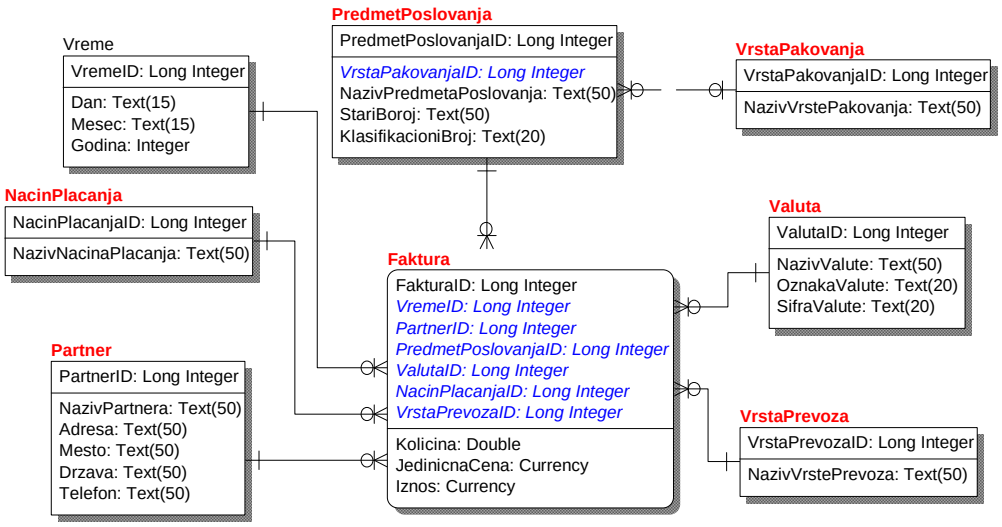
Multiplikativnost definiše broj instanci jednog entiteta (buduća tabela u bazi) u relaciji sa jednom instancom drugog entiteta. Problem multiplikativnosti je rešen direktnim preuzimanjem ovih parametara sa modela objekti – veze, definisanog ranije.

Referencijalni integritet se praktično reflektuje kao postojanje prenesenog ključa u nekoj tabeli. Tamo gde se nalazi preneseni ključ, postoji "ciljna tabela", a tamo gde je definisan primarne ključ, nalazi se "izvorna" tabela. Referencijalni integritet tabele zahteva da unesena vrednost atributa odgovara vrednosti atributa koji je primarni ključ druge tabele. Referenacijalni integritet se definiše za operacije ubacivanja, brisanja i ažuriranja.

Važno je, kod transformacije iz dijagrama klasa u fizički model, istaći da je pravilo da se veze kompozicije i agregacije transformišu u identifikujuću vezu između tabela. U ovom slučaju je odstupljeno od ovog pravila jer se zahtevala fleksibilnost fizičkog modela, zbog specifičnih upita koji su kasnije rađeni nad skladištem podataka.

Kreiranje indeksa je izvršeno automatski za sve primarne ključeve u entitetima i za prenesene ključeve u entitetu Ispit. Ovo se radi iz razloga što će se buduća pretraživanja u okviru skladišta podataka vršiti na osnovu ovih polja.

Na sledećoj slici je prikazan fizički model šeme zvezde za primer EDIFACT fakture.



Slika 60. Fizički model šeme zvezde za primer EDIFACT fakture

Generisanje baze podataka

Aktivnost generisanja baze podataka vrši se korišćenjem SQL jezika. Naime, alat u kome je izvršeno kreiranje fizičkog modela (npr. ERWin) omogućava automatsko generisanje koda preko takozvanih DDL (Data Definition Language) datoteka.

U sledećem koraku se vrši izvršavanje DDL datoteka pomoću Query Analyzer-a, alata koji je sastavni deo SQL Servera 2000. Ovaj alat omogućava direktno zadavnje SQL naredbi i njihovo izvršavanje u cilju generisanja baze podataka.

Treba napomenuti da se prilikom generisanja baze mora doneti odluka o načinu mapiranja operacija koje će biti na raspolaganju u okviru fizičke baze podataka. Način koji je u ovom slučaju primenjen jeste da se operacije mapiraju u obliku okidača (kod okidača je dat, u okviru SQL koda za generisanje baze, u prilogu).

Kada se svi ovi poslovi uspešno urade, baza (skladište) podataka je generisana.

Učitavanje podataka

Pošto su završene sve pripreme, može se pristupiti učitavanju podataka u skladište podataka. U toku učitavanja se mogu eventalno izvršiti još neke transformacije, mada bi sa transformacijama podataka trebalo završiti pre učitavanja zbog problema konzistentnosti baze. Za učitavanje

podataka može se koristiti alat MS SQL Server-a DTS (Data Transformation Servicess) i njegova procedura učitavanja podataka pomoću takozvanih DTS paketa.

Primena skladišta podataka

Primena skladišta podataka je širokog dijapazona, počev od državnih organa, zdravstva i obrazovanja, pa do finansija, prodaje, marketinga, nabavke i proizvodnje.

Primena skladišta podataka u *finansijama* vezana je za, npr.:

- izveštaj o protoku novca po grupama proizvoda i organizacionim jedinicama koji omogućuje analizu prihoda i troškova po velikom broju parametara;
- detaljnu analizu profita, multidimenzione izveštaje po kategorijama profitnih i troškovnih centara;
- izradu bilansnih računa;
- izradu sumarnih izveštaja o ključnim finansijskim parametrima.

Primena skladišta podataka u *marketingu* vezana je za izradu strateških marketinških analiza koje se implementiraju putem multidimenzionih izveštaja, sa detaljima o prodaji viskoprofitnih proizvoda i praćenjem po vremenu, regionima, distributerima, sektorima potrošnje i dr. Mogućnost slobodnog izvlačenja sumarnih izveštaja i zalaženja u detalje u područjima koja ne ispunjavaju očekivanja daje sektoru marketinga uvid u promenu na tržištu. Kako su marketinške kampanje skupe, to arhiviranje podataka o tržištu daje odgovor na pitanje koji je segment tržišta reagovao, preko kojih kanala, na kom geografskom području i preko kojih medija. Ovi podaci su od neprocenjivog značaja za organizovanje narednih kampanja.

Primena skladišta podataka u *prodaji* vezana je za analizu prodaje i davanje odgovora na pitanja kao što su:

- Koji proizvod donosi najveći profit?
- Koji kupci kupuju najprofitabilnije proizvode?

Primena multidimenzionih izveštaja dovodi do uočavanja Pareto pravila, pokazujući odnos između proizvoda, profatibilnosti i geografske distribucije (npr. 20% proizvoda koji donose 80% profita). Precizna analiza može biti osnova za stimulaciju prodavaca (stimulacija po ostvarenom profitu a ne po prihodu) i za planiranje prodaje.

Primena skladišta podataka u *nabavci* veoma je važna jer nabavka određuje profitabilnost preduzeća na više načina. Pre svega, to su troškovi materijala, praćenje odnosa sa dobavljačima kroz vreme. Još su veće mogućnosti upravljanja zalihama, u smislu oslobađanja obrtnog kapitala zarobljenog u zalihama.

Upravljanje odnosima sa dobavljačima je posebno važno u svetlu JIT (Just in Time) proizvodnje. Uz dobro planiranje omogućuje se dobro analitičko praćenje prformansi dobavljača, tačno ispunjenje dogovorenih rokova, kvaliteta i drugih parametara, kroz koncept kartice rezultata dobavljača.

Primena skladišta podataka u *proizvodnji* vezana je za davanje odgovora na pitanja kao što su:

- Koliko vremena treba da se napravi proizvod A?
- Gde je usko grlo proizvodne linije?
- Gde se javlja najviše problema vezanih za kvalitet?

Analiza upravljanja kapacitetima rešava probleme u operativnom planiranju i otvara mogućnost za "provlačenje" proizvodnje kroz ograničene kapacitete bez većih investicija.

I ovde se može uključiti Pareto analiza, tj. 80% problema potiče od 20% uzroka, što je posebno važno za praćenje kvaliteta proizvoda. Da bi se problem kvaliteta mogao dobro analizirati i u transakcionim sistemima i u skladištu podataka, potrebno je obezbediti potpunu sledljivost proizvodnog procesa, od ulaznih sirovina do gotovog proizvoda.

Primena skladišta podataka u upravljanju *kadrovima* vezana je za analizu i planiranje razvoja kadrova i predstavlja prvi korak u uvođenju upravljanja znanjem kao novog kvaliteta i prednosti organizacija u tržišnoj utakmici.

Skladišta podataka se mogu primeniti i za praćenje i analizu zarada i troškova (po grupama, organizacionim jedinicama, regionima), efikasno ulaganje u obrazovanje kadrova i dr.

OLAP sistemi

Interaktivno analitičko procesiranje (On line Analytical Processing – OLAP) namenjeno je on line analizama i izveštavanjima, za razliku od produkcionih sistema namenjenih ažuriranju baza podataka i obradi transakcija (On Line Transaction Processing – OLTP).

Postavlja se pitanje: šta je to krajnjem korisniku potrebno? Ono što krajnjem korisniku treba je sledeće:

- da može da postavi bilo koje poslovno pitanje,
- da bilo koji podatak iz preduzeća koristi za analizu,
- mogućnost neograničenog izveštavanja.

Donosiocima poslovnih odluka su potrebni odgovori na pitanja koji direktno utiču na njihovu mogućnost da budu kompetentni na današnjem brzo promenljivom tržištu. Njima su potrebni jasni odgovori na koliko god teška pitanja, i to u što kraćem vremenskom periodu. U tu svrhu se

koriste analitički OLAP (on line analytical processing) sistemi koji obezbeđuju informacije koje se koriste za analizu problema ili situacija. Analitičko procesiranje se primarno vrši korišćenjem poređenja ili analiziranjem šablona i trendova. Naprimer, analitički sistem bi mogao da prikaže kako se određena vrsta štampača prodaje u različitim delovima zemlje. Takođe, mogao bi da prikaže i kako se jedna vrsta proizvoda prodaje sada u odnosu na period kada se proizvod prvi put pojavio na tržištu.

Analiziranje šablona podataka i trendova zahteva postojanje velikog broja istorijskih podataka. Zato analitičke baze podataka ne sadrže ažurne podatke, već čuvaju informacije iz određenog trenutka vremena. Naprimer, moguće je utvrditi da je prodaja u jednom mesecu znatno opala samo ako u sistemu postoje podaci o prodaji u prethodnim mesecima, tako da se može vršiti poređenje.

U početku su upiti korisnika bili relativno jednostavni. Međutim, vremenom su korisnički upiti postali toliko složeni da relacioni alati (OLTP alati) nisu bili u mogućnosti da daju odgovore u prihvatljivom vremenskom periodu. Upravo u tu svrhu se koriste OLAP sistemi. Oni omogućavaju jednostavnu sintezu, analizu i konsolidaciju podataka. Koriste se za intuitivnu, brzu i fleksibilnu manipulaciju transakcionim podacima. OLAP sistemi podržavaju kompleksne analize koje sprovode analitičari i omogućavaju analizu podataka iz različitih perspektiva (poslovnih dimenzija).

OLAP sistemi kao skladišta podataka koriste multidimenzionalnost i denormalizaciju i može se reći da predstavljaju nadgradnju skladišta podataka. U sledećoj tabeli date su neke uporedne karakteristike OLTP sistema, skladišta podataka i OLAP sistema.

Tabela 3

Karakteristike	OLTP sistemi	Skladište podataka	OLAP sistemi
Tipične operacije	ažuriranje	izveštavanje	analiza
Analitički zahtev	nizak	srednji	visok
Ekrani	nepromenljivi	definiše korisnik	definiše korisnik
Količina podataka po transakciji	mala	srednja	velika
Nivo podataka	detalji	detalji i sumarni podaci	sumarni podaci
Starost podataka	tekući podaci	tekući i istorijski podaci	tekući, istorijski i projektovani podaci
Namena	transakcionim podacima	rad sa istorijskim podacima	analiza
Tip pristupa	čitanje i pisanje	samo čitanje	čitanje i pisanje

Karakteristike odziva	brzo ažuriranje, promenljivo trajanje vremena odziva sistema	dugo vreme odziva sistema	kratko vreme odziva sistema
Nivo detaljnosti podataka	transakcioni podaci	delimično sumarni podaci	sumarni i agregacioni podaci
Struktura podataka	normalizovana (zapisi)	normalizovana ili denormalizovana	dimenziona i hijerarhijska
Količina podataka	gigabajti podataka	gigabajti/terabajti podataka	gigabajti podataka
Adaptivnost sistema	ograničena, uz značajnu upotrebu resursa	slaba	jednostavnost modifikacije
Brzina uvođenja sistema	mala (reda godine)	mala (reda godine)	velika (reda dana ili nedelje)

Osnovni elementi OLAP sistema su:

- baza podataka, koja služi kao osnova za analizu,
- OLAP server, za upravljanje i manipulaciju podacima,
- interfejs sistem, prema korisniku i prema drugim aplikacijama,
- alati za administriranje.

Pokušaj korišćenja OLAP pristupa nad bazama podataka koje su nastale na osnovu modela podataka projektovanog da podrži transakcioni nivo informacionih sistema i obezbedi zahtevani nivo integracije podataka ne može se izvesti dovoljno efikasno za praktičnu upotrebu, a takođe ugrožava nivo performansi transakcionog nivoa. Za korišćenje OLAP složene procedure potrebno je transakcione podatke prebaciti u posebnu bazu podataka.

OLAP pristup mora od hardvera da poseduje poseban računar, tzv. OLAP server, na koji se povezuju relacione BP, eksterni izvori podataka i ostali interni podaci, koji su podržani grafičkim interfejsima, radnim tabelama i ostalim PC alatima.

OLAP serveri koriste višedimenzione strukture za čuvanje podataka i veza između njih.

Višedimenzione strukture se najbolje vizuelizuju kao kocke podataka i kao kocke u kockama podataka. Svaka strana kocke se naziva dimenzijom. Kao što smo ranije rekli, dimenzija predstavlja kategoriju podataka, kao što su tip proizvoda, region, vreme itd. Svaka ćelija kocke sadrži agregirane podatke koji su u vezi sa dimenzijama. Naprimer, jedna ćelija može sadržati podatke o ukupnoj prodaji za dati proizvod i region u toku jednog meseca.

OLAP serveri podržavaju tipične analitičke operacije:

- konsolidacija – ovom operacijom se vrši agregacija podataka po zadatom kriterijumu,
- drill down/up – ove operacije omogućavaju prikazivanje više ili manje detalja podataka,
- isecanje (slice & dice) – ove operacije obezbeđuju prikazivanje podataka iz različitih perspektiva, pri čemu se isecanje najčešće vrši po vremenskoj dimenziji da bi se analizirali trendovi (naprimer, jedan isečak kocke može prikazivati sve podatke o prodaji za zadati tip proizvoda za sve regione, a drugi isečak može prikazivati sve podatke o prodaji po kanalima za svaki tip proizvoda).

Još jedna karakteristika OLAP servera jeste ta što oni smeštaju podatke u sabijenom, zgusnutom obliku. Ovo se postiže dinamičkom selekcijom tehnika za kompresiju podataka da bi se što bolje iskoristili prostori za čuvanje podataka. Retko popunjene matrice se čuvaju odvojeno od dosta popunjenih matrica. Na ovaj način OLAP serveri minimizuju zahteve za čuvanje podataka.

U sledećoj tabeli dat je pregled koristi koje se dobijaju uvođenjem OLAP servera u skladište podataka.

Tabela 4

Koristi za korisnike	**Koristi za sistem**
Izolacija korisnika od SQL jezika	Jednostavno upravljanje sistemom
Izolacija korisnika od relacionog modela	Automatsko održavanje
Povećanje performansi	Smanjenje učitavanja podataka
Povećane mogućnosti proračuna	Sistem ne mora više da generiše izveštaje za korisnike

Interfejs OLAP sistema treba da omogući korisniku komforan rad, samostalno izvođenje analitičkih operacija i dobijanje pregleda i poslovne grafike, bez znanja programiranja i strukture baze podataka.

OLAP alati veoma efikasno omogućuju prelaženje sa tabela na višedimenzione grafikone korišćenjem ekrana koji su dinamički promenljivi.

Ovako definisana OLAP ili hiper kocka sadrži desetine hiljada mogućih izveštaja koji se lako menjaju, brzo definišu i još brže izvršavaju.

OLAP je način obrade podataka koji karakterišu ad-hoc upiti, slabo struktuirani izveštaji i analiza koja obuhvata relativno mali broj transakcija, ali koja uključuje veliki broj tabela i zapisa u njima.

Zahtevi koje OLAP mora da ispuni su:

- mogućnost rada sa velikim skupom podataka i velikim brojem korisnika,
- kratko vreme odziva na upit,
- integrisani meta podaci koji povezuju OLAP server i relacionu bazu podataka,
- mogućnost rada sa podacima sa različitim nivoima detalja,
- sposobnost proračuna složenih matematičkih funkcija,
- podrška za šta-ako analizu, modelovanje i planiranje,
- jednostavnost uvođenja i održavanja sistema,
- zaštita podataka,
- mogućnost rada sa velikim brojema alata pomoću kojih će se pristupati podacima, vršiti analiza i prikazivati podaci.

Arhitekture OLAP sistema

Prenošenje samo izmenjenih podataka zahteva kompleksno programiranje i održavanje, dok kompletno zanavljanje podataka u skladištu postaje ozbiljan zadatak kada osnovna baza naraste preko, recimo, 5 gigabajta.

Neki od OLAP alata fizički prenose sve povezane transakcione podatke iz relacione baze podataka i iz drugih izvora u višedimenzionalnu bazu podataka, koristeći meta nivo, i pune skladište podataka preko noći (batch pristup), tj. izvode kompletno zanavljanje podataka u određenim vremenskim intervalima.

Drugi pristup, tzv. on-line, prenosi svaku pojedinačnu izmenu relacione u višedimenzionu bazu podataka, tj. izvodi se ažuriranje skladišta samo onim podacima koji su se izmenili između dva intervala ažuriranja.

Ovo su dva krajnja pristupa. Međutim, obično se koristi srednje rešenje, gde je batch osnovna metoda, dok se samo neke izmene na starim podacima prenose pojedinačno.

Oba gore definisana pristupa uslovila su pojavu i dve osnovne arhitekture, tzv. multidimenzioni OLAP (MOLAP) i relacioni OLAP (ROLAP). MOLAP je rešenje kada se koriste multidmenzione baze podataka, a ROLAP nastaje kao nadgradnja relacionih baza podataka.

MOLAP u osnovi podrazumevaju fizičko manipulisanje podacima, gde se fizička multidimenzionalnost postiže korišćenjem višestrukog indeksiranja ćelijskih struktura.

ROLAP definiše virtuelnu multidimenzionalnost, gde se koriste mehanizmi za logičku transformaciju radi proširenja performansi OLAP-a u RDBMS okruženju.

Kao što je gore rečeno, tendencija je u kombinovanju ova dva pristupa za izgradnju skladišta podataka.

Dakle, postoje sledeće arhitekture OLAP sistema:

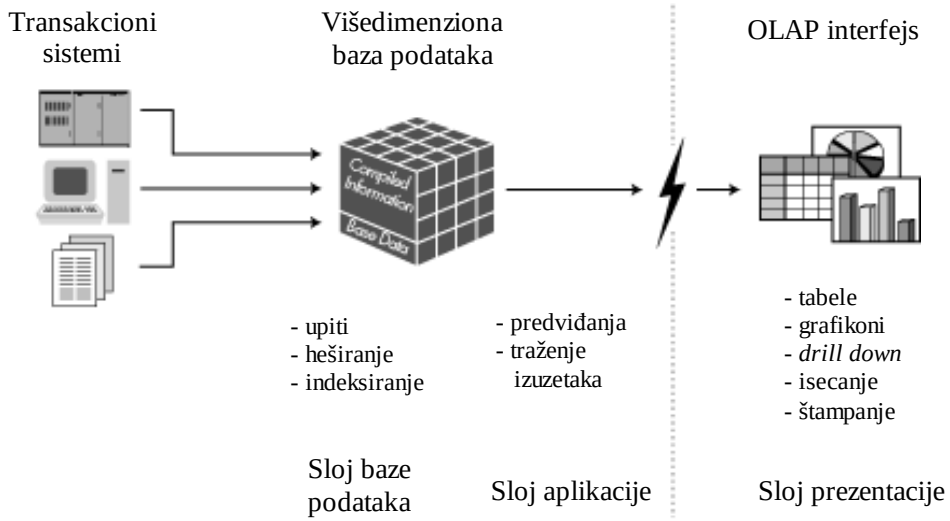
- višedimenzioni OLAP (MOLAP),
- relacioni OLAP (ROLAP),
- hibridni OLAP (HOLAP).

MOLAP i ROLAP se razlikuju po načinu fizičkog čuvanja podataka. Kod MOLAP sistema podaci se čuvaju u višedimenzionoj strukturi, a u slučaju ROLAP sistema podaci se čuvaju u relacionim bazama podataka.

Vi{edimenzioni OLAP (MOLAP)

MOLAP baze podataka imaju ograničenje fizičke veličine skupa podataka sa kojima mogu da barataju. Takođe, postoji i ograničenje na broj dimenzija koje još uvek obezbeđuju dobre performanse sistema. Da bi se vršila bilo kakva analiza, potrebno je prvo učitati podatke u višedimenzione strukture. Pri tome se vrše razni proračuni da bi se kreirale agregacije i popunili podaci, što vremenski može trajati relativno dugo. Po završenom procesu, korisnik može započeti analizu. Prednost MOLAP sistema je što obezbeđuju odlične performanse sistema kada se radi sa već sračunatim podacima (agregacijama). Nedostatak MOLAP sistema je teškoća dodavanja novih dimenzija. Prema tome, MOLAP sisteme je pogodno koristiti u slučajevima kada je moguće podeliti veliki skup podataka na više manjih skupova podataka.

Na sledećoj slici je prikazana arhitektura MOLAP sistema.



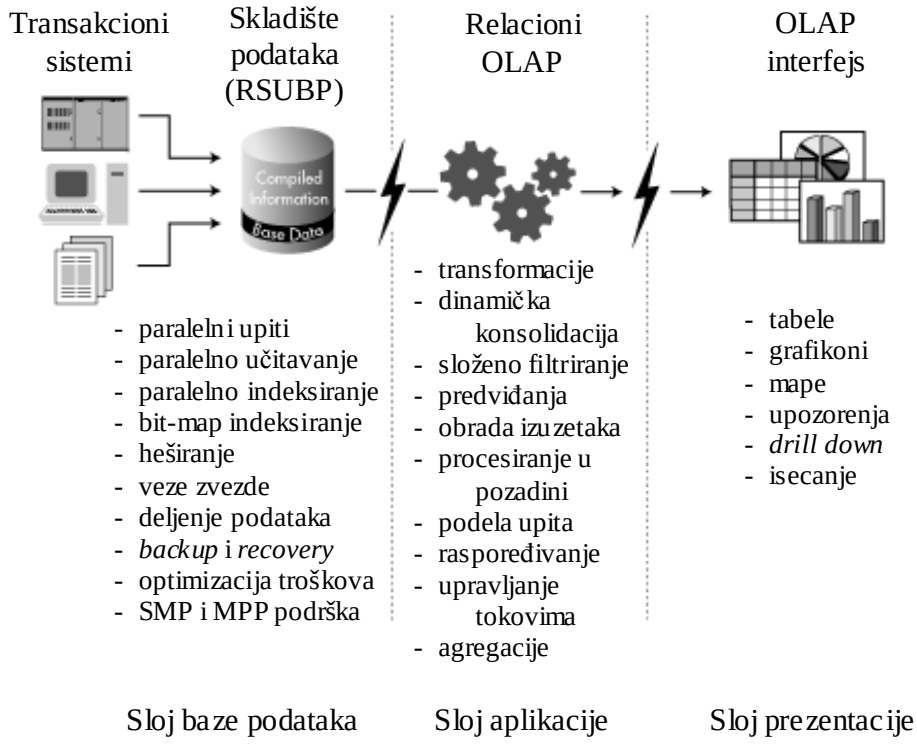
Slika 61. Arhitektura MOLAP sistema

Sa slike se vidi da se podaci iz različitih transakcionih sistema učitavaju u višedimenzionu bazu podataka pomoću batch rutina. Kada se završi sa učitavanjem podataka atomskog nivoa, prelazi se na kreiranje agregacija, nakon čega je baza podataka spremna za rad. Korisnici zadaju svoje zahteve za OLAP izveštajima putem interfejsa.

Relacioni OLAP (ROLAP)

ROLAP sistemi pristupaju podacima direktno iz skladišta podataka i rade sa relacionim bazama podataka. Ovi sistemi mogu da rade sa velikim skupovima podataka. Čim se odredi izvor podataka, korisnik može započeti analizu. S obzirom da se radi direktno nad bazom podataka, korisniku su uvek na raspolaganju tekući podaci. Takođe, kod ROLAP sistema ne postoje ograničenja po pitanju broja dimenzija koja postoje u slučaju MOLAP sistema.

Na sledećoj slici je prikazana arhitektura ROLAP sistema.



Slika 62. Arhitektura ROLAP sistema

Nakon što se definiše model podataka za skladište podataka, podaci iz transakcionih sistema se učitavaju u skladište podataka. Višedimenziona analiza, koju korisnik zahteva, dinamički se transformiše u niz SQL naredbi koje se dalje prenose na relacionu bazu podataka. ROLAP sistemi su optimizovani za pristupanje podacima, dok su MOLAP sistemi optimizovani za prikupljanje podataka. Prednost ROLAP sistema je što su sumarne tabele kreirane direktno u RSUBP-u, čime se obezbeđuje kratko vreme odziva sistema na upit, i što su tabele veoma čitljive.

Evo nekih karakteristika MOLAP i ROLAP sistema:

- višedimenziona analiza moguća je korišćenjem ROLAP i MOLAP sistema,
- za manje količine podataka ROLAP sistemi imaju skoro iste performanse kao i MOLAP sistemi,
- MOLAP sistemi nisu pogodni za rad sa velikim skupom podataka,
- MOLAP sistemi su manji od ROLAP sistema, te je potrebno manje U/I operacija pri pribavljanju podataka, što uslovljava da su MOLAP sistemi brži.

Postavlja se pitanje zašto onda uopšte i koristiti ROLAP sisteme. Odgovor na ovo pitanje je jednostavan. Za ROLAP sisteme već postoje razvijeni i provereni alati za čuvanje podataka. Takođe, u slučaju da se radi sa ROLAP sistemima, nema potrebe za premeštanjem podataka kao što je to slučaj sa MOLAP sistemima. ROLAP sistemi su otvoreni sistemi jer obezbeđuju direktan pristup podacima iz tabela te ne postoji potreba za dupliciranjem podataka kao što je to slučaj sa MOLAP sistemima.

Hibridni OLAP (HOLAP)

Možda najbolje rešenje predstavlja HOLAP, čiji alati mogu pristupati i relacionim i višedimenzionim bazama podataka. Cilj korišćenja HOLAP alata jeste da se iskoriste prednosti MOLAP alata (kratko vreme odziva sistema i analitičke mogućnosti) i ROLAP alata (dinamički pristup podacima). Pri tome se ne može reći da je HOLAP prost zbir MOLAP-a i ROLAP-a. To je zapravo ROLAP koji ima mogućnost izvršavanja vrlo složenih SQL naredbi. Dakle, cilj je bio da se zadrže sve prednosti ROLAP-a, ali da se pri tome dodaju i neke nove mogućnosti za rad sa višedimenzionim bazama podataka. Jedan od HOLAP alata je i Oracle Express.

Potrebe korisnika su:

- višedimenzioni pogled na podatke – ovu mogućnost poseduju i MOLAP i ROLAP alati,
- odlične performanse sistema – ovu mogućnost poseduju MOLAP alati,
- analitička fleksibilnost (za potrebe simulacija) – ovu mogućnost poseduju MOLAP alati,
- pristup podacima u realnom vremenu – ovu mogućnost poseduju ROLAP alati,
- veliki kapacitet podataka – ovu mogućnost poseduju ROLAP alati.

HOLAP alati moraju imati sledeće karakteristike:

- dimenzije se mogu dinamički ažurirati – ne samo da treba obezbediti brz pristup samim podacima, već se mora omogućiti i jednostavna izmena struktura podataka;
- mogućnost višedimenzionih pogleda zasnovanih na meta podacima relacionih sistema za upravljanje bazama podataka – HOLAP alati moraju koristiti meta podatke relacionih SUBP pri kreiranju višedimenzionog modela;
- brz pristup svim nivoima agregacionih podataka;
- jednostavno održavanje agregacija.

Danas postoje tri prilaza pri radu sa HOLAP alatima:

- Izbor između relacionih SUBP i višedimenzionih baza podataka. Sistem podržava obe strukture baza podataka, tako da se kreira MOLAP model koji se čuva u relacionom formatu. S obzirom da se ne može pristupati ovim modelima istovremeno, korisnik mora da izvrši izbor. Ako se koristi ovaj način rada, sve dimenzije se moraju unapred zadati u

višedimenzionom modelu, tj. ovaj način rada ne podržava dinamičku izmenu strukture modela.

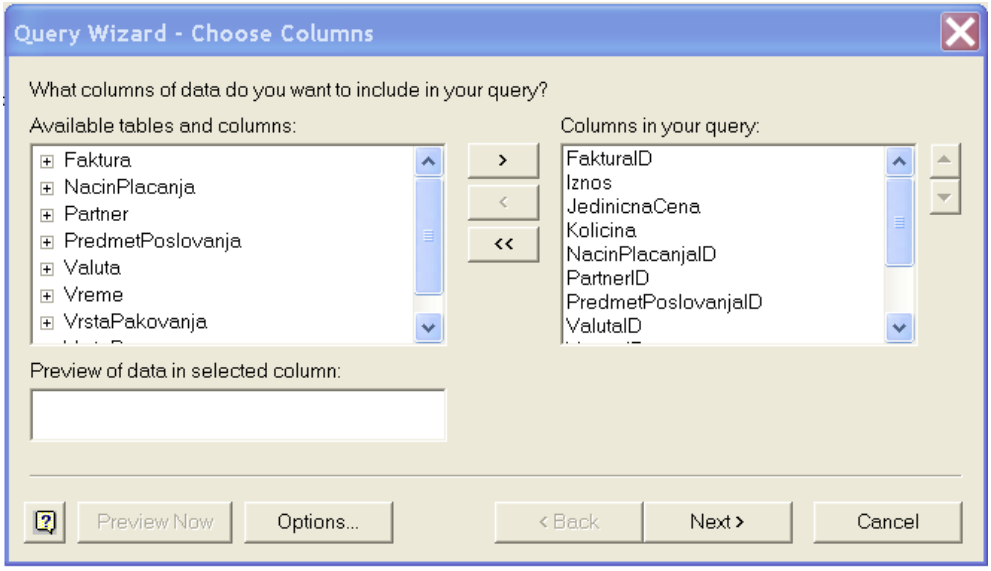
- Učitavanje rezultata relacionih upita u višedimenzionu bazu podataka. U ovom slučaju korisnik zadaje upit na osnovu kojeg se generišu SQL naredbe koje prikupljaju podatke iz relacionog SUBP-a i dostavljaju ih višedimenzionoj bazi podataka.
- Korišćenje višedimenzionih baza podataka za keširanje podataka i relacionih SUBP za dinamički pristup detaljnim podacima. Pri ovom načinu rada koriste se obe strukture baza podataka. Kreira se višedimenzioni model koji je statične strukture. Podaci se keširaju te kada korisnik zada upit, prvo se provere ti podaci. Ako se oni ne nalaze u kešu, generišu se SQL naredbe koje će prikupiti podatke iz relacionog SUBP-a u keš.

Kreiranje OLAP kocki pomoću Cube Wizard-a

Kreiranje OLAP kocki korišćenjem Cube Wizard-a integrisanog u Microsoft Query vrši se na osnovu prethodno definisanih upita nad skladištem podataka. Proces kreiranja kocke se odvija kroz sledeće korake:

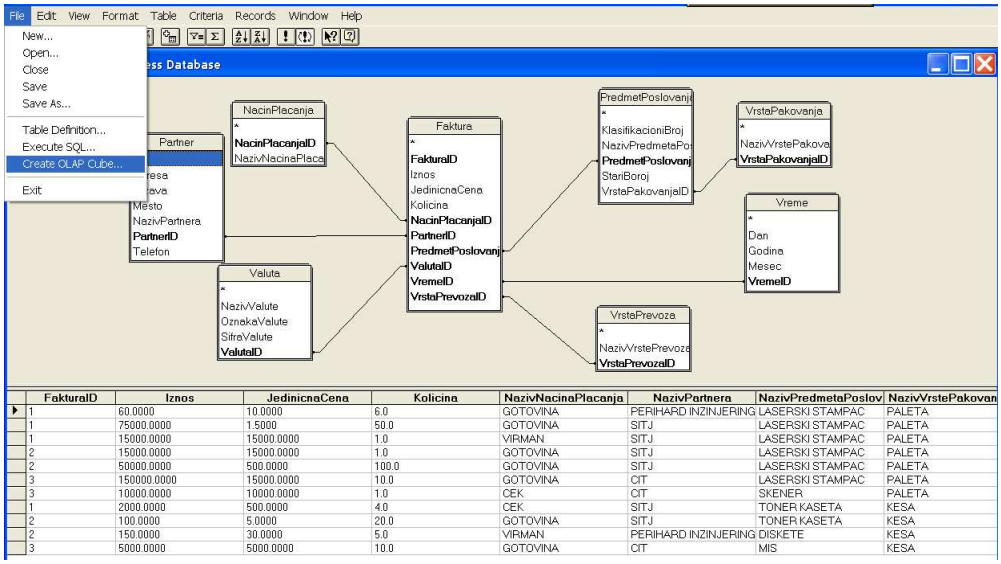
1. Po startovanju MS Query-ja izabere se tip baze podataka (u našem slučaju MS ACCESS) i pronađe odgovarajuća baza pdataka (u našem slučaju Faktura2000w.mdb).

Na osnovu prikazanih tabela biraju se kolone, kao što je pokazano na sledećoj slici.



Slika 63. MS Query izbor kolona

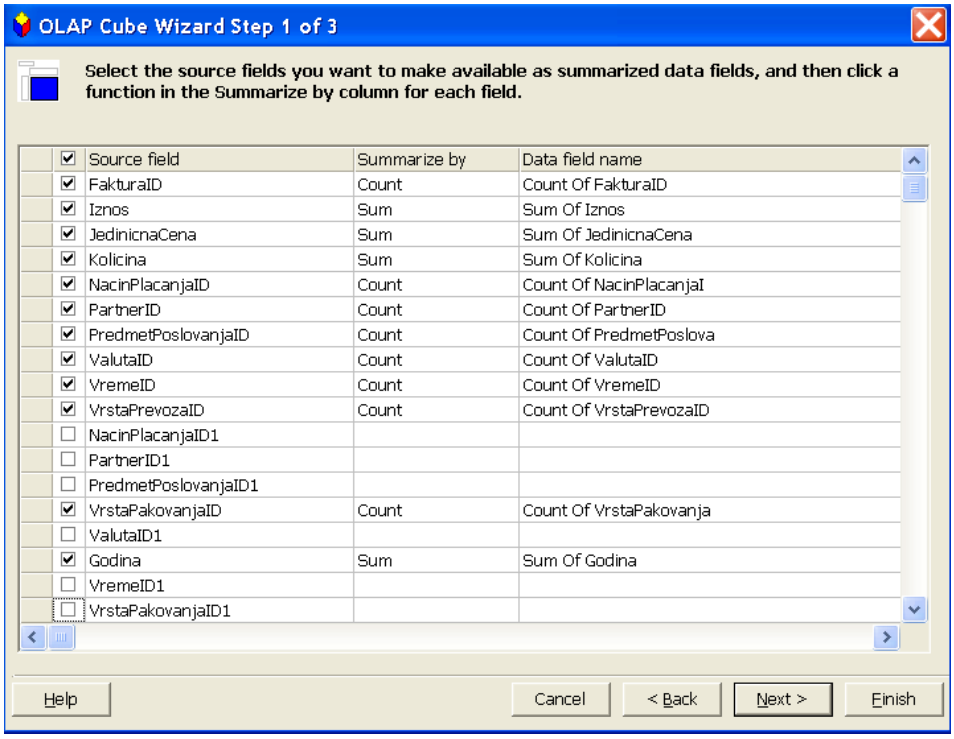
2. Pritiskom na tipku Next prikazuje se MS Query upit i bira opcija Create OLAP Cube...



FakturaID	Iznos	JedinicnaCena	Kolicina	NazivNacinaPlacanja	NazivPartnera	NazivPredmetaPoslov	NazivVrstePakovan
1	60.0000	10.0000	6.0	GOTOVINA	PERIHARD INZINJERING	LASERSKI STAMPAC	PALETA
1	75000.0000	1.5000	50.0	GOTOVINA	SITJ	LASERSKI STAMPAC	PALETA
1	15000.0000	15000.0000	1.0	VIRMAN	SITJ	LASERSKI STAMPAC	PALETA
2	15000.0000	15000.0000	1.0	GOTOVINA	SITJ	LASERSKI STAMPAC	PALETA
2	50000.0000	500.0000	100.0	GOTOVINA	SITJ	LASERSKI STAMPAC	PALETA
3	150000.0000	15000.0000	10.0	GOTOVINA	CIT	LASERSKI STAMPAC	PALETA
3	10000.0000	10000.0000	1.0	CEK	CIT	SKENER	PALETA
1	2000.0000	500.0000	4.0	CEK	SITJ	TONER KASETA	KESA
2	100.0000	5.0000	20.0	GOTOVINA	SITJ	TONER KASETA	KESA
2	150.0000	30.0000	5.0	VIRMAN	PERIHARD INZINJERING	DISKETE	KESA
3	5000.0000	5000.0000	10.0	GOTOVINA	CIT	MIS	KESA

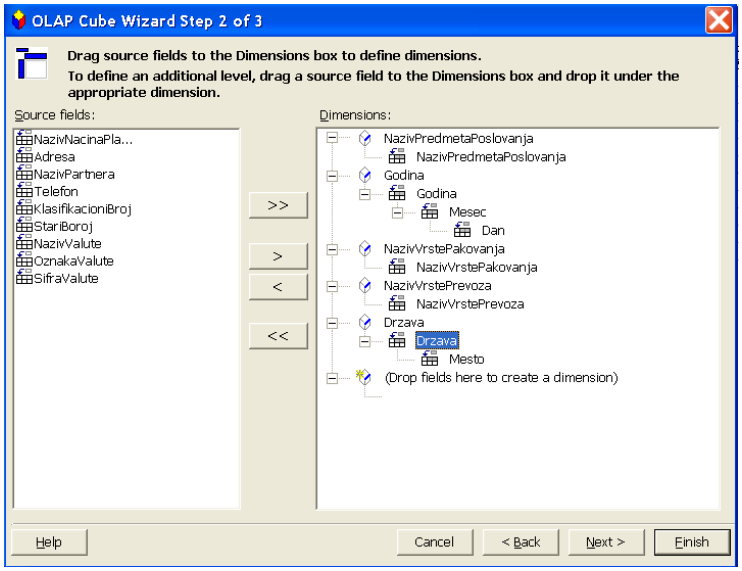
Slika 64. Grafički prikaz MS Query-ja

3. Ovo je prvi korak u formiranju OLAP Cube gde se određuju polja koja formiraju sumarne podatke i matematičke operacije koje će se izvršiti nad tim poljima. Polja koja se ne koriste za sumarne podatke predstavljaju kandidate za dimenzije kocke.



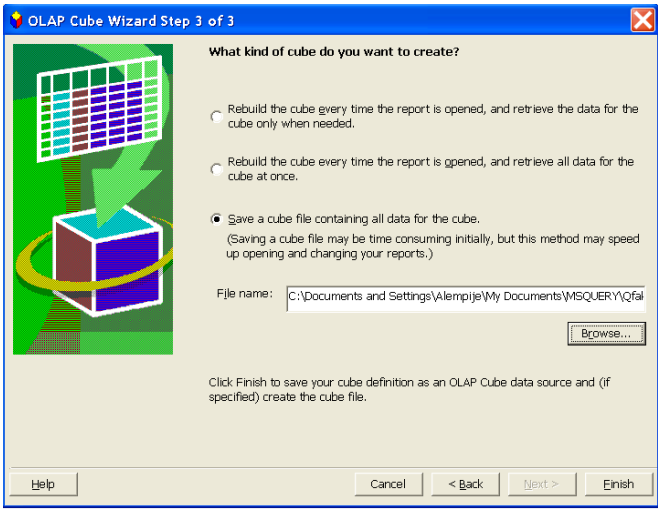
Slika 65. Izbor polja sumarnih podataka

4. Definisanje dimenzija OLAP kocke – izbor između preostalih polja (kandidata za dimenzije) onih koja će formirati dimenzije kocke. U ovom koraku se vrši i formiranje hijerarhije dimenzija. Hijerarhija dimenzija treba da omogući više nivoa detaljnosti, u zavisnosti od potreba korisnika u procesu analize podataka.



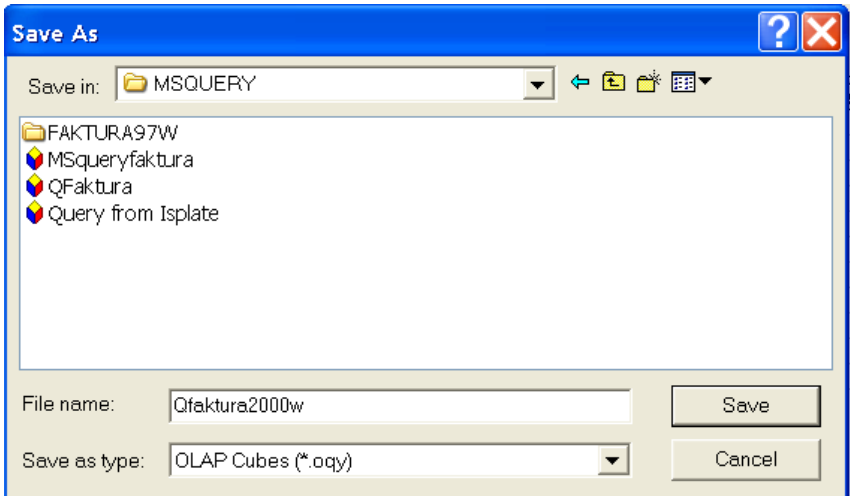
Slika 66. Formiranje dimenzija OLAP kocke

5. Snimanje fajla kocke na disk računara – u trećem koraku se bira mesto snimanja OLAP kocke, što se vidi na sledećoj slici.



Slika 67. Izbor mesta snimanja OLAP kocke

I, na kraju se izvrši snimanje fajla kocke na disk računara, kao što se vidi na sledećoj slici.

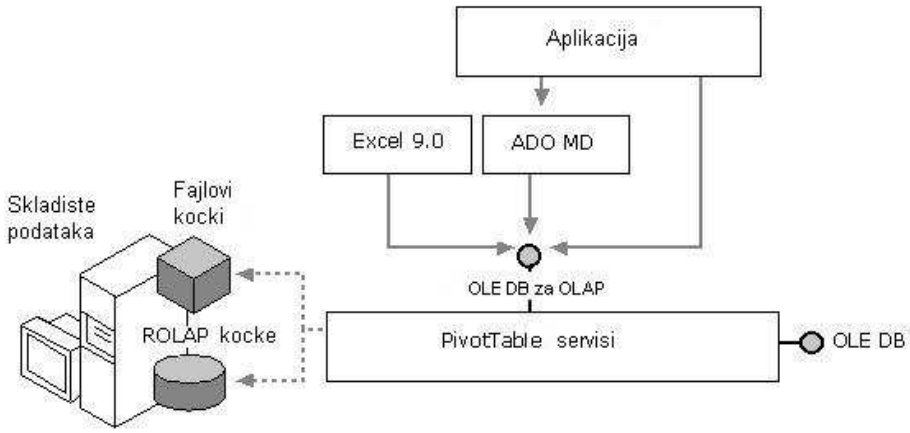


Slika 68. Snimanje fajla kocke na disk računara

Na osnovu ovako generisane OLAP kocke (fajl kocke) se, posredstvom Microsoft Excel-a, vrši kreiranje pivot tabele za pristup podacima koje ta kocka sadrži.

Analiza podataka korišćenjem Microsoft Excel dinamičke tabele

Analiza podataka organizovanih u OLAP kocke može da se vrši korišćenjem PivotTable (dinamička tabela sa objedinjenim podacima iz neke baze podataka) servisa koji omogućavaju pristup podacima u OLAP kockama. Na sledećoj slici su prikazana dva načina pristupa podacima u OLAP kockama, korišćenjem Microsoft Excel-a ili izradom posebne aplikacije, primenom takozvanih ADO mehanizama.

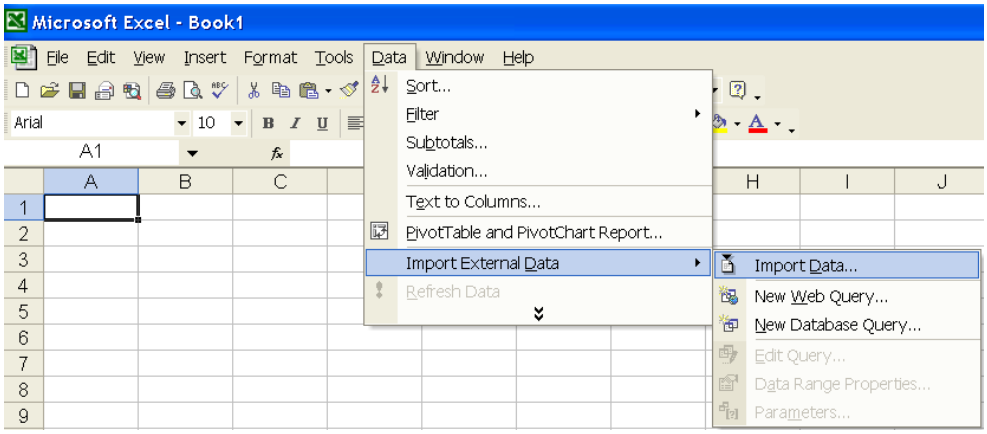


Slika 69. Pristup OLAP kockama

U našem slučaju, odlučeno je da se koristi Microsoft Excel jer je to alat čija je osnovna namena analiza podataka (Microsoft klasifikacija). Analiza podataka organizovanih u OLAP kocke u Excel-u se vrši izradom takozvanih pivot tabela. Microsoft Excel omogućava i vršenje analiza korišćenjem dodatnih alata, koji su njegov sastavni deo. Korisnik ima mogućnost da direktno iz Excel-a vrši štampanje izveštaja za određeni pogled na podatke (izabrani nivo detaljnosti i raspored dimenzija).

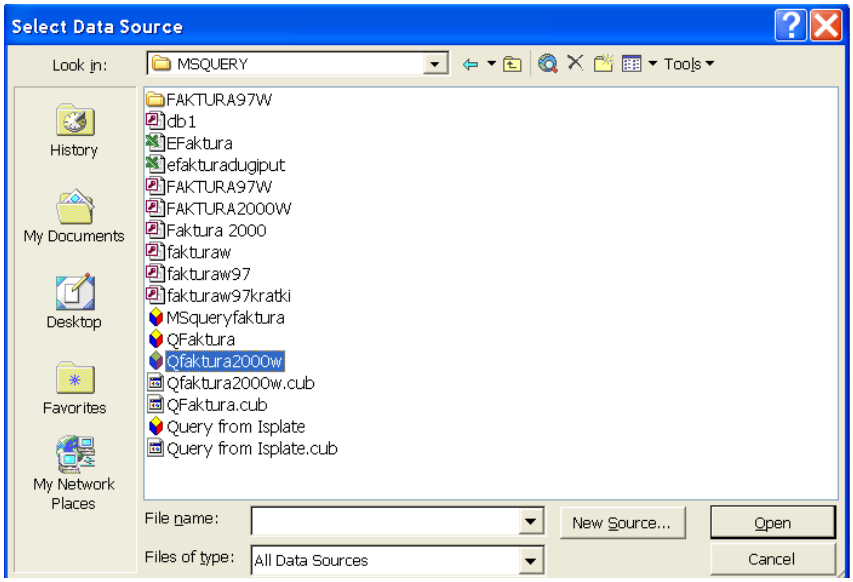
Pivot tabela predstavlja dinamičku tabelu sa objedinjenim podacima iz neke baze podataka. Ona služi za tabelarno prikazivanje više vrsta (dimenzija) podataka. U okviru nje se sumarni podaci mogu prikazivati na bilo kom nivou detaljnosti. Za potrebe izrade pivot tabela, u Microsoft Excel-u postoji čarobnjak (PivotTable Wizard). Sam postupak izrade se odvija u sledećim koracima:

1. Određivanje lokacije podataka – da bismo pristupili prethodno generisanim OLAP kockama, bira se opcija spoljni izvor podataka (eng. External data source).



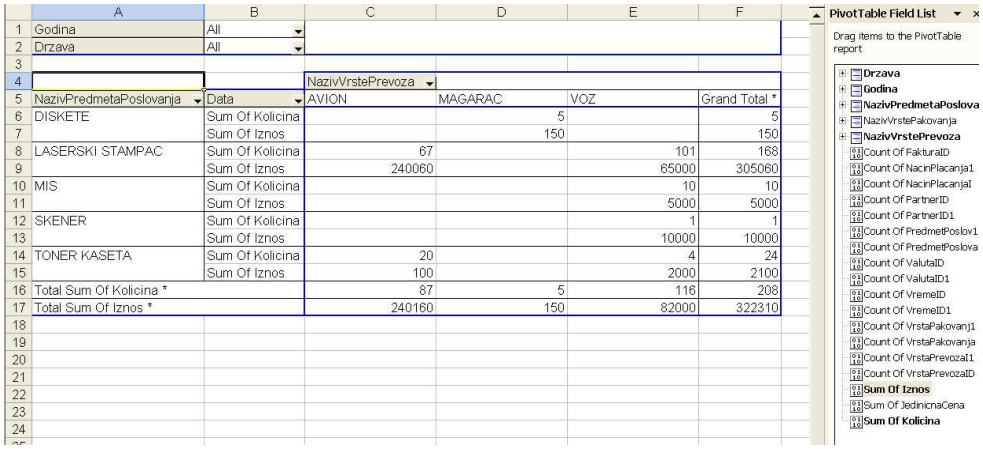
Slika 70. Određivanje lokacije podataka

2. Povezivanje sa spoljnim izvorom podataka – bira se opcija OLAP Cubes i okviru nje se bira kocka na osnovu koje formiramo pivot tabelu.



Slika 71. Izbor OLAP kocke

3. Formiranje pivot tabele – sa prozora PivotTable Field vrši se prevlačenje polja (dimenzija i sumarnih podataka) na odgovarajuća mesta u pivot tabeli (polja strane, reda i kolone). Formirana pivot tabela je prikazana na sledećoj slici.



	A	B	C	D	E	F
1	Godina	All				
2	Drzava	All				
3						
4			NazivVrstePrevoza			
5	NazivPredmetaPoslovanja	Data	AVION	MAGARAC	VOZ	Grand Total
6	DISKETE	Sum Of Kolicina		5		5
7		Sum Of Iznos		150		150
8	LASERSKI STAMPAC	Sum Of Kolicina	67		101	168
9		Sum Of Iznos	240060		65000	305060
10	MIS	Sum Of Kolicina			10	10
11		Sum Of Iznos			5000	5000
12	SKENER	Sum Of Kolicina			1	1
13		Sum Of Iznos			10000	10000
14	TONER KASETA	Sum Of Kolicina	20		4	24
15		Sum Of Iznos	100		2000	2100
16	Total Sum Of Kolicina *		87	5	116	208
17	Total Sum Of Iznos *		240160	150	82000	322310
18						
19						
20						
21						
22						
23						
24						

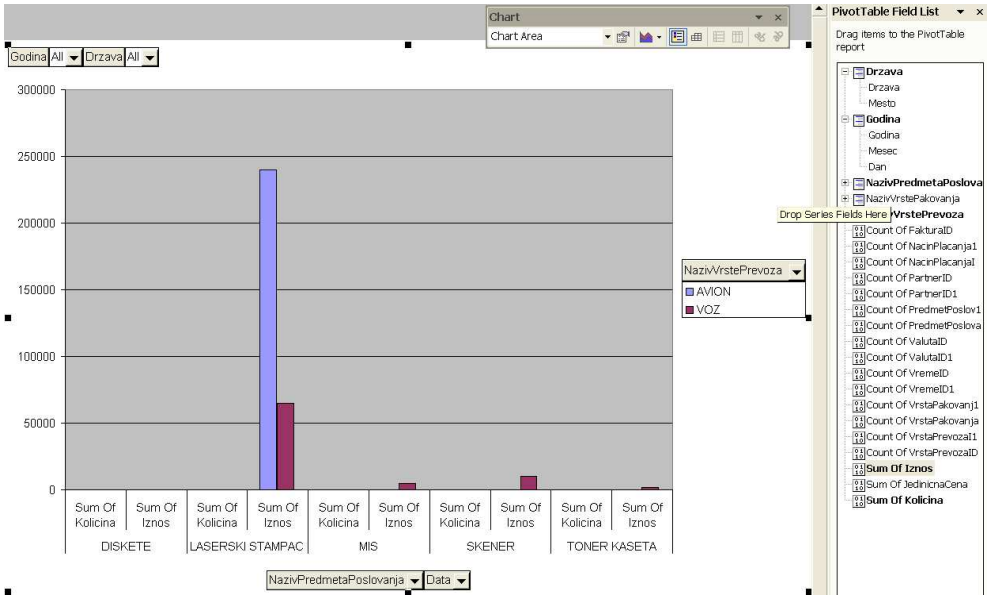
PivotTable Field List

Drag items to the PivotTable report

- Drzava
- Godina
- NazivPredmetaPoslova
- NazivVrstePakovanja
- NazivVrstePrevoza
- Count Of FakturaID
- Count Of NacinPlacanja1
- Count Of NacinPlacanjaI
- Count Of PartnerID
- Count Of PartnerID1
- Count Of PredmetPoslov1
- Count Of PredmetPoslova
- Count Of ValutaID
- Count Of ValutaID1
- Count Of VremeID
- Count Of VremeID1
- Count Of VrstaPakovanj1
- Count Of VrstaPakovanja
- Count Of VrstaPrevozaI1
- Count Of VrstaPrevozaID
- Sum Of Iznos
- Sum Of JedinicnaCena
- Sum Of Kolicina

Slika 72. Formiranje pivot tabele

Grafički prikaz pivot tabele prikazana je na sledećoj slici.



Slika 73. Grafički prikaz *Pivot* tabele

Data mining i otkrivanje znanja

Korisnici informacionih sistema s pravom zaključuju da su im uvođenjem automatizovanog informacionog sistema obećavali sve i svašta, a dobili su samo gomilu podataka. To je tačno, tim pre što se baze podataka uvećavaju zbog realnih potreba poslovnog procesa. Ono što se nameće jeste potreba za automatizacijom analize podataka. Upravo je to ono što omogućuje data mining.

Data mining je komponenta skladišta podataka. Data mining se zove i Knowledge Discovery in Databases (KDD). Data mining treba da uključi elemente baze znanja koji se koriste u ekspertnim sistemima i da analizira delove podataka da bi se identifikovala veza između naizgled "nepovezanih podataka".

Data mining je proces otkrivanja koji omogućuje korisnicima da shvate sisteme i veze između njihovih podataka. Data mining otkriva oblike i trendove u sadržaju ove informacije. Primer otkrivanja nekih podataka je i matični broj građana, gde su u strukturi broja smešteni podaci koji se mogu koristiti kao elementi za pretraživanje. Data mining mora da poseduje takva znanja da bez uplitanja korisnika nalazi elemente koji se mogu koristiti za grupisanje i identifikaciju oblika.

Data mining otkriva relacije našeg svakodnevnog komuniciranja sa podacima.

Drugi element su oblici, šabloni ili obrasci (patterns) koji nastaju na osnovu navika korisnika i koji se mogu aproksimovati na nova ponašanja. Data mining dozvoljava sagledavanje informacija na način koji ranije nije bio sagledavan.

Osnovna poruka data mininga jeste da je potrebno da iz ogromne količine operativnih podataka i veza koje se ne mogu odmah sagledati definišu odgovarajuće relacije, obrasci ponašanja, što u krajnjem slučaju treba da od podataka da potrebne informacije.

Sam podatak je sastavljen od serije karaktera koja sama po sebi ne znači ništa. Grupisani zajedno u obliku elemenata podataka, oni nešto znače. U sledećem koraku, elementi podataka podvrgnuti data mining analizi postaju veoma korisne informacije.

Dakle, data mining se može definisati kao proces podrške odlučivanju u kojem se traže šabloni infomacija u podacima. Osnovni cilj data mininga jeste otkrivanje skrivenih veza, predvidivih sekvenci i tačnih klasifikacija. Ovo pretraživanje može vršiti korisnik, naprimer izvođenjem upita (tada je to zaista teško), ili ga može vršiti neki "pametni" program koji automatski pretražuje bazu umesto korisnika i nalazi značajne šablone. Kada se nađe, informacija treba da se prezentuje na odgovarajući način, sa grafikonima, izveštajima itd.

Faze u izradi Data mininga

Proces izrade data mininga sastoji se od sledećih faza:

- selekcija – odabiraju se (segmentiraju) podaci po nekom kriterijumu (naprimer, svi ljudi koji poseduju automobile) da bi se odredili podskupovi podataka;
- preprocesiranje – ovo je faza "čišćenja" podataka, u kojoj se određene informacije odbacuju jer se smatraju nepotrebnim, a koje bi usporile obradu upita;
- transformacija – u ovoj fazi podaci se rekonfigurišu tako da postanu korisni i jednostavniji za pretraživanje;
- Data mining – u ovoj fazi se traže šabloni među podacima;
- interpretacija i razvoj – šabloni koji su otkriveni u prethodnim fazama interpretiraju se kao znanje koje se dalje može koristiti za procese podrške odlučivanju.

Proces data mininga uključuje još neke procese, kao što su induktivno učenje, statistika, mašinsko učenje i drugi. Induktivno učenje je proces kojim se analiziraju baze podataka da bi se našli šabloni. Slični objekti se grupišu u klase i pravila, čime se omogućava da se predvide klase do tada nepostojećih objekata. Mašinsko učenje je automatizacija procesa učenja. Mašinskim učenjem se prvo ispituju primeri i njihovi izlazi, a zatim se traži način da se oni reprodukuju i da se izvrši generalizacija za nove slučajeve. Proces data mininga prvo izvrši translaciju informacija iz baza podataka u oblike koji su pogodni za mašinsko učenje. Zatim se mašinskim učenjem na osnovu informacija određuje znanje koje se dalje upotrebljava za potrebe podrške odlučivanju.

Korisnici, aktivnosti i procesi data mininga

Potrebno je razlikovati korisnike, procese i aktivnosti. Aktivnost u sebe uključuje procese koji se mogu izvršiti. Prvo će biti reči o korisnicima. Postoje tri vrste korisnika:

- izvršioci,
- krajnji korisnici,
- analitičari.

Izvršiocima je potreban pogled "sa vrha". To su ljudi koji koriste računar mnogo manje od ostalih grupa. Oni su često nezadovoljni svojim informacionim sistemom za izvršioce (Executive Information System – EIS) i žele informacije, a ne sumarne podatke prikazane u vidu grafikona.

Krajnji korisnici znaju da koriste tabele, ali ne znaju program. To su, naprimer, prodavci, naučnici, inženjeri itd.

Analitičari znaju kako da interpretiraju podatke i vrše računanja, ali nisu programeri. To su

finansijski analitičari, statističari, konsultanti.

Sve ove vrste korisnika služe se trima vrstama data mining aktivnosti:

- epizodne,
- strategijske,
- kontinualne.

Korišćenjem epizodnih aktivnosti posmatraju se podaci iz jedne određene epizode (naprimer, određena marketinška kampanja) koji se mogu koristiti za predviđanje. Korisnik može da tumači ove podatke ili da ih koristi za predviđanje. Ove aktivnosti najčešće sprovode analitičari.

Korišćenjem strategijskih aktivnosti posmatra se veći skup međusobno povezanih podataka, s namerom da se dobije saznanje o globalnim merama ili vrednostima (naprimer, profit). Ove aktivnosti pokušavaju da daju odgovore na pitanja odakle nešto dolazi i slično.

Upotrebom kontinualnih aktivnosti pokušava se doći do saznanja kako se svet menja tokom nekog vremenskog perioda, kao i da se nađu faktori koji su uslovili to menjanje.

Postoje tri vrste procesa u data miningu:

- otkrivanje,
- modelovanje na osnovu predviđanja,
- sudska analiza.

Otkrivanje je proces kojim se posmatranjem baze podataka traže skriveni šabloni bez prethodnog saznanja kako bi ti šabloni trebalo da izgledaju. Drugim rečima, program sam traži šablone koji bi mogli biti od interesa, bez uplitanja korisnika (korisnik ne mora da misli o bitnim pitanjima). U velikim bazama podataka postoji veliki broj šablona pa korisnik ne može o svima da vodi računa. Glavno pitanje je bogatstvo šablona, kao i kvalitet podataka koji se dobijaju njihovom upotrebom. Određivanje "snage" i korisnosti je osnova tehnike otkrivanja.

Šabloni dobijeni modelovanjem na osnovu predviđanja koriste se za predviđanje budućnosti. Modelovanje na osnovu predviđanja omogućava da korisnik ostavi neka polja nepopunjena jer će sistem pokušati sam da pogodi vrednosti za ta polja na osnovu šablona dobijenih iz baze podataka. Ova tehnika koristi šablone koji su dobijeni tehnikom otkrivanja za pogađanje novih vrednosti koje treba upisati.

Sudska analiza je proces primene šablona radi dobijanja elemenata podataka koji su neuobičajeni, tj. koji su anomalije. Da bi se takvi elementi našli, prvo se zadaju tačni elementi, a zatim se u okviru baze podataka traže elementi koji odstupaju od zadatih.

Svaki od ovih procesa se može dalje klasifikovati. Naprimer, postoji nekoliko tipova otkrivanja šablona kao što su asocijacije, IF/THEN pravila itd.

Proces data mininga se sastoji od dve faze:

- izrada modela i
- predviđanje budućih rezultata.

Model je matematička formula koja objašnjava uticaj ulaza na izlaz. Iterativnom obradom podataka ova formula se može menjati i dovesti do tačnog oblika, tj. oblika koji u potpunosti odražava uticaj ulaza na izlaz. Kada se model kreira, može se iskoristiti za predviđanje budućih događaja. Ovaj model se može koristiti samostalno ili u sprezi sa tradicionalnim metodama analize, kao što su upiti nad skladištem podataka.

Prostori posmatranja

Pristup podacima i analiza podataka su različiti aspekti podrške odlučivanju i koriste se nad različitim računskim prostorima. Postoje četiri računska prostora koji karakterišu procese podrške odlučivanju:

- prostor podataka,
- prostor agregacija/OLAP,
- prostor uticaja,
- prostor varijacija.

Operacije pristupa podacima, kao što su upiti i izveštaji, koriste se nad računskim prostorom podataka, OLAP koristi višedimenzioni prostor agregacija, a data mining se koristi nad prostorom uticaja.

Upiti koji se postavljaju nad ovim prostorima potpuno su različiti. Na pitanje kao što je "Šta utiče na prodaju?" skoro je nemoguće odgovoriti direktno iz prostora podataka. Takođe, ovi prostori su često toliko veliki da se ne mogu unapred izračunati i sačuvati, kao što je to slučaj sa prostorom podataka (naprimer, nemoguće je unapred odrediti sve faktore uticaja unutar baze podataka).

Prostor podataka sadrži sve informacije iz ostalih računskih prostora, ali u manjem obliku. Ostali prostori sadrže manje infomacija od prostora podataka, ali su te informacije čitljivije i pristupačnije.

Interesantno je što su ova četiri računska prostora povezana sa četiri različita matematička koncepta. U prostoru podataka, relacije koriste skupove i članove. Struktura prostora agregacija je aritmetička. Prostor uticaja se odnosi na logiku. Struktura prostora varijacija koristi različite oblike proračuna. Često se spominje i peti prostor, koji se odnosi na geografske informacije. U ovom prostoru osnovu prestavljaju karte (mape).

OLAP i prostor agregacija

Za razliku od prostora podataka, u kome se čuvaju sami elementi podataka, u prostoru agregacija se čuvaju rezultati proračuna agregacija izvršenih nad podacima (naprimer, prosečan broj godina svake osobe u Beogradu, ukupna prodaja po gradovima za dati proizvod itd.). Sama ideja višedimenzione baze podataka veoma je jednostavna:

- Uzeti u obzir niz dimenzija, kao što su PROIZVOD, PRODAVNICA, GRAD. Ove dimenzije su najčešće u vezi sa nenumeričkim poljima u relacionim bazama podataka.
- Odrediti potreban broj mera, kao što su PRODAJA i POPUST. Mere su najčešće u vezi sa numeričkim poljima relacionih baza podataka.
- Naći agregacije nad merama uzimajući u obzir dimenzije (naprimer, prosečna mesečna prodaja u svim gradovima) i sačuvati ih za kasniju upotrebu.

Prema tome, prostor agregacija sadrži sve sumarne podatke izabranih proračuna koji su izvršeni nad prostorom podataka. Način na koji se ovi proračuni čuvaju može se posmatrati u vidu dimenzija i koordinata i time produbiti pojam višedimenzionalnosti. Pri kreiranju agregacija, često je potrebno da se u prostor podataka dodaju još neki podaci vezani za hijerarhije i periodična ponašanja (istoriju). Ovim se delimično izlazi iz okvira relacionih modela. Naprimer, relacioni modeli ignorišu prirodne hijerarhije tipa DRŽAVA, REGION, GRAD ili GODINA, POLUGODIŠTE, MESEC. Dodavanjem ovih informacija obezbeđuju su dodatne mogućnosti za korisnika.

Jedan od razloga zbog kojeg OLAP sistemi imaju kraće vreme odziva na upit od relacionih modela jeste u tome što se kod OLAP sistema agregacije jednom izvršavaju, a njihovi rezultati se čuvaju za kasniju upotrebu. Prema tome, OLAP sistemi samo pristupaju unapred izračunatim podacima smeštenim u prostor agregacija.

Data mining i prostor uticaja

Za razliku od prostora podataka i agregacija, priroda prostora uticaja je logička. Ovde se radi sa uticajima specifične grupe podataka na druge podatke. Ono što ovaj prostor čini interesantnim jeste to što su informacije koje se nalaze u njemu možda najkorisnije za samog korisnika, jer su one najčešće opšte te se mogu smatrati "znanjem".

Proces data mining se definiše kao proces podrške odlučivanju pomoću kojeg se nalaze šabloni informacija u podacima. Traženje ovih šablona može vršiti sam korisnik korišćenjem upita, što je težak i naporan posao, ili mu u tome može pomoći program koji automatski pretražuje bazu podataka i nalazi značajne šablone. Ovaj postupak se naziva otkrivanje (discovery).

Otkrivanje je proces u kome se traže skriveni šabloni podataka u bazi podataka bez unapred određene ideje ili hipoteze o tome kako oni izgledaju. Drugim rečima, program preuzima inicijativu u traženju interesantnih šablona, bez potrebe da korisnik mora da unapred misli o bitnim pitanjima. U velikim bazama podataka postoji puno šablona koje korisnik verovatno ne bi ni otkrio. Zato se ovaj računski prostor razlikuje od ostalih prostora. Izlaz procesa otkrivanja

često se može prikazati pravilima tipa ako-onda. Naprimer:

```
IF
        30 < Starost_Kupca < 42 AND
        Tip_Vozila = Kamion AND
        Broj_Dece < 2
THEN
 Popust = 5%
```

Proračunavanje izmena u prostoru varijacija

U prostoru varijacija se posmatraju izmene nastale u dimenzijama. Posmataju se ne samo izmene, već i stopa izmena.

Veza sa skladištem podataka

Postavlja se pitanje veze između skladišta podataka i data mininga. Kao i za skladište podataka, i za potrebe data mininga mora postojati jedinstven, odvojen, integrisan, konzistentan i "prečišćen" izvor podataka. Prema tome, data mining alati mogu u potpunosti da koriste skladište podataka jer ono zadovoljava te potrebe.

Cilj data mininga jeste da se nađu šabloni u podacima, ali je poželjno nalaziti samo one šablone koji su od značaja za dati posao. S obzirom da data mining alati pretražuju sve podatke, potrebno je odrediti skup podataka nad kojima će oni da rade, tj. treba zadati upit nad bazom podataka. Pošto je broj podataka najčešće veliki, vreme odziva takvog upita nad bazom podataka može biti veliko. Međutim, mehanizam procesiranja upita u skladištima podataka obezbeđuje da je vreme odziva na upit relativno kratko, te se za potrebe data mininga može koristiti. Postoji još jedan razlog zbog kojeg treba koristiti skladišta podataka za potrebe data mininga. Rezultati data mininga mogu biti korisni samo ako postoji način da se dalje istražuju otkriveni šabloni u podacima. Upravo skladište podataka daje mogućnost da se postavljaju nova pitanja samim izvorima podataka.

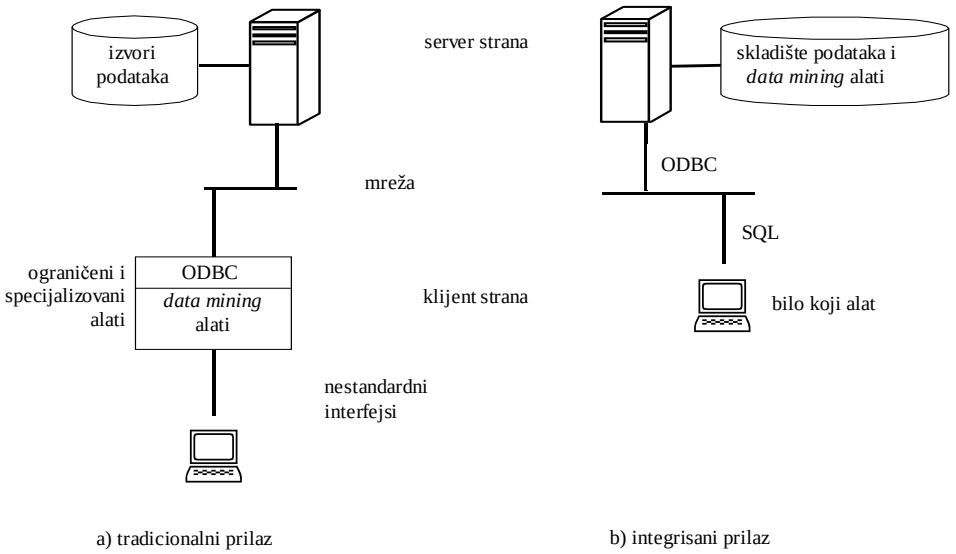
Danas se radi na integraciji data mining alata sa skladištem podataka. Postoji više razloga za ovu integraciju.

Prvo, kao što je već rečeno, data mining alati zahtevaju postojanje "prečišćenih" i integrisanih podataka. Tradicionalni data mining alati bi iz tih razloga prvo izvršili transfer podataka (možda i stotine gigabajta) putem mreže. Nakon završenog rada često se javlja potreba za novim podacima, što bi značilo da bi se ceo proces transfera morao ponoviti. Pri ovome se neprestano moralo voditi računa o zaštiti podataka i greškama pri prenosu.

Drugi razlog za integraciju data mining alata sa skladištem podataka jeste poboljšani korisnički

interfejs. Stariji data mining alati su zahtevali postojanje niza stručnjaka da bi se postigli zadovoljavajući rezultati. Danas, svaki poznavalac SQL jezika može koristiti mogućnosti data mininga.

Treći razlog za integraciju su performanse sistema i mogućnost proširivanja koje obezbeđuje skladište podataka, a koje su potrebne za data mining alate. Na sledećoj slici prikazan je tradicionalni i integrisani prilaz.



Slika 74. Prikaz tradicionalnog i integrisanog prikaza

Jedan od načina da se ostvari integracija jeste da se kreiraju modeli koji se u bazama podataka predstavljaju tabelama. Na ovaj način se ovim modelima može pristupati upotrebom SQL naredbi. Nakon kreiranja ovih tabela, u njih treba smestiti podatke koje će data mining alati da pretražuju. Obradom podataka, data mining alati će kreirati nove tabele u kojima će smeštati rezultate i koji se mogu pregledati kao i sve ostale tabele (korišćenjem SQL naredbi).

OLAP i data mining

OLAP i data mining su integralni delovi svakog procesa podrške odlučivanju. Ipak, sve do danas, većina OLAP sistema je težila samo da obezbedi pristup višedimenzionim podacima, dok su data mining sistemi analizirali uticaje podataka u okviru jedne dimenzije. Ovde će biti reči o tome da OLAP i data mining ne bi trebalo razmatrati kao odvojene procese već da ih treba u potpunosti spojiti.

Korišćenjem OLAP pristupa, korisnik postavlja pitanja tipa "Koje su prodaje po gradovima i po mesecima?" i sistem daje odgovor. Ipak, sam pristup nije dovoljan, s obzirom da se na taj način

dobija samo prikaz podacima koji se nalaze u skladištu podataka. Korisniku je potrebno da pretražuje podatke po više dimenzija ne bi li našao različite šablone podataka koji postoje u OLAP prostoru (naprimer, kako na profit utiču dimenzije GRAD i VREME). Prema tome, OLAP sistemi se moraju fokusirati ne samo na pristup podacima, već i na proces otkrivanja. Takođe, sistemi za podršku odlučivanju moraju uzeti u obzir i data mining. Ustvari, OLAP i data mining se moraju koristiti zajedno da bi se izbegli netačni rezultati jer, za razliku od transakcionih sistema, u kojima greške dizajniranja mogu da smanje performanse sistema i mogu se relativno brzo otkriti, greške koje se načine u sistemima za podršku odlučivanju se, najčešće, ne mogu uvideti za duži vremenski period.

U stvari, OLAP mining se koristi nad hibridnim prostorom formiranim od prostora podataka, agregacija i uticaja. OLAP mining pristupa prostoru podataka i prostoru agregacija putem SQL-a i OLAP-a.

Komponente OLAP data mininga su:

- relaciona baza podataka koja sadrži granularne podatke (ne mora biti skladište podataka),
- OLAP koji obezbeđuje brz pristup sumarnim podacima između više dimenzija,
- višedimenzioni proces otkrivanja koji će vršiti otkrivanje između dimenzija i spajati rezultate.

Obrazloženje za OLAP data mining

Zašto je uopšte potrebno da se obraća pažnja na hibridni prostor? Zašto ne napraviti veliku datoteku u kojoj će se čuvati svi podaci i po kojoj se pretražuje jer se, uostalom, svi podaci tu nalaze? Ili, zašto ne bismo samo pretaživali relacione baze podataka, bez OLAP alata koji samo komplikuju stvar?

Odgovor na ova i slična pitanja je jednostavan. Bez upotrebe OLAP data mininga, moguće je izostaviti ključne informacije ili se mogu dobiti netačni rezultati, kao što će biti pokazano.

Da bi se ovo najbolje shvatilo, najlakše je dati primer u kome će se prikazati kako se mogu doneti pogrešni zaključci ako korisnik nije pažljiv pri pretraživanju po više dimenzija. Takođe, prikazaće se kako obična datoteka nije pogodna za višedimenzioni data mining.

Razmotrimo podatke o prodaji date u tabeli 5.

Tabela 5

Proizvod	Boja proizvoda	Cena proizvoda	Prodavnica	Veličina prodavnice	Profit
Jakna	Plava	200	S1	1000	-200
Jakna	Plava	200	S2	5000	-100

Jakna	Plava	200	S3	9000	7000
Kapa	Zelena	70	S1	1000	300
Kapa	Zelena	70	S2	5000	-1000
Kapa	Zelena	70	S3	9000	-100
Rukavica	Zelena	50	S1	1000	2000
Rukavica	Plava	50	S2	5000	-300
Rukavica	Zelena	50	S3	9000	-200

Cilj procesa data mining je da se nađu šabloni uticaja. Naprimer, kako boja proizvoda ili veličina prodavnice utiču na profit. Da bismo malo uprostili stvari, posmatraćemo samo šablone koji nam govore kada su profiti pozitivni, a kada su negativni.

Ako korisnik nije pažljiv, ova tabela se može analizirati bilo kojom od metoda za detekciju uticaja, kao što su stabla odlučivanja, jednostavna pravila itd. Sve ove metode su dizajnirane tako da nalaze sličnosti. S obzirom da one mere sličnosti, one su nepogodne za traženje uticaja u prostoru agregacija. Bilo koja od ovih metoda će na osnovu podataka iz tabele 5 reći sledeće:

Pouzdanost = 75%

IF Boja Proizvoda = Plava

THEN Unosan = NE

Pouzdanost = 100%

IF Veličina Prodavnice > 5000 AND

Boja Proizvoda = Plava

THEN Unosan = DA

Ova na prvi pogled deluje dobro jer se iz ovoga može zaključiti da je jedino potrebno da se svi proizvodi plave boje prodaju u velikim prodavnicama. Ali, ovo nam ne govori o veličini profita. Pogledajmo sada tabelu 6, u kojoj se samo treća vrednost profita izmenila.

Tabela 6

Proizvod	Boja proizvoda	Cena proizvoda	Prodavnica	Veličina prodavnice	Profit
Jakna	Plava	200	S1	1000	-200

Jakna	Plava	200	S2	5000	-100
Jakna	Plava	200	S3	9000	100
Kapa	Zelena	70	S1	1000	300
Kapa	Zelena	70	S2	5000	-1000
Kapa	Zelena	70	S3	9000	-100
Rukavica	Zelena	50	S1	1000	2000
Rukavica	Plava	50	S2	5000	-300
Rukavica	Zelena	50	S3	9000	-200

Dva gornja pravila su takođe istinita i za tabelu 6. Drugim rečima, tabele 5 i 6, sa stanovišta sličnosti, daju iste šablone. Ali, ako se posmatra sa stanovišta agregacija, to nije tačno (rezultati agregacija su prikazani u tabelama 7 i 8).

Tabela 7			Tabela 8	
Boja Proizvoda	Profit		Boja Proizvoda	Profit
Plava	6400		Plava	-500
Zelena	1000		Zelena	1000

Prema tome, same sličnosti nisu dovoljne. Moraju se uzeti u obzir i agregacije. Ali, pre nego što zaključimo da su proizvodi plave boje odlični, razmotrimo tabele 9 i 10 koje su agregacije tabela 5 i 6.

Tabela 9 Tabela 10

Proizvod	Boja proizvoda	Profit		Proizvod	Boja proizvoda	Profit
Jakna	Plava	6700		Jakna	Plava	-200
Kapa	Zelena	-800		Kapa	Zelena	-800
Rukavica	Plava	-300		Rukavica	Plava	-300
Rukavica	Zelena	1800		Rukavica	Zelena	1800

U ovom slučaju, tabela 3A nam daje sledeće:

```
Pouzdanost = 50%
IF Boja Proizvoda = Plava
THEN Unosan = NE
```

Tabela 3B daje sledeći zaključak:

```
Pouzdanost = 100%
IF Boja Proizvoda = Plava
THEN Unosan = NE
```

Podsetimo se da su ova pravila, primenjena nad tabelama 5 i 6, bila ista, ali u OLAP prostoru ova sličnost u ponašanju počinje da nestaje.

Da li ovo uopšte ima smisla za samu tabelu 5? Ustvari, analiza može biti složenija nego što na prvi pogled deluje. Ako se pogledaju dve agregacije tabele 5 koje su date u tabelama 11 i 12, uočiće se novi podaci. Iz tabele 5 se vidi da je profit za jakne negativan (dva od tri puta se ustvari gubi novac), a ako se pogleda tabela 11, rad sa jaknama izgleda kao unosan posao. Na sličan način, iz tabele 5 se može videti da se u pet od šest slučajeva prodavnica čija je veličina veća od 1000 gubi novac, dok se to ne može zaključiti ako se pogleda tabela 12.

Tabela 11

Proizvod	Profit
Jakna	6700
Kapa	-800
Rukavica	1500

Tabela 12

Prodavnica	Veličina prodavnice	Profit
S1	1000	2100
S2	5000	-1400
S3	9000	6700

Prema tome, mnogi izrazi koji se generišu iz različitih pogleda ne moraju značiti ono što korisnik i očekuje jer se odnose na to da li je proizvod unosan, a ne na prave vrednosti. A takvi izrazi najčešće ne uključuju dimenzije, iako bi trebalo.

Takođe, ovaj primer pokazuje zašto i OLAP prilaz može biti pogrešan. Korisnik može samo da pogleda tabelu 7, da na osnovu nje zaključi da su svi proizvodi plave boje unosni i da na osnovu toga donese odluku da u svaku prodavnicu stavi samo te proizvode. Drugi OLAP korisnik može na osnovu tabele 12 da zaključi da su jakne unosan posao, te da ih zbog toga treba stavljati u svaku prodavnicu. Ustvari, plave jakne iz velikih prodavnica su proizvodi koji doprinose profitu, ali OLAP korisnici će to teško moći da uvide. Većina OLAP korisnika i nema dovoljno vremena

na raspolaganju da analizira sve relevantne scenarije, te zbog toga mogu dobijati pogrešne razultate.

Pravo rešenje ovog problema jeste korišćenje OLAP data mining sistema, koji pretražuju podatke po više dimenzija i koji su "svesni" šablona koji postoje između ovih dimenzija, spajaju ih i rade zajedno sa korisnikom. Takvi alati mogu dati informacije kao što su:

- "Proizvodi plave boje su u proseku unosni, ali većina profita dolazi od jakni koje se prodaju u većim prodavnicama."
- "Proizvodi zelene boje su takođe unosni, ali ako se prodaju u malim prodavnicma."

Ovo je jedino moguće ako se kombinuju proračunavanja sličnosti sa agregacijama. Ne postoji drugi način da se izbegne konfuzija. Drugim rečima, čim se prihvati činjenica da su podaci ionako višedimenzioni, onda nema smisla da se podaci proučavaju po jednoj od dimenzija.

Algebra višedimenzionih objekata

Osnovni koncept je da se sa objektima manipuliše u višedimenzionom prostoru ili u prostoru uticaja. Prikazivanje ovih elemenata kao objekata olakšava rad sa dimenzijama i atributima.

Osnovna jedinica višedimenzionog prostora je kocka, kao što je osnovna jedinica u relacionom svetu tabela. Kocke imaju atribute, kao što relacije imaju kolone. Osnovni koncepti su:

- klase ili osnovne dimenzije,
- atributi,
- objekti ili primerci (instance),
- hijerarhije,
- vrednosti atributa,
- mere.

Osnovna dimenzija je klasa, kao što su PROIZVOD, PRODAVNICA, KUPAC itd. Svaka klasa/dimenzija može imati niz atributa (naprimer, dimenzija KUPAC može imati atribute STAROST, PRIHODI itd.; dimenzija PRODAVNICA može imati atribute VELIČINA, LANAC itd.; dimenzija PROIZVOD može imati atribute CENA, BOJA, KATEGORIJA itd.).

Svaka dimenzija ima objekte ili elemente kao primerke (naprimer, dimenzija KUPAC može imati objekte "Petar Petrović" i "Marko Marković"; dimenzija PROIZVOD može imati objekte kao što su "Kape" i "Jakne"). Svaki atribut objekta ima vrednost (naprimer, "Petar Petrović" ima 26 godina, cena "Kape" je 75).

Hijerarhijske osobine se mogu prikazati u okviru dimenzija i atributa. Naprimer, klasa TIP može biti roditeljska klasa klasi PROIZVOD. Tada vrednost atributa klase TIP nasleđuje i klasa

PROIZVOD, kao što je to slučaj u objektnom svetu. Ipak, mora se naglasiti da se sva proračunavanja mogu vršiti nad tipovima proizvoda kao da su oni sami klase za sebe. Na sličan način, prodavnice se grupišu po zonama, gradovi po državama, države po regionima itd. Koncept GRAD, DRŽAVA, REGION naziva se hijerarhija.

Mere su proračuni izvršeni nad prostorom kojeg formira dimenzija (naprimer, PRODAJA, PROFIT, MARGINE itd.). One su najčešće numerička izračunavanja, dok ostali atributi mogu biti numerički, ali i nenumerički.

Svaka kocka predstavlja agregaciju izračunavanja nad skupom dimenzija (naprimer, suma svih prodaja za sve proizvode u svim prodavnicama, suma svih prodaja za sve proizvode u jednoj prodavnici itd.). Tačka kocke se naziva koordinata ili primerak (naprimer, prodaja za jakne u prodavnici S1). Vrednosti svih mera se izračunavaju na osnovu kocke (naprimer, "prodavnice po državama" daju niz država sa vrednostima za prodaju).

Algebra za kocku obezbeđuje operacije koje se mogu primeniti nad njom, isto kao što relaciona algebra obezbeđuje operacije za rad sa tabelama. Može se izvršiti projekcija kocke. Projekcije vrše agregacije nad dimenzijom (naprimer, ako postoji "prodaja po državama i po mesecima" i ako se izvrši projekcija po mesecima, dobiće se samo prodaja po državama). Selekcija je krajnje jednostavna. Naprimer, ako postoji "prodaja po državama i po mesecima", selekcija se može izvršiti sa "WHERE DRŽAVA = Jugoslavija". Ono što je više interesantno jeste da su potrebne i operacije za rad sa unutrašnjošću kocke (naprimer, može se raditi sa "prodajom po državama i po mesecima" i sa "troškovima po državama i po mesecima" da bi se dobio "profit po državama i po mesecima").

Put od kocke do domena uticaja

Domen uticaja je logički ekvivalent kocke u višedimenzionom prostoru. Dok nam kocka daje podatke o agregacijama, domen uticaja nam govori o implikacijama. Kao i kocka, i domen uticaja ima atribute i vrednosti, ali domen uticaja daje faktore pouzdanosti, a ne samo sume brojeva kao što je prodaja (naprimer, koja je pouzdanost da boja proizvoda utiče na profit). Pouzdanost se najčešće izražava u procentima od 0 do 100%. Domen uticaja je, u stvari, funkcija preslikavanja intervala parova u kocki i mere u linearnu meru pouzdanosti.

Većina algoritama za stabla odlučivanja, neuralne mreže itd. ne snalaze se sa višedimenzionalnošću zbog sledećih razloga:

- koreni tih algoritama leže u teoriji verovatnoće i statistike, te oni proračunavaju mogućnosti, a ne agregacije,
- rade sa jednom tabelom u datom trenutku, umesto sa svim dimenzijama,
- brojne vrednosti moraju da podele na specifične intervale te time promašuju šablone između više dimenzija.

Prema tome, rad sa domenom uticaja zahteva novu teoriju, nove strukture podataka i algoritme.

Danas, većina korisnika vidi OLAP odvojen od relacionog prostora. Mora se uočiti da je domen uticaja različit od OLAP-a, kao što je OLAP različit od relacionog prostora. U tabeli 13 prikazano je kako svaki računski prostor zahteva različit prilaz.

Tabela 13

Teorija/Metoda	Osnovna jedinica	Struktura podataka	
Relacioni	relaciona algebra	relacija	treći normalni oblik
OLAP	OLAP algebra	kocka	šeme zvezde
Data mining	algebra uticaja	domen uticaja	rotacione šeme

Čim se prihvati činjenica da je domen uticaja različit od kocke, trebalo bi da postane jasno da same šeme zvezda ne mogu da posluže za data mining jer one ne mogu da reprezentuju domen uticaja. One su namenjene za rad sa agregacijama, a ne za analizu uticaja. One odlično rade sa OLAP alatima, ali ne i sa OLAP data miningom. Šeme zvezda se moraju proširiti da bi se podržao rad sa domenom uticaja.

Prvo treba uočiti da je veličina domena uticaja najčešće veća od same kocke nad kojom je domen i nastao. A sami prostori smešteni u kocki mogu biti veliki. Proračun uticaja se mora vršiti nad šemama koje mogu da smeste više agregacija nego što to mogu šeme zvezda (bilo bi previše skupo da se proračun stalno ponovo vrši). Za vreme analize uticaja, potrebno je da se fokusira na neku tabelu dimenzije (naprimer, PROIZVOD ili PRODAVNICA) u okviru zvezde, a onda su nam za svaku od dimenzija potrebni još neki podaci koji se nalaze u zvezdi. Može se zamisliti da se za vreme analize fokus premešta u smeru kazaljke na satu sa prodavca na prodavnicu, a onda na proizvod itd. U svakom trenutku tog premeštanja, potrebno je da se izvrši obogaćivanje fokusirane dimenzione tabele dodatnim odabranim podacima ili agregacijama. S obzirom da se čini da taj fokus rotira unutar zvezde, uveden je termin rotacione šeme da bi se naznačile ovakve strukture.

Tabela rotacione šeme sadrži pet osnovnih delova:

- fokusirana dimenzija (naprimer, KUPAC),
- mera koja je proračunata za fokus (naprimer, PROFIT),
- interni atributi (naprimer, STAROST KUPCA),
- spoljni atributi (naprimer, omiljena boja proizvoda za datog kupca),
- nefokusirane mere (naprimer, prosečan broj proizvoda koji kupac uzima).

Za vreme procesa OLAP otkrivanja, fokus neprestano rotira između različitih dimenzionih tabela, dok se prelazi hibridni prostor da bi se izvršila analiza uticaja.

Sada se postavlja pitanje da li rotacione šeme treba koristiti u skladištima podataka. Odgovor je: ne. Njih treba koristiti samo u sistemima tipa data mining. One su pogodne za prostor uticaja, ali ne i za prostor podataka.

Otkrivanje obrazaca u korišćenju Weba

Tehnike za otkrivanje obrazaca u Web miningu potrebne su jer se korisnici suočavaju sa problemima vezanim za pronalaženje odgovarajućih informacija (mala preciznost i spor odziv), nemogućnost kreiranja novog znanja na osnovu raspoloživih informacija na Webu, različita reagovanja na sadržaje i prezentacije podataka na Webu i otkrivanje potreba kupaca ili pojedinačnih korisnika.

Web mining otkriva značajne poslovne trendove i veze pomoću integracije Internet i intranet poslovnih informacija i drugih poslovnih podataka.

Web mining tehnike mogu biti korišćene u rešavanju problema prevelike količine informacija, bilo direktno, bilo indirektno.

Značaj efikasnog poslovanja na Webu

Internet je najznačajnija tehnologija koju je svet ikada video, i predstavlja ogroman potencijal za revoluciju poslovanja i marketing tehnika. Prvi korak koji treba da se preduzme jeste da bolje razumete ko posećuje Vaš sajt i kako ga koristi. Ova informacija omogućava donošenje boljih poslovnih odluka, usmeravanjem marketinga prema najboljim potrošačima. WM istraživanja su konvergirajuća oblast koja se sastoji od nekoliko istraživačkih celina, kao što su baze podataka, otkrivanje informacija (Information Retrieval, IR), ekstrakcija informacija (Information Extraction, IE), mašinsko učenje (Machine Learning, ML) i procesiranje prirodnog jezika (NLP). Ali, pošto je WM ogromna, interdisciplinarna i veoma dinamična oblast istraživanja, nesumnjivo postoji i mnogo propusta u ovoj oblasti.

Na Web sajtovima se stvaraju ogromne količine informacija, od kojih su mnoge veoma korisne. Međutim, stvarna moć i značaj ovih informacija dolazi do izražaja tek kada se one povežu sa drugim bazama podataka, kao što su automatizovani sistemi o kupcima, sistemi izveštavanja i istraživački sistemi. Ključni problem je u povezivanju podataka, pošto obično postoji veliki broj različitih sistema, posebno različitih baza podataka. Upravo u ovoj oblasti Web mining tehnike mogu dati veliki doprinos. Web mining sistem može integrisati sve raspoložive izvore podataka, bez obzira na oblik baza podataka u kojima se nalaze. Pošto se informacije menjaju svakoga minuta, neke od povezanih baza podataka mogu biti veličine stotinu gigabajta, što za Web mining sistem ne predstavlja teškoću, jer se unutar njega podaci prvo prečiste i filtriraju, a zatim i

koriste za neophodne analize. Integracijom različitih izvora podataka o kupcima i poslovanju, Web mining pomaže u davanju odgovora na kritična pitanja za poslovanje.

Šta je Web mining?

Web mining predstavlja korišćenje data mining (DM) tehnika za automatsko otkrivanje i ekstrakciju korisnih informacija iz Web dokumenata.

WM izvodi sledeće podzadatke:

- pronalaženje resursa: ovaj zadatak obuhvata nalaženje zahtevanih Web dokumenata;
- selekcija informacija i pretprocesiranje: obuhvata automatsku selekciju i pretprocesiranje specifičnih informacija iz zahtevanih Web dokumenata;
- generalizacija: obuhvata automatsko otkrivanje zajedničkih obrazaca na pojedinačnim Web sajtovima;
- analize: obuhvata validaciju i/ili interpretaciju otkrivenih obrazaca.

Pod pronalaženjem resursa se podrazumeva proces otkrivanja podataka, bilo online ili offline, iz tekstualnih izvora raspoloživih na Webu, kao što su: elektronska pisma, elektronski telegrami, razni tekstualni sadržaji HTML dokumenata. Ovde treba uključiti i tekstualne izvore kojima se, generalno, ne može pristupiti na WWW-u, ali su pristupačni kao online tekstovi napravljeni jedino za istraživačke svrhe, tekstualne baze podataka itd.

Selekcija informacija i pretprocesiranje predstavljaju obiman korak bilo kog procesa transformacije izvornih odataka otkrivenih u IR procesu (procesu otkrivanja informacija). Ove transformacije mogu biti bilo koji vid pretprocesiranja koji je ranije pominjan, ili ciljno procesiranje radi dobijanja željenih oblika podataka.

Za generalizaciju se obično koriste tehnike mašinskog učenja, ili DM tehnike. Treba napomenuti da ljudi igraju veoma važnu ulogu u procesu otkrivanja znanja ili informacija na Webu, pošto je Web jedan vrlo interaktivan medijum. Ovo je veoma važno za validaciju i/ili interpretaciju. Zbog toga je interaktivno, upitno-orijentisano otkrivanje znanja na Webu mnogo važnije od automatizovanog otkrivanja znanja na osnovu podataka.

Web mining obuhvata celokupan proces otkrivanja potencijalno korisnih i prethodno nepoznatih informacija, ili znanja, iz podataka na Webu.

Web mining kategorije

Web mining se može definisati kao otkrivanje i analiza korisnih informacija na WWW-u .Web mining se deli na tri istraživačke oblasti, u zavisnosti od toga koji deo Web-a ispituju, a to su: otkrivanje sadržaja na Web-u (Web Content Mining), otkrivanje strukture veza na Web-u (Web Structure Mining) i otkrivanje obrazaca u korišćenju Web-a (Web Usage Mining).

Otkrivanje sadržaja na Web-u (Web Content Mining) predstavlja otkrivanje korisnih informacija iz Web sadržaja, podataka i dokumenata. Sadržaj Web-a se sastoji od nekoliko vrsta podataka, kao što su tekstualni podaci, slike, audio i video zapisi i metapodaci, poznatiji kao hiperlinkovi. Poslednja istraživanja u otkrivanju višestrukih vrsta podataka se nazivaju multimedijalni DM, tako da možemo smatrati multimedijalni DM kao jedan deo sadržaja Web Mininga . Ipak, najveći deo podataka na Web-u čine nestrukturirani tekstualni podaci. Istraživanja u oblasti primene DM tehnika na nestrukturirane tekstove se nazivaju otkrivanje znanja u tekstovima (Knowledge Discovery in Texts, KDT), ili tekstualni Data Mining, ili Tekst Mining. Stoga, možemo smatrati Tekst Mining delom sadržaja Web Mining-a. Istraživanja urađena u oblasti Web Content Mining-a mogu se podeliti u dva različita pristupa: pristup zasnovan na agentima i pristup zasnovan na bazama podataka. Cilj Web Content Mining-a u pristupu zasnovanom na agentima je da pomaže korisnicima u pronalaženju relevantnih informacija, ili u njihovom filtriranju, na osnovu odgovarajućih profila korisnika, dok je cilj Web Content Mining-a u pristupu zasnovanom na bazama podataka da pokuša da formira model podataka na Web-u i da ga integriše sa mnogo sofisticiranijim upitima nego što su ključne reči, na osnovu kojih pretraživači vrše pretraživanja.

Otkrivanje strukture veza na Web-u (Web Structure Mining) nastoji da otkrije fundamentalni model strukture linkova na Web-u . Model mora biti zasnovan na topologiji hiperlinkova sa opisom linkova, ili bez opisa. Ovaj model može biti korišćen za kategorizaciju Web strana, a koristan je i za generalizaciju informacija, kao što su sličnosti i veze između različitih Web sajtova. Web Structure Mining može biti korišćen i za otkrivanje autorizovanih sajtova.

Otkrivanje obrazaca u korišćenju Web-a (Web Usage Mining) pokušava da da smisao podacima generisanim u Web korisničkim sesijama, ili podacima o ponašanju korisnika . Dok Web Content Mining i Web Structure Mining koriste realne i primarne podatke na Web-u, Web Usage Mining otkriva sekundarne podatke izvedene iz interakcija korisnika u toku njihovog rada na Web-u. Ova oblast koristi podatke iz pristupnih logova Web servera, logova proksi servera, logova pretraživača, korisničkih profila, registracionih podataka, korisničkih sesija ili transakcija, korisničkih upita, pokretanja ili pritiskanja tastera miša, i bilo koje druge podatke koji nastaju kao rezultat interakcije korisnika. Međutim, treba naglasiti da između navedenih kategorija ne postoji granica koja ih jasno razdvaja. Njihovo razdvajanje je moguće sa tačke gledišta opsega (scope) koji je obuhvaćen mnogim radovima urađenim u ovim oblastima. Lokalni opseg obuhvata individualne Web sajtove, dok se globalni opseg rasprostire na čitavom Web-u. Opseg Web Structure Mining-a i Web Content Mining-a u pristupu zasnovanom na agentima je globalan, dok je opseg Web Usage Mining-a i Web Content Mining-a u pristupu zasnovanom na bazama podataka lokalan.

Otkrivanje sadržaja na Web-u vezano je za pristup zasnovan na agentima i pristup zasnovan na bazama podataka.

Pristup zasnovan na agentima definisan je sa tri kategorije: inteligentni agenti za pretraživanje, filtriranje/kategorizacija informacija i lični web agenti.

Inteligentni agenti za pretraživanje koji korišćenjem domenskih karakteristika i profila korisnika

organizuju i interpretiraju otkrivene informacije. Agenti kao što su Harvest, FAQ-Finder, Information Manifold, OCCAM, i ParaSit se oslanjaju na prespecificirane domenske informacije o različitim tipovima dokumenata, ili na kodirane modele informacionih izvora u cilju objašnjenja ili interpretacije dokumenata. Agenti kao što su Shop-Bot i ILA (Internet Learning Agent) međusobno sarađuju radi razvoja strukture nepoznatih informacionih izvora. Shop-Bot otkriva informacije o proizvodima sa različitih sajtova prodavaca korišćenjem uopštenih domenskih informacija o proizvodima. ILA razvija modele različitih informacionih izvora i prevodi ih u svoj koncept hijerarhije.

Filtriranje/ kategorizacija informacija koriste različite tehnike otkrivanja informacija i karakteristike otvorenih hipertekst Web dokumenata da bi automatski otkrili informacije, izvršili njihovo filtriranje, a zatim i kategorizaciju. HyPursuit koristi semantičke informacije obuhvaćene strukturama linkova i sadržajem dokumenata da bi kreirao klaster hijerarhiju hipertekstualnih dokumenata i njihovu strukturu u informacionom prostoru . BO (Bookmark Organizer) kombinuje tehnike hijerarhijskog klasterovanja i interakcije korisnika da bi organizovao kolekciju Web dokumenata zasnovanu na konceptualnim informacijama.

Lični Web agenti uočavaju prioritete korisnika i otkriva izvore informacija na Webu koji su zasnovani na tim prioritetima. Neki od primera ovih agenata su WebWatcher, PAINT, Syskill & Webert, GroupLens, Firefly i drugi. Na primer, Syskill & Webert koristi profile korisnika za procenu interesantnih Web strana korišćenjem Bayes-ove klasifikacije.

Pristup zasnovan na bazama podataka se fokusiraju na tehnike za organizovanje polu-strukturiranih podataka na Webu u strukturirane kolekcije resursa unutar Web dokumenata, kao i na korišćenje standardnih mehanizama upita nad bazom podataka i korišćenje Data Mining tehnika za analizu njihovih rezultata. Mogu se svrstati u dve kategorije: višenivovske baze podataka I websisteme upita.

Višenivovske baze podataka gde niži nivoi baza podataka sadrže polu-strukturirane informacije kao hipertekstualne dokumente uskladištene u različitim Web skladištima. Na višim nivoima su meta podaci, ili generalizacije, koji su ekstrahovani iz nižih nivoa i organizovani u strukturirane kolekcije, tj. relacione ili objektno-orijentisane baze podataka. Na primer, Han koristi višenivovsku bazu podataka u kojoj je svaki nivo prikazan preko generalizacija i operacija transformacije izvedenih preko nižih nivoa. Khosla predlaže stvaranje i održavanje meta baza podataka za svaki informacioni domen i korišćenje šeme za svaku meta bazu podataka. King & Novak predlažu postepenu integraciju delova šema za svaki informacioni izvor, da bi se omogućilo stvaranje šeme globalne heterogene baze podataka.

Web sistemi upita podržava standardne upitne jezike nad bazama podataka, kao što je SQL, strukturirane informacije o Web dokumentima.

Otkrivanje strukture veza na Web-u(Web Structure Mining)

U pristupu zasnovanom na bazama podataka u Web Content Mining-u, predmet interesovanja je struktura unutar Web dokumenata (intra-dokumentna struktura), dok je u Web Structure Mining-u predmet interesovanja struktura hiperlinkova unutar samog Web-a (inter-dokumentna struktura). Ova oblast istraživanja je inspirisana analizama društvenih mreža i analizama citata (study of social networks and citation analysis). Pomoću analiza društvenih aspekata mreža možemo otkriti specifične vrste Web strana koje se zasnivaju na ulaznim i izlaznim linkovima ("incoming and outgoing links"). Web Structure Mining koristi strukture hiperlinkova na Web-u da bi primenio analize društvenih mreža na model osnovne strukture linkova na samom Web-u.

Otkrivanje obrazaca u korišćenju Web-a (Web Usage Mining)

Web Usage Mining predstavlja automatizovano otkrivanje obrazaca u korisničkim pristupima Web serverima.Organizacije prilikom obavljanja svakodnevnih operacija prikupljaju ogromne količine podataka koje Web serveri automatski generišu i prikupljaju u logovima za pristup serveru. Druge izvore korisničkih informacija predstavljaju referencirani logovi ("referrer logs") koji sadrže informacije o referenciranim stranama za svaku referencu na strani, zatim informacije o registraciji korisnika ili pregledu prikupljenih podataka preko CGI skripta. Analiza ovakvih podataka može pomoći organizacijama u određivanju životnog ciklusa potrošača, životnog ciklusa proizvoda, u formulisanju marketing strategije, u efektivnijoj promotivnoj kampanji i u nizu drugih slučajeva. To takođe može obezbediti informacije o tome kako treba prestrukturirati Web sajt organizacije u cilju kreiranja efikasnijeg prisustva organizacije na Web-u, ali i baciti akcenat na efikasnijeg upravljanje komunikacijama između radnih grupa i organizacionom infrastrukturom. Za potrebe reklamiranja prodaje na WWW-u, analiza obrasca u korisničkim pristupima serveru može pomoći u kreiranju reklama za specifične grupe korisnika. Mnoštvo postojećih alata za Web analize obezbeđuje mehanizme za izveštavanje o korisničkim aktivnostima na serverima, kao i mehanizme za različite oblike filtriranja podataka. Korišćenje takvih alata omogućava određivanje broja pristupa serverima i pojedinačnim fajlovima, određivanje vremena pristupa, kao i naziv domena i URL-a korisnika. Ipak, ovi alati su dizajnirani u cilju smanjivanja manipulacija u poslovanju na Web-u, pa obično omogućavaju male, ili skoro nikakve, analize povezanosti podataka preko pristupnih fajlova i direktorijuma unutar Web prostora.

Alati za otkrivanje obrazaca

U traganju za znanjem unutar kolekcija podataka, najnoviji alati za otkrivanje korisnih obrazaca koriste sofisticirane tehnike iz inteligentnih informacionih sistema, Data Mining-a, psihologije i teorije informacija.

Na primer, sistem WEBMINER, uvodi uopštenu arhitekturu za Web Usage Mining. WEBMINER automatski otkriva asocijativna pravila i sekvencijalne obrasce iz pristupnih logova na serveru.

On može biti korišćen za izvođenje različitih vrsta analiza korisničkih putanja, kao što je, na primer, analiza najposećenijih putanja na Web lokacijama.

Dizajniran je i novi alat za Data Mining Web log fajlova, WebLogMiner, koji koristi tehnike Data Mining-a i OLAP-a (On-Line Analytical Processing) u obradi i transformaciji Web log fajlova. Primena Data Mining tehnika na log fajlove omogućava otkrivanje interesantnih obrazaca u log fajlovima koji se mogu koristiti u prestrukturiranju sajtova, u njihovom efikasnijem dizajniranju, u određivanju lokacija za reklame i primamljivanju specifičnih kupaca na kupovinu proizvoda.

Alati za analizu obrazaca

Kada su obrasci u korisničkim pristupima Webu otkriveni, analitičarima su neophodni i odgovarajući alati i tehnike za razumevanje, vizualizaciju i interpretaciju tih obrazaca, kao što je npr. WebViz sistem. Neki predlažu korišćenje OLAP tehnika, kao što su "trodimenzionalne strukture" podataka (“data cubes”) u cilju pojednostavljenja analize korisničkih logova pri pristupu serverima .

Sistem WEBMINER predlaže jedan mehanizam upita, nalik na SQL, za analizu otkrivenog znanja (u obliku asocijativnih pravila i sekvencijalnih obrazaca). Predlaže se i korišćenje analize pojedinačnih trendova (analysis of individual trends). Cilj ove analize je prilagođavanje sajtova korisnicima. Prikazane informacije, dubina sajta i format resursa mogu biti dinamički prilagođeni svakom korisniku na osnovu otkrivenih obrazaca u njihovim pristupima serveru. Ova analiza zahteva različite aplikacije za analizu Web log fajlova. Važno je znati da uspeh svake aplikacije zavisi od toga kakvo i koliko validno i pouzdano znanje se može otkriti iz velikih log fajlova. Međutim, ipak je za jednu efikasnu analizu obrazaca naeophodno prethodno izvršiti predprocesiranje i transformaciju podataka.

Otkrivanje obrazaca u Web transakcijama

Analiza korisničkih pristupa sajtovima je kritična za određivanje efikasne marketing strategije i optimizaciju logičke strukture Web sajta. Zbog mnogih jedinstvenih karakteristika modela klijent – server na WWW-u, uključujući razlike između fizičke topologije Web skladišta i razlike između korisničkih pristupnih putanja, kao i razlike u jedinstvenoj identifikaciji korisnika – kao što su korisničke sesije, ili transakcije, neophodno je razviti novi okvir koji će omogućiti proces traganja na Web-u. Postoji mnoštvo zadataka u predprocesiranju podataka koji moraju biti završeni pre nego što bi algoritmi za otkrivanje obrazaca bili startovani. To zahteva razvoj modela podataka unutar pristupnih logova, razvoj tehnika za prečišćavanje/ filtriranje podataka u cilju eliminisanja izuzetaka i/ili irelevantnih podataka, grupisanje pojedinačnih Web strana u semantičke jedinice (tj. transakcije), integraciju različitih izvora podataka kao što su informacije o registraciji korisnika, i razvoj generičkog Data Mining algoritma koji bi bio prilagođen specifičnoj prirodi podataka u logovima za pristup Internetu.

Zadaci predprocesiranja

Prvi zadatak predprocesiranja je prečišćavanje podataka. Tehnike prečišćavanja logova servera, u cilju eliminacije irelevantnih podataka, su od velikog značaja za svaku vrstu Web log analize, a ne samo za Data Mining. Otkrivene asocijacije ili statističke zavisnosti su korisne jedino ako podaci koji se nalaze u logovima servera daju odgovarajuću sliku korisničkih pristupa Web sajtu. Eliminisanje irelevantnih podataka može biti veoma uspešno izvršeno proveravanjem sufiksa URL naziva. Na primer, u svim logovima se sa nazivom fajla upisuju i sufiksi kao što su: gif, jpeg, GIF, JPEG, jpg, JPG, pa ovakve mape lako mogu biti uklonjene.

U vezi s tim je i mnogo teži problem određivanja veoma važnih pristupa serveru koji nisu snimljeni u pristupnom logu. Mehanizmi kao što su lokalno keširanje i "proxy" serveri mogu ozbiljno iskriviti globalnu sliku korisničkih putanja kroz Web sajtove. Odgovarajući mehanizmi za savladavanje ovog problema koriste "cookie" fajlove i eksplicitne registracije korisnika. Neke od ovih metoda imaju veoma ozbiljne poteškoće. "Cookie" fajl može biti obrisan od strane korisnika, a registracija korisnika je dobrovoljna, pa korisnici često daju netačne informacije. Međutim, zadovoljavajuće rezultate daju metode za rad sa problemima keširanja koje koriste topologiju sajtova ili referenciranih logova, zajedno sa privremenim informacijama, u cilju zaključivanja o nedostajućim referencama.

Drugi problem povezan sa "proxy" serverima je u identifikaciji korisnika. Korišćenje imena mašine za jedinstvenu identifikaciju korisnika može izazvati to da je nekoliko korisnika pogrešno grupisano zajedno kao jedan korisnik. Ako Web strana koja je zahtevana nije direktno povezana sa prethodnom stranom, sigurno je da na toj mašini postoje višestruki korisnici. U drugim algoritmima se za identifikaciju korisnika koristi automatski određena dužina korisničkih sesija, zasnovana na navigaciji obrasca. Druge heuristike obuhvataju korišćenje kombinacija IP adresa, imena mašina, agenata pretraživača, kao i temporarnih informacija za identifikaciju korisnika.

Drugi, glavni zadatak predprocesiranja je identifikacija transakcija. Pre nego što se izvrši bilo kakvo otkrivanje korisnih podataka na Web-u, nizovi referenciranih Web stranica moraju biti grupisani u logičke jedinice koje predstavljaju Web transakcije ili korisničke sesije. Korisničku sesiju čine sve Web stranice koje je korisnik posetio za vreme poslednje posete sajtu. Identifikacija korisničkih sesija je problem sličan problemu identifikacije individualnih korisnika, koji je prethodno opisan. Razlika transakcije i korisničke sesije je u tome što se veličina transakcije može kretati od jedne referencirane strane do svih strana referenciranih u korisničkoj sesiji, zavisno od korišćenog kriterijuma za identifikaciju transakcije. Nasuprot tradicionalnim domenima za Data Mining, kao što su velike baze podataka o prodaji, ne postoji konvencionalna metoda za klasterizaciju referenciranih Web stranica unutar transakcije manje od jedne cele korisničke sesije.

Tehnike otkrivanja obrazaca u Web transakcijama

Kada je identifikovana jedna korisnička transakcija ili sesija, na raspolaganju je nekoliko načina

otkrivanja obrazaca u korisničkim pristupima, koji mogu biti korišćeni zavisno od potreba analize, kao što su: analiza putanja, otkrivanje asocijativnih pravila ili sekvencijalnih obrazaca, klasterizacija ili klasifikacija.

Postoji mnogo različitih vrsta grafova koji mogu biti formirani za izvođenje analize putanja ("path analysis"), pošto grafovi najbolje reprezentuju neke relacije definisane na Web stranicama (ili drugim objektima). Najznačajnije je to što graf reprezentuje fizički raspored Web sajtova, sa Web stranama kao čvorovima i hipertekst linkovima između strana kao granama. Neki grafovi mogu biti formirani na osnovu tipova Web strana, sa granama koje predstavljaju sličnost između strana, ili kreiranjem grana koje daju broj korisnika koji prelaze sa jedne Web strane na drugu .

Mnoštvo radova obuhvata određivanje obrazaca čestih prelazaka sa sajta na sajt, ili velikih sekvenci referenci na sajtove, kroz fizički raspored unutar grafa. Analiza putanja može biti korišćena za određivanje najfrekventnije posećenih putanja na Web sajtu. Analizom putanja direktno mogu biti otkriveni razni primeri korisnih informacija.

Uopšteno rečeno, tehnike otkrivanja asocijativnih pravila se primenjuju nad bazama podataka o transakcijama, gde se svaka transakcija sastoji od skupa članova. U tom kontekstu, problem se sastoji u otkrivanju asocijacija i korelacija između podataka o članovima transakcija, gde prisustvo jednog skupa članova u transakciji implicira prisustvo drugih članova. U kontekstu Web Usage Mining-a, ovaj problem je istovetan problemu otkrivanja korelacija između referenci na različite fajlove raspoložive korisnicima servera. Svaka transakcija obuhvata skup URL-ova kojima je korisnik pristupio prilikom jedne posete serveru.

Pošto obično takva baza podataka o transakcijama sadrži ekstremno veliki skup podataka, odgovarajuće tehnike otkrivanja asocijativnih pravila pokušavaju da smanje istraživački prostor, da bi obezbedile podršku za članove koje je neophodno uzeti u obzir prilikom razmatranja. Podrška je mera zasnovana na broju slučajeva korisničkih transakcija unutar logova servera.

Organizacijama angažovanim u elektronskoj trgovini otkrivanje asocijativnih pravila može pomoći u razvoju efikasne marketing strategije. Ali, otkrivena asocijativna pravila iz pristupnih logova WWW-u mogu ukazati organizacijama i na to kako najbolje da organizuju svoj Web prostor.

Problem otkrivanja sekvencijalnih obrazaca se sastoji u nalaženju obrasca između transakcija, ali tako što je prisustvo u skupu članova određeno vremenskim obeležjima skupa transakcija. U logovima transakcija Web servera, posete klijenata se snimaju po vremenskim periodima. Vremensko obeležje dodeljeno transakciji u ovom slučaju može biti vremenski interval koji je determinisan i vezan za transakciju tokom predprocesiranja podataka, ili tokom procesa identifikacije transakcije. Otkrivanje sekvencijalnih obrazaca u pristupnim logovima Web serveru, omogućava organizacijama koje svoje poslovanje zasnivaju na Web-u da predvide obrasce u posetama korisnika njihovim Web sajtovima, ali i prilagođavanje reklamnih ciljeva grupama korisnika na osnovu otkrivenih obrazaca. Analizom ovakvih informacija, Web mining sistem može odrediti privremene veze između podataka.

Druga važna vrsta zavisnosti između podataka koja može biti otkrivena korišćenjem privremenih

karakteristika podataka je vremenski redosled. Na primer, može nas interesovati nalaženje zajedničkih karakteristika klijenata koji su posetili pojedini fajl u toku određenog vremenskog perioda.

Otkrivanje klasifikacionih pravila omogućava razvoj profila članova koji pripadaju pojedinim grupama, na osnovu njihovih zajedničkih atributa. Ovaj profil može kasnije biti korišćen za klasifikaciju novih podataka članova koji se dodaju u bazu podatka. U Web Usage Mining-u, tehnike klasifikacije omogućavaju razvoj profila korisnika koji posećuju pojedine fajlove na serveru, zasnovanog na demografskim informacijama raspoloživim o datom klijentu, ili zasnovanog na obrascima u pristupima Web serveru. Na primer, klasifikacija pristupnih logova WWW-u može voditi otkrivanju veza.

Klaster analiza otkriva grupe klijenata, ili podatke o članovima koji imaju slične karakteristike. Klasterizacija informacija o klijentima, ili podataka o članovima, u logovima transakcija na Web-u, može olakšati razvoj buduće marketing strategije, bilo on-line ili off-line, kao što je automatsko vraćanje mail-a klijentima koji pripadaju odgovarajućem klasteru, ili dinamičko menjanje pojedinih sajtova za klijente, na osnovu prethodno izvršene klasifikacije podataka datog klijenta.

Analiza otkrivenih obrazaca

Obrasci koji su otkriveni na Web-u pomoću ranije opisanih tehnika, neće biti veoma korisni analitičarima ukoliko ne postoje mehanizmi i alati koji će im pomoći u njihovom boljem razumevanju. Pored razvoja tehnika za otkrivanje korisnih obrazaca unutar Web logova, postoji i potreba za razvojem tehnika i alata koji omogućavaju analizu otkrivenih obrazaca. Od ovih tehnika se očekuje statističko predstavljanje brojnih podataka, korišćenje grafike, vizualizacija, primena analize ponašanja, kao i korišćenje upita nad bazama podataka. Analiza ponašanja korisnika prilikom pristupa Web-u je veoma nova oblast istraživanja, o kojoj je urađeno veoma malo radova, tako da i odgovarajući pregled nije naročito obiman.

Vizualizacija se veoma uspešno koristi u cilju pomaganja ljudima da razumeju različite vrste fenomena, kako prirodnih, tako i apstraktnih. Dakle, to je prirodan način za razumevanje ponašanja korisnika Weba. Pitkow je razvio WebViz sistem za vizualizaciju obrazaca u pristupu WWW-u . Predložena je i paradigma Web putanja (“Web path paradigm”) u kojoj je skup logova za pristup serveru korišćen za ekstrakciju podsekvenci u obrascima kretanja po Webu, nazvanim Web putanje. WebViz sistem omogućava analitičaru selektivnu analizu dela Web-a za koji je zainteresovan, bez analize irelevantnih delova. Web je predstavljen kao usmeren graf, pri čemu čvorovi predstavljaju Web strane, a grane hiperlinkove između strana.

OLAP (On-Line Analitycal Processing) se pojavio kao snažna paradigma za strategijske analize baza podataka u poslovnim okruženjima. Nedavno je pokazano da funkcije i zahtevi koji se stavljaju pred OLAP procesiranje zahtevaju dizajniranje nove informacione strukture. To je dalo prednost razvoju “data cube” informacionog modela i tehnika za njegovu efikasnu implementaciju. Međutim, nedavna istraživanja su pokazala da analize koje se zahtevaju u

postupku Web Mining-a imaju mnoštvo zajedničkih karakteristika sa skladištima podataka, pa su i OLAP tehnike sasvim primenljive. Informacije o korisničkim pristupima u logovima servera su modelirane kao jedna “append-only” kolekcija, koja raste tokom vremena. Pošto veličina logova servera rapidno raste, nemoguće je obezbediti njihovu on-line analizu. Stoga, postoji potreba za sumiranjem podataka unutar logova, na različite načine, kako bi njihova on-line analiza postala izvodljiva.

Rečnik

Ad-hoc upit (Ad-Hoc Query): svako spontano i neplanirano pitanje, ili upit. To je upit koji se sastoji od dinamički generisanog SQL-a, koji je obično generisan preko neke desktop alatke.

Agregacija (Aggregation): specijalna forma asocijacija koja specifira odnos između agregacije (celine) i komponentnog dela.

Atribut (Attribute): svojstvo ili karakteristika koja je uobičajena za neke ili sve objekte entiteta. Jedan atribut prikazuje korišćenje domena u kontektsu entiteta.

Baza podataka (Database): Kolekcija podataka koji su u međusobnoj relaciji, često sa kontrolisanom redundansom podataka, organizovanom po modelu (šemi) da koristi jednoj ili više aplikacija.

Data Mining: klasa analitičkih aplikacija koja traži obrasce u bazi podataka. To je proces "prosejavanja" velike količine podataka da bi se dobili podaci od interesa. Ovaj alat koristi različite tehnike, uključujući i rezonovanje na osnovu slučaja, vizuelizacije podataka, upita i analiza.

Domen (Domain): imenovani skup vrednosti podataka istih tipova podataka, preko kojih se formiraju stvarne vrednosti atributa objekata. Svaki atribut može biti definisan pod samo jednim domenom.

Drill Down/Up: tehnika analize koja dopušta korisnicima SPO-a navigaciju nivoima podataka rangiranim od najsumarnijih (up) do najdetaljnijih (down).

Druga normalna forma (Second Normal Form-2NF): entitet je u drugoj normalnoj formi, ako je prvo u prvoj normalnoj formi i ako je svaki atribut koji nije ključ u direktoj zavisnosti sa primarnim ključem.

Egzistencijalna zavisnost (Existence Dependency): uslov između dva entiteta u relaciji, koji pokazuje da ne može postojati objekat jednog entiteta koji nije u relaciji sa objektima drugog entiteta.

Element podatka (Data Element): najelementarnija jedinica podataka koja može da bude prepoznata i opisana u rečniku ili skladištu i koja ne može dalje da bude dekomponovana.

Enitet roditelj (Entity Parent): entitet čiji objekti mogu da budu u vezi sa više objekata drugog entiteta (entiteta dete).

Entitet dete (Entity Child): entitet u specifičnoj povezujućoj relaciji, čiji objekti mogu biti u vezi sa nula ili jednim objektom drugog entiteta (roditelja).

Entitet (Entity): prezentacija realnih i apstraktnih stvari (ljudi, objekata, slučaja...) koji se prepoznaju pod istim tipom podataka, jer dele iste karakteristike i mogu učestvovati u istim relacijama.

Funkcionalna zavisnost (Functional Dependency): veza entiteta kojom se opisuje uslov "bar jedan".

Grafički korisnički interfejs (Graphical User Interface – GUI): Programski interfejs koji koristi grafičke mogućnosti računara u cilju olakšavanja rada pri upotrebi računara. Grafički interfejs koristi pokazivačke uređaje za selektovanje objekata, uključivanje ikona, menija, tekst boksova itd.

Identifikator zavisnosti (Identifier Dependency): iskaz između dva entiteta u vezi koji zahteva da primarni ključ u jednom (entitetu detetu) sadrži primarni ključ drugog (entiteta roditelja).

Ime uloge (Role Name): ime dodeljeno prenesenom ključu i predstavlja upotrebu prenesenog ključa u entitetu.

Informacija (Information): podatak koji se obrađuje radi dobijanja nekog značenja i znanja za osobu koja je prima. Ona je izlaz iz informacionog sistema.

Kategorija entiteta (Entity Category): entitet čiji se objekti prikazuju podtipom i potklasifikacijom drugog entiteta (podtip, potklasa).

Klijent/server arhitektura (Client/server architecture): mrežna arhitekura u kojoj računari na mreži učestvuju kao serveri u upravljaju podacima i servisima mreže, ili kao klijenti, gde korisnici pokreću aplikacije i pristupaju serveru.

Ključ, Kandidat (Key, Candidate): atribut, ili kombinacija atributa entiteta čije vrednosti jednoznačno određuju sve objekte entiteta.

Ključ, Opcioni (Key, Alternate): svaki ključ kandidat koji nije primarni ključ.

Ključ, Preneseni (Key, Foreign): atribut ili kombinacija atributa deteta, ili nekog drugog entiteta čija se vrednost primarnog ključa poklapa sa vrednošću primarnog ključa entiteta roditelja.

Ključ, Primarni (Key, Primary): kandidatski ključ koji jednoznačno identifikuje entitet.

Ključ, Složeni (Key, Composite): ključ sastavljen od dva ili više atributa.

Korisnički interfejs (User Interface): komponenta računarskog sistema za podršku u odlučivanju koja omogućava bidirekcionu (dvosmernu) komunikaciju između sistema i korisnika.

Metapodaci (Metadata or Meta Data): podaci o podacima u skladištu podataka. Pomažu u

definisanju sadržaja skladišta podataka. To su semantičke informacije odgovarajućih promenljivih. Moraju da uključuju poslovne definicije podataka, tačne opise tipova podataka, potencijalne vrednosti, originalni izvorni sistem, formate podataka i druge karakteristike. Definišu i opisuju poslovne podatke. Sadrže stvari poput imena, dužine, validne vrednosti i opisa podataka nekog podatka elementa. Čuvaju se u rečniku podataka. Izoluju skladište podataka od promena usled rada pod nekim operativnim sistemom.

Model podataka (Data Model): grafička i tekstualna prezentacija analize koja identifikuje podatke koji su potrebni organizaciji koja učestvuje u poslu. Prezentuje entitete, domen (atribute) i relacije sa drugim podacima i konstruiše konceptualni pogled podataka i relacija između podataka.

N-arna asocijacija (N-ary Association): asocijacija preko tri ili više klasa. Svaka instanca asocijacije je n-ta vrednost odgovarajuće klase. Suprotno: binarna asocijacija.

Normalizacija (Normalization): proces redefinisanja i regrupisanja atributa u entitetima, u skladu sa normalnom formom.

Normalna forma (Normal Form): stanje entiteta koje relativno zadovoljava skup normalizacija njegovih atributa. Specifična normalna forma je izvedena sukcesivnom redukcijom entiteta iz njegovog izvornog stanja u neki željeni oblik forme. Procedura je reverzibilna.

Nul (Null): stanje gde vrednost nekog atributa nije poznata za neki objekat entiteta.

Ograničenje egzistencije (Constraint, Existence): uslov gde objekti jednog entiteta ne mogu da postoje ukoliko ne postoje objekti entiteta sa kojim je ovaj u relaciji.

Ograničenje kardinalnosti (Constraint, Cardinality): ograničenje broja objekata entiteta koje može da bude asocirano u relaciji.

Ograničenje (Constraint): pravilo koje pokazuje validnost stanja podataka.

On-line Analytical Processing (OLAP): softver koji se koristi za rad sa višedimenzionalnim podacima iz različitih izvora koji se smeštaju u skladište podataka. Formira različite poglede na podatke. Omogućuje brži, sadržajniji i interaktivniji pristup višedimenzionalnim podacima.

Osnovni entitet, generički (Entity Generic): entitet čiji su objekti klasifikovani u jedan ili više podtipova ili potklasa (supertip, superklasa).

Podaci (Data): Binarna (digitalna) prezentacija atomskih činjenica, teksta, grafika, bit mapa, zvuka, analognih ili digitalnih video segmenata. Podak je sirovina sistema koju ovaj dobija preko procedura i koja se koristi radi kreiranja informacija.

Poslovne transakcije (Business Transaction): to je jedinica posla nad strukturama podataka u cilju kreiranja, modifikacije ili brisanja poslovnih podataka. Svaka transakcija predstavlja jednu vrednovanu činjenicu, koja opisuje jedan poslovni slučaj.

Poslovni model (Business Model): u skladištu podataka, to je dizajnerski pogled na to kako posao funkcioniše. Pogled može biti sa aspekta posla, podataka, slučaja ili resursa i može da

bude o prošlom, sadašnjem ili budućem stanju posla.

Poslovni podaci (Business Data): podaci o ljudima, mestima, stvarima, poslovnim pravilima i slučajevima koji se upotrebljavaju pri vođenju posla. Nisu metapodaci.

Pravilo (Rule): formalni pristup specifičnim preporukama, direktivama ili strategiji, iskazanim kroz IF-THEN konstrukcije.

Prva normalna forma (First Normal Form – 1NF): entitet je u prvoj normalnoj formi ako su njegov sadržaj samo atomske vrednosti.

Rapidni razvoj aplikacija (Rapid Application Development – RAD): deo metodologije koji navodi na inkrementalni razvoj uz podršku naručioca. Cilj je da razvoj projekta ostane usredsređen na stalno ostvarivanje komunikacije. Jedino ograničenje pri ovakvom radu jeste različitost govora lica koja su u komunikaciji.

Rečnik podataka (Data Dictionary): baze podataka o podacima i strukturama.

Sistem (System): kolekcija povezanih jedinica koje su organizovane da izvršavaju određenu svrhu. Sistem može biti opisan jednim modelom ili sa više njih, najverovatnije sa različitih aspekata posmatranja.

Skalabilnost (Scalability): sposobnost raslojavanja hardvera i softvera radi podržavanja većih ili manjih količina podataka i više ili manje korisnika.

Skladište podataka (Data Warehouse): skladište podataka je objektno orijentisana, integrisana, vremenski promenljiva, postojana kolekcija podataka u upravljanju procesima donošenja odluke.

Tip podataka (Data Type): kategorizacija apstraktnog skupa vrednosti, karakteristika i skupa operacija koji se odnose na atribute. Celi brojevi, realni brojevi, znakovni tipovi podataka itd.

Transakcione baze podataka (Operational or Transaction Database): baze podataka za transakciju (razmenu) podataka. One su izvor za skladišta podataka.

Treća normalna forma (Third Normal Form-3NF): jedan entitet je u trećoj normalnoj formi najpre ako je u drugoj normalnoj formi, i ako svaki atribut koji nije ključ nije u direktnoj zavisnosti od primarnog ključa.

Upit (Query): postavljanje pitanja (kriterijuma). Obično koristi složene SQL konstrukcije.

Višedimenzionalna baza podataka (Multi dimensional Database – MDBS): baza podataka koja omogućava korisnicima analize velikih količina podataka. Predstavlja podatke kao nizove koji su organizovani u višestruke dimenzije. Promenljive su objekti koji se čuvaju u višedimenzionalnim bazama. To su jednostavni nizovi vrednosti (numeričkih najčešće) koji su dimenzionisani po dimenzijama u bazi podataka. Može da ima višestruke promenljive, sa različitim ili jedinstvenim skupom dimenzija. Ovaj višedimenzionalni pogled na podatke naročito je važan za OLAP aplikacije.

Životni ciklus razvoja sistema (Systems Development Life Cycle-SDLC): proces sistemske

analize, softverskog inženjeringa, programiranja i korisničke izgradnje sistema.

Šema (Schema): definicija strukture podataka.

Skraćenice

1NF	First Normal Form
2NF	Second Normal Form
3NF	Third Normal Form
ADO (MD)	ActiveX® Data Objects (MultiDimensional)
ANSI	American National Standards Institute
API	Application Programming Interface
ASP	Active Server Pages
B2B	Business to Business
B2C	Business to Consumer
B2E	Business to Employee
BDC	Backup Domain Controller
BLOB	Binary Large OBject
BPR	Business Process Reengineering
BPwin	Business Process for Windows
CASE	Computer Aided System Engineering
CGI	Common Gateway Interface
CLI	Call-Level Interface
CLR	Common Language Runtime
COM	Component Object Model
CPU	Central Processing Unit
CSF	Critical Success Factors
DAO	Data Access Object
DBCC	Database Consistency Checker
DBCS	Double-Byte Character Set
DBMS	DataBase Management System
DCL	Data Control Language
DDL	Data Definition Language
DLL	Dynamic-Link Library
DM	Data mining
DML	Data Manipulation Language

DMO	Distributed Management Objects
DPC	Deferred Process Call
DRI	Declarative Referential Integrity
DSN	Data Source Name
DSO	Decision Support Objects
DSS	Decision Support Systems
DTS	Data Transformation Services
DW	Data Warehouse
EC	Electronic Commerce
EDI	Electronic Data Interchange
EDMS	Electronic Document Management System
EIS	Executive Information Systems
ERwin	Entity Relationships for Windows
ESS	Executive Support Systems
FAT	File Allocation Table
FIPS	Federal Information Processing Standard
FK	Foreign Key
GDSS	Group Decision Support Systems
GIS	Geographic Information Systems
GUI	Graphical User Interface
GUID	Globally Unique IDentifier
HOLAP	Hybrid OLAP
I/O	Input/Output
ID	IDentification
IDEF0	Integration Definition Functional Modeling
IE	Information Engineering
IEC	International Electrotechnical Commission
IPC	InterProcess Communication
IPX/SPX	Internet Packet eXchange/Sequenced Packet eXchange
ISO	International Organization for Standardization
ISQL	Interactive Structured Query Language
KE	Knowledge Engineering
KM	Knowledge Management
LAN	Local Area Network
MAPI	Messaging Application Programming Interface
MCP	Microsoft Certified Professional
MCSD	Microsoft Certified Solution Developer
MCSE	Microsoft Certified System Engineer
MCT	Microsoft Certified Trainer
MDX	MultiDimensional Expression

MFC	Microsoft Foundation Class
MMC	Microsoft Management Console
MOC	Microsoft Official Curriculum
MOLAP	Multidimensional OLAP
MS DTC	MicroSoft Distributed Transaction Coordinator
NGWS	Next Generation Windows Services
NIC	Network Interface Card
NIST	National Institute of Standards and Technology
NTFS	Windows NT file system
ODBC	Open DataBase Connectivity
ODS	Open Data Services
OEM	Original Equipment Manufacturer
OLAP	OnLine Analytical Processing
OLE (DB)	Object Linking and Embedding (DataBase)
OLTP	Online Transaction Processing
PDC	Primary Domain Controller
PK	Primary Key
POSIX	Portable Operating System Interface for Unix
RAD	Rapid Application Development
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RDBMS	Relational DataBase Management System
RDO	Remote Data Object
RI	Referential Integrity
ROLAP	Relational OLAP
RPC	Remote Procedure Call
SDLC	Systems Development Life Cycle
SID	Security IDentifier
SMP	Symmetric MultiProcessor
SNA	Systems Network Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SQL-DMF	SQL Distributed Management Framework
SQL-DMO	SQL Distributed Management Objects
TCP/IP	Transmission Control Protocol / Internet Protocol
TDS	Tabular Data Stream
TQM	Total Quality Management
UDDI	Universal Description Discovery & Integration
UML	Unified Modeling Language
UNC	Universal Naming Convention

VINES/IP	VIrtual NEtworking Software / Internet Protocol
WAN	Wwide Area Network
Windows DNA	Windows Distributed interNet Applications Architecture
WSDL	Web Service Description Language
XML	Extensible Markup Language

Literatura

1. ERwin, Methods Guide, 1995.
2. IT E 04-2-1, Sistemska analiza, podsetnik Intertrade, TOZD zastupništvo IBM, Izobrazevalni centar, Ljubljana, Ver.1.0, 1984.
3. IT FA 40-2-1, Planiranje informacionih sistema, podsetnik Intertrade, TOZD zastupnistvo IBM, Izobrazevalni centar, Ljubljana, Ver.1.0, 1984.
4. Lazarević B., Baze podataka, FON, Beograd, 2003.
5. Milačić V., Sistem Analiza, Proizvodni informacioni sistem, Institut Mašinskog fakulteta, Odeljenje za primenu kompjutera, Beograd, 1974.
6. User Guide for ERwin.
7. Veljović A. Elementi osiguranja kvaliteta u ambijentu osvajanja novog proizvoda korišćenjem CASE alata, 21 JUPITER konferencija sa međunarodnim učešćem, 1. simpozijim KVALITET, strana 5.97-5.103., Beograd, februar 1995.
8. Veljović A. Gotova rešenja upravljanja kvalitetom za JUS ISO 9004 orjentisanih računarskoj obradi podataka. Seminar, Jugoslovenska organizacija za standardizaciju i kvalitet (JUSK), Beograd, 14 - 15. mart 1995. godine.
9. Veljović A. i dr., Detaljni projekat informacionog sistema programa A-85, interni materijal, 1990.
10. Veljović A. i dr., Idejni projekat informacionog sistema programa A-85, interni materijal, 1989.
11. Veljović A. i dr., Projekat revizije po standardu ISO 9000:2000 i veza sa informacionim sistemom, Sojaprotein, Bečej, 2000.godina
12. Veljović A. Kako definisati informacije potrebne računaru vezane za postavljanje sistema kvaliteta serije standarda JUS ISO 9000. Seminar, Jugoslovenska organizacija za

standardizaciju i kvalitet (JUSK), Beograd, 20 - 21. decembar 1994. godine.

13. Veljović A. Naučite da kreirate informacioni sistem za upravljanje kvalitetom, Informativno instruktivni seminar, Savez inžinjera i tehničara Jugoslavije, Beograd, 22 - 23. juna 1995. godine.
14. Veljović A., Avram S., MS Access, Računari, Beograd, 1997. godina
15. Veljović A., Dekomponovanje procesa petlje kvaliteta, Časopis za unapređenje kvaliteta Kvalitet, broj 5-6, strana 31-33, Poslovna politika, Beograd, 1995.
16. Veljović A., Menadžment informacioni sistemi, Kompjuter biblioteka, Čačak, 2002. godina
17. Veljović A., Menadžmet razvojem, Tehnički fakultet, Čačak, 2003. godina
18. Veljović A., Modeliranje informacionih sistema, Megatrend, Beograd, 2002. godina
19. Veljović A., Njeguš A., Praktikum iz poslovnih računarskih aplikacija, Megatrend, Beograd, 2004.