BAZE PODATAKA

Jelena Žakula

Metodika nastave računarstva A, 2025/2026

Sana Stojanović Đurđević

Sadržaj

1. Rad sa tabelama i veze između njih

Tabele predstavljaju osnovnu komponentu u relacionim bazama podataka. One omogućavaju organizaciju i skladištenje podataka na način koji olakšava upravljanje podacima i dobijanje korisnih informacija.

1.1. Uređenje polja i slogova u relaciji

U relacionim bazama podataka, svaka tabela se sastoji od polja i slogova. Polja definišu tipove podataka koje tabela može da sadrži, dok slogovi predstavljaju pojedinačne unose podataka. Pravilno uređenje polja uključuje odabir odgovarajućih tipova podataka, kao i postavljanje ključeva i indeksa radi brže i efikasnije pretrage. Slogovi se unose tako da budu konzistentni i u skladu sa definisanim tipovima polja.

Svi slogovi sadrže ista polja, koja se u svakom redu pojavljuju u istom redosledu, dok svako polje ima jedinstven naziv i unapred definisan tip podataka, kao što su numerički, tekstualni, logički ili datumski tip. Podaci unutar jednog polja moraju pripadati istom domenu, odnosno skupu dozvoljenih vrednosti za taj atribut.

Jedno od osnovnih pravila relacionog modela jeste da se u tabeli ne mogu nalaziti dva potpuno ista sloga. Ova jedinstvenost se ostvaruje uvođenjem primarnog ključa, koji predstavlja atribut ili skup atributa čije vrednosti jednoznačno identifikuju svaki slog u tabeli. Primarni ključ može biti prost, ukoliko ga čini jedan atribut, ili složen, kada se sastoji od više atributa zajedno.

U praksi se kao primarni ključ često koriste jedinstveni identifikatori, poput broja indeksa studenta, registracije vozila itd.

U slučajevima kada ne postoji prirodan atribut koji ispunjava uslove primarnog ključa, sistemi za upravljanje bazama podataka omogućavaju korišćenje veštačkog ključa, najčešće polja tipa *AutoIncrement*, koje se automatski uvećava za svaki novi slog.

Pored primarnog ključa, u tabelama se mogu koristiti i sekundarni ključevi, odnosno indeksi. Indeksi ne moraju imati jedinstvene vrednosti, ali omogućavaju brže pretraživanje i sortiranje podataka, slično indeksima u knjigama.

Pretraga podataka može se vršiti sekvencijalno, čitanjem svakog sloga, ili korišćenjem indeksa, gde se vrednosti čuvaju u dodatnoj strukturi radi efikasnijeg pristupa podacima.

U sledećem primeru tabele, učenik sa atributom SifraUc može da prima samo jednu stipendiju, tako da ovo polje može da bude primarni ključ, i to prost ključ.

SifraUc	**Davalac**	**Iznos**
100	Republika	9000
150	Grad	6000
175	Opština	3000
200	Grad	6000

Tabela 1: Primer tabele sa prostim primarnim ključem

U drugom primeru učenik može da prima više različitih stipendija (ali samo jednu od jednog davaoca stipendije), tako da u ovom slučaju polja SifraUc i Davalac zajedno mogu da čine složeni primarni ključ (učenik sa šifrom 100 može dobiti samo jednu stipendiju od Republike, ne mogu se u tabeli naći dve iste takve vrednosti).

SifraUc	**Davalac**	**Iznos**
100	Republika	9000
100	Grad	6000
150	Grad	6000
175	Opština	3000
175	Grad	6000
200	Grad	6000
200	Republika	9000

Tabela 2: Primer tabele sa složenim primarnim ključem

1.2. Formatiranje podataka u tabeli

Formatiranje podataka u tabeli ima ključnu ulogu u obezbeđivanju preglednosti, konzistentnosti i pravilne interpretacije informacija. Ono obuhvata definisanje tipova podataka za polja, postavljanje širine kolona, kao i izbor odgovarajućeg formata prikaza za numeričke vrednosti, datume i tekstualne podatke. Pravilno formatirana tabela omogućava lakše čitanje podataka i smanjuje mogućnost pogrešne interpretacije.

Pored osnovnog izgleda, formatiranje uključuje i primenu pravila prikaza, kao što su ograničenja unosa, podrazumevane vrednosti i maskiranje podataka. Ovakva pravila doprinose tačnosti unosa i obezbeđuju da se podaci unose u skladu sa unapred definisanim standardima. Dobar dizajn i formatiranje tabele olakšavaju rad sa podacima, unapređuju kvalitet baze i predstavljaju osnovu za efikasnu obradu, pretragu i analizu informacija u relacionim bazama podataka.

1.3. Opis referencijalnog integriteta

U relacionim bazama podataka tabele se često ne posmatraju izolovano, već su međusobno povezane kako bi se izbeglo dupliranje podataka i obezbedila njihova logička povezanost. Takve veze između tabela ostvaruju se pomoću ključeva, pri čemu se najčešće povezuje primarni ključ jedne tabele sa stranim ključem druge tabele. Da bi ove veze bile ispravne i pouzdane, neophodno je obezbediti referencijalni integritet.

Referencijalni integritet predstavlja skup pravila koja garantuju doslednost podataka između povezanih tabela. Njegova osnovna uloga je da obezbedi da svaka vrednost stranog ključa u jednoj tabeli odgovara postojećoj vrednosti primarnog ključa u povezanoj tabeli, ili da bude prazna ukoliko je to dozvoljeno. Na taj način sprečava se pojava nevažećih ili „visećih" zapisa, odnosno slogova koji se pozivaju na nepostojeće podatke u drugim tabelama.

Primena referencijalnog integriteta posebno je važna prilikom povezivanja tabela koje opisuju različite aspekte istog sistema, kao što su npr. tabele učenika i stipendija, kupaca i porudžbina ili proizvoda i magacina. Kada se jednom uspostavi veza između tabela, sistem za upravljanje bazom podataka automatski kontroliše operacije unosa, izmene i brisanja podataka, čime se čuva logička ispravnost baze.

Referencijalni integritet takođe omogućava definisanje dodatnih pravila ponašanja, kao što su kaskadne izmene i kaskadna brisanja. Ova pravila određuju kako će se promene u jednoj tabeli odraziti na povezane zapise u drugim tabelama. Na taj način se obezbeđuje stabilnost baze podataka i olakšava rad sa povezanim tabelama, što predstavlja osnovu za efikasno spajanje tabela, formiranje upita i izradu složenijih struktura u relacionim bazama podataka.

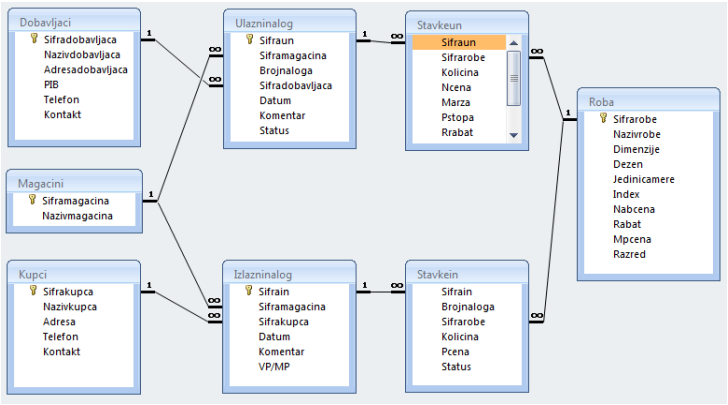
1.4. Izmene veze između tabela

Veze između tabela mogu se menjati dodavanjem, uklanjanjem ili modifikovanjem veza između polja. Tipične veze su *one-to-one*, *one-to-many* i *many-to-many*. Prilikom izmene veza važno je paziti na integritet podataka i primenu odgovarajući.

Relacija jedan prema više se najčešće koristi. Kod ovakve relacije, jednom slogu iz prve tabele odgovaraju više slogova iz druge tabele, i obrnuto svaki slog iz druge tabele ima jedan odgovarajući iz prve tabele. Polje iz prve tabele, koje je u relaciji, mora biti primarni ključ dok je polje iz druge tabele strani ključ. Imena polja za primarni i strani ključ ne moraju biti ista, ali tip podatka mora.

Relacija jedan prema jedan se retko koristi i kod ove relacije svaki slog iz prve tabele ima samo jedan odgovarajući iz druge tabele, i obrnuto. Npr. jedan rukovodilac može obavljati samo jednu funkciju.

Relacija više prema više je dosta komplikovana i takoreći je nemoguća kod relacionog modela baze podataka.



2. Forme (obrasci)

2.0.1 Kreiranje formi (obrazaca) sa i bez *čarobnjaka*

Forme predstavljaju jedan od najvažnijih alata u radu sa bazama podataka jer omogućavaju lakši i pregledniji unos i prikaz podataka. Mogu biti kreirane na više načina: korišćenjem *Wizard* funkcije, tj pomoću čarobnjaka, koja vodi korisnika kroz proces kreiranja, ili ručno, u *Design view*, gde se sve postavke i raspored kontrola podešavaju samostalno. *Wizard* je pogodan za početnike jer automatski dodaje standardne kontrole i postavlja osnovni izgled forme.

U kreiranju forme bez *Wizard-a*, korisnik ima potpunu kontrolu nad rasporedom polja, vrstom kontrola i vizuelnim izgledom forme. Ovo je naročito korisno kada je potrebno napraviti prilagođenu formu koja sadrži složene kontrole ili kada se želi postići specifičan vizuelni identitet baze podataka. Forme kreirane ovako zahtevaju detaljnije planiranje, ali pružaju maksimalnu fleksibilnost.

2.0.2 Unos podataka pomoću formi

Forme služe prvenstveno za unos podataka u bazu. Pomoću povezanih kontrola, korisnik može unositi vrednosti direktno u polja iz osnovne tabele ili *query-a*. Svaka kontrola može biti tipa *text box*, *combo box*, *list box* ili *check box*, zavisno od vrste podataka koji se unose. Prednost formi je u tome što korisnik ne vidi kompletnu strukturu tabele, već

samo relevantna polja, što smanjuje mogućnost greške.

Takođe, forme omogućavaju validaciju podataka i korišćenje izraza (*expressions*) za izračunavanje vrednosti, što olakšava unos i održavanje integriteta baze podataka.

2.0.3 Dodavanje specijalnih kontrola formi

Pored standardnih tekstualnih i numeričkih polja, forme mogu sadržati specijalne kontrole koje olakšavaju rad sa podacima i navigaciju. *List boxes* i *combo boxes* omogućavaju izbor vrednosti iz unapred definisanih lista, dok *command buttons* mogu izvršavati akcije kao što su čuvanje, brisanje ili otvaranje drugih formi. Korišćenje ovih kontrola povećava efikasnost i smanjuje mogućnost greške korisnika.

2.0.4 Kreiranje multitabularnih formi

Multitabularne forme ili višestruke tabele unutar jedne forme omogućavaju pregled i unos podataka iz više tabela ili upita istovremeno. Ovakve forme su korisne za prikaz povezanih podataka, na primer prikaz proizvoda i njihove trenutne količine po magacinima.

Kreiranje ovakvih formi zahteva pravilno povezivanje kontrola sa izvorom podataka i često korišćenje *subforms* za prikaz podataka iz drugih tabela ili upita.

Multitabulare forme poboljšavaju preglednost podataka i omogućavaju ažuriranje povezanih informacija, što je posebno korisno u poslovnim aplikacijama sa velikim bazama podataka.

2.1. Pretraživanje i sortiranje

2.2. Traženje informacija u tabeli

Pretraživanje podataka predstavlja osnovnu funkcionalnost u radu sa bazom podataka. Kroz forme, korisnik može brzo pronaći željene slogove prema različitim kriterijumima. *Search* funkcija omogućava filtriranje podataka po tekstu, brojevima ili datumu, dok *Advanced Filter* omogućava kombinovanje više kriterijuma.

Pravilno postavljeni kriterijumi i indeksiranje tabela omogućavaju brže pronalaženje podataka i optimizuju rad sa velikim bazama.

2.2.1 Sortiranje, filtriranje i indeksiranje

Sortiranje i filtriranje podataka poboljšavaju preglednost i analizu informacija. *Sort* opcija omogućava raspoređivanje slogova po rastućem ili opadajućem redosledu, dok *Filter* omogućava prikaz samo onih slogova koji ispunjavaju određene kriterijume. Indeksiranje tabela dodatno ubrzava ove operacije, jer omogućava brži pristup podacima po definisanim poljima.

Kombinacija sortiranih, filtriranih i indeksiranih tabela je posebno korisna prilikom izveštavanja i vizuelizacije podataka u formama i izveštajima, jer omogućava korisniku da se fokusira na relevantne informacije i olakšava donošenje poslovnih odluka.

3. Upiti

Upiti predstavljaju osnovni način komunikacije sa bazom podataka, omogućavajući korisnicima da pregledaju, dodaju, menjaju i brišu podatke prema svojim potrebama.

Upitni jezik SQL omogućava korisnicima da izvrše sve potrebne operacije nad bazom podataka, od kreiranja same baze i tabela u njoj, upisivanja, menjanja i pretraživanja podataka, do brisanja nekih podataka, tabela ili cele baze.

3.1. Osnovne SQL komande (definicione, kontrolne i manipulacione)

Komande jezika *SQL* se tradicionalno nazivaju upitima. Prva reč u svakom *SQL* upitu određuje vrstu akcije koju korisnik zahteva od sistema. Na osnovu te akcije, *SQL* komande se logički dele u tri osnovna seta: komande za definiciju podataka, komande za manipulaciju podacima i komande za kontrolu pristupa.

Ova podela se u praksi ne ističe posebno, jer se sve naredbe koriste kroz upite na sličan način, ali je veoma značajna za razumevanje njihove namene i uloge u radu sa bazama podataka.

Najčešće korišćeni *SQL* upiti su:

- **Upiti *SELECT***, koji služe za čitanje podataka iz baze,
- **Upiti *INSERT***, koji služe za unos novih podataka u bazu,
- **Upiti *DELETE***, koji služe za brisanje postojećih podataka, i
- **Upiti *UPDATE***, koji služe za izmenu postojećih podataka u bazi.

Od svih navedenih tipova upita, za nas je najznačajniji upit ***SELECT***, pomoću kojeg veoma brzo možemo da dođemo do podataka koji su nam potrebni. Međutim, da bismo u potpunosti razumeli mogućnosti jezika *SQL*, neophodno je sagledati i ostale grupe naredbi.

3.1.1 Set SQL naredbi za kontrolu

Naredbe za kontrolu koriste se za upravljanje pravima pristupa na serveru. One određuju koji korisnici mogu da izvršavaju određene operacije nad bazama i objektima u njima. U ovaj set spadaju naredbe *GRANT* i *REVOKE*.

3.1.2 Set SQL naredbi za definiciju

Set definicionih naredbi koristi se za rukovanje objektima na serveru. U ovaj set spadaju naredbe *CREATE*, *ALTER* i *DROP*. Njima se kreiraju, menjaju ili brišu baze podataka, tabele, pogledi, uskladištene procedure i drugi objekti.

Pošto se ovim naredbama upravlja strukturom baze, one su najčešće dostupne samo administratorima servera. Korisnici obično imaju prava da rade unutar svoje baze, ali ne i da brišu baze ili upravljaju drugim bazama na serveru.

3.1.3 Set SQL naredbi za manipulaciju

Naredbe za manipulaciju podacima predstavljaju srž jezika *SQL*. One se koriste najčešće i u najvećoj količini, jer omogućavaju svakodnevni rad sa podacima. U ovaj set spadaju naredbe *SELECT*, *INSERT*, *UPDATE* i *DELETE*.

Iako je ovaj set naredbi fleksibilniji po pitanju korisničkih prava, neophodno je biti oprezan, jer pogrešna upotreba naredbi *INSERT*, *UPDATE* i *DELETE* može dovesti do ozbiljnih oštećenja podataka.

3.2. Kreiranje upita (sa i bez čarobnjaka)

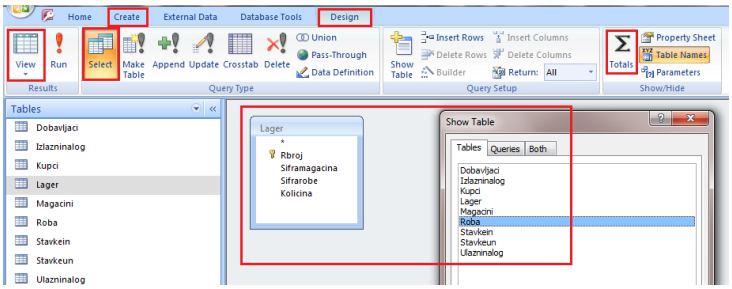
Upiti u sistemu za upravljanje bazama podataka mogu se kreirati na više načina, najčešće pomoću čarobnjaka ili u *design* modu. Čarobnjak pojednostavljuje formulisanje složenih upita, posebno kada je potrebno povezati više tabela ili primeniti filtere i sortiranja.
Kreiranje upita u *design* modu omogućava preciznu kontrolu nad izborom tabela, polja i uslova, ali zahteva poznavanje strukture baze podataka, kao i relacija između tabela. Pre samog kreiranja upita, neophodno je jasno definisati zadatak, odnosno utvrditi koje podatke je potrebno prikazati.

Primer zadatka Prikazati trenutni lager u svim magacinima tako da se vide sledeća polja: *Naziv magacina*, *Šifra robe*, *Naziv robe*, *Dimenzije*, *Dezen*, *Jedinica mere*, *Količina* i *Maloprodajna cena*. Kreirani upit snimiti pod nazivom *Trenutni lager*.

Upit se kreira na tabu *Create* izborom komande *Query Design*. Tom prilikom se otvara prozor *Show Table*, u kojem se biraju tabele potrebne za realizaciju upita. U ovom slučaju, u početnoj fazi bira se tabela *Lager*, dok se tabela *Roba* može dodati naknadno.

Nakon izbora tabele, u donju zonu prozora za dizajn upita prevlače se odgovarajuća polja, kao što su *Šifra magacina*, *Šifra robe* i *Količina*. Polja se mogu dodati prevlačenjem levim tasterom miša ili dvostrukim klikom na njihovo ime.

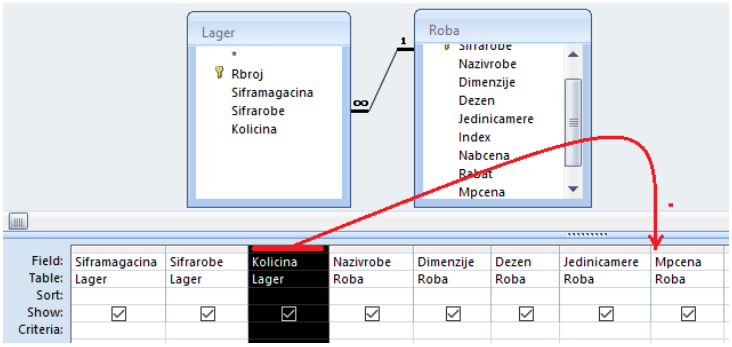
Radi provere ispravnosti upita, koristi se dugme *View*, koje omogućava prikaz rezultata izvršavanja upita. Isto dugme se zatim menja u *Design*, čime se omogućava povratak u režim dizajna. Na taj način moguće je naizmenično prelaziti između *view* i *design* moda i pratiti razvoj i rezultate kreiranog upita.



Nakon izvršavanja upita može se proveriti koliko slogova sadrži rezultat. Dobija se nešto više od 5300 slogova, odnosno isti broj kao u tabeli Lager. Razlog za to je činjenica da u upit nije dodat nijedan kriterijum, dok će kriterijumi biti razmatrani kasnije. U trenutnoj verziji upita još uvek nije jasno o kojim proizvodima je reč.

Zbog toga je potrebno u upit dodati i tabelu Roba, kao i polja Nazivrobe, Dimenzije, Dezen, Jedinicamere i Mpcena. Dodavanje tabele vrši se desnim klikom na pozadinu upita, nakon čega se bira opcija *Show Table* i dodaje tabela Roba.

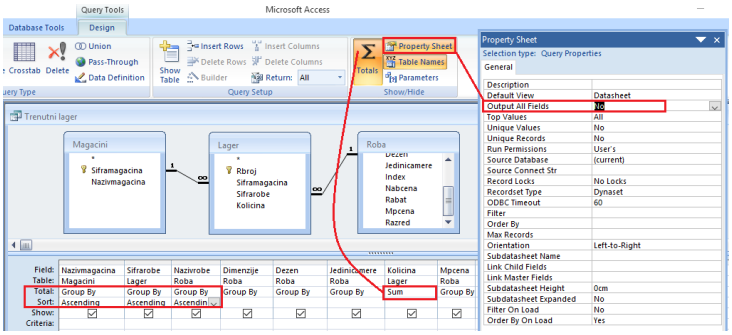
Nakon dodavanja potrebnih polja, neophodno je njihov raspored prilagoditi redosledu zahtevanom u zadatku, pri čemu polje Kolicina treba da se nalazi ispred polja Mpcena. Kolona se premešta tako što se najpre selektuje klikom na njen vrh (obeležen crvenom linijom na slici ispod), a zatim se levim tasterom miša prevuče na željenu poziciju.



Nakon pokretanja upita može se uočiti da se, pored šifre robe, prikazuju i ostala polja koja su dodata iz tabele Roba. Međutim, zadatak još uvek nije u potpunosti završen. U zahtevu zadatka navedeno je da se prikaže naziv magacina, a ne šifra magacina. Zbog toga je potrebno u upit dodati i tabelu Magacini, iz koje se ubacuje polje Nazivmagacina, dok se iz upita uklanja polje Siframagacina.

Nakon toga, podatke je potrebno sortirati u rastućem redosledu, u skladu sa prikazom na slici ispod, i ponovo pokrenuti upit.

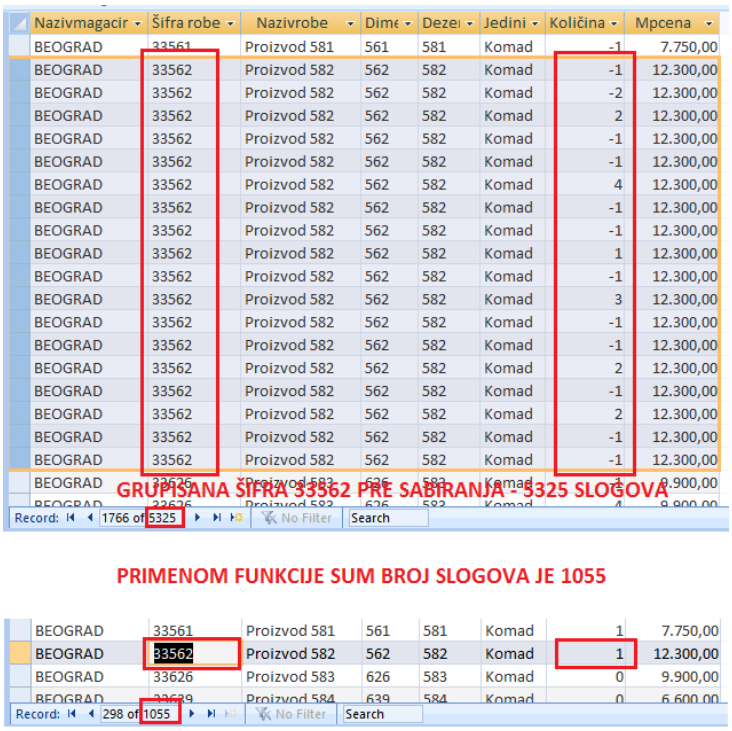
Ukoliko se prilikom izvršavanja upita prikažu i polja koja nisu eksplicitno dodata u upit, neophodno je uključiti *Property Sheet* i opciju *Output All Fields* postaviti na *No*. Nakon ove izmene, u rezultatu upita biće prikazana samo tražena polja. Ista opcija važi i prilikom korišćenja dugmeta *Totals*, kada se može pojaviti poruka *Cannot group on fields selected with **, čija će upotreba biti objašnjena u nastavku.



Uključivanjem dugmeta *Totals* dobija se mogućnost grupisanja podataka ili primene neke od ponuđenih funkcija, kao što su *Sum*, *Avg*, *Min*, *Max* i *Count*. U tabeli Lager, za svaku šifru robe evidentirane su količine, pri čemu su količine od nabavke pozitivne, dok su količine od prodaje negativne.

Grupisanjem slogova prema šifri robe i primenom funkcije *Sum* nad poljem Kolicina, dobijaju se sabrane količine, odnosno trenutni lager za pojedinačne šifre robe. Nakon pri-

mene grupisanja, broj slogova u rezultatu upita više nije 5325, već iznosi 1055.



3.3. Pregled rezultata upita

Rezultati izvršenih upita prikazuju se u tabelarnom obliku, što omogućava korisnicima da brzo analiziraju i interpretiraju podatke. Pregled rezultata može uključivati sortiranje, filtriranje i grupisanje podataka, što pomaže u identifikaciji važnih informacija i donošenju odluka.

S obzirom da je upravo **SELECT** upit osnovni način za čitanje podataka iz baze, sledeći primer prirodno ilustruje ovaj proces.

SELECT upit određuje koje kolone i iz kojih tabela želimo da dobijemo podatke, a izvršavanjem upita dobijamo tabelarni prikaz koji možemo analizirati. Na ovaj način, veza između pisanja upita i pregleda rezultata postaje jasna i praktično primenljiva.

Svaki, pa i najjednostavniji upit **SELECT** mora da sadrži sledeće elemente:

```
SELECT kolone
FROM tabele;
```

Ovim se sistemu nalaže da nam iz navedene tabele (ili više tabela) izdvoji sve podatke koji su dati u navedenim kolonama.

Jezik nije osetljiv na razliku između velikih i malih slova (moguće je pisati i `select` i `from`). Sve beline u upitu se zanemaruju, pa samim tim nije bitno da li se upit piše u jednom ili više redova.

Za početak ćemo uvek čitati podatke samo iz jedne tabele. Najjednostavniji slučaj je onaj u kome se želi čitanje svih podataka iz jedne tabele. Tada se umesto imena svih kolona može navesti samo simbol `*`. Tačka-zarez (`;`) može i ne mora da se doda na kraj upita.

Prikazati sve podatke o učenicima koji su upisani u bazi:

```
SELECT *
FROM ucenik;
```

Izvršavanjem upita dobija se sledeći rezultat:

id	ime	prezime	pol	datum_rodjenja	razred	odeljenje
1	Petar	Petrović	m	2006-07-01	1	1
2	Milica	Jovanović	ž	2006-04-03	1	1
3	Lidija	Petrović	ž	2006-12-14	1	1
4	Petar	Milovanović	m	2005-12-08	2	1
5	Ana	Pekić	ž	2005-02-23	2	1

Ovaj upit znači:

ODABERI sve kolone
IZ REDOVA tabele `ucenik`

On je funkcionalno ekvivalentan sledećem upitu, ali je od njega jednostavniji za pisanje:

```
SELECT id, ime, prezime, pol, datum_rodjenja, razred, odeljenje
FROM ucenik;
```

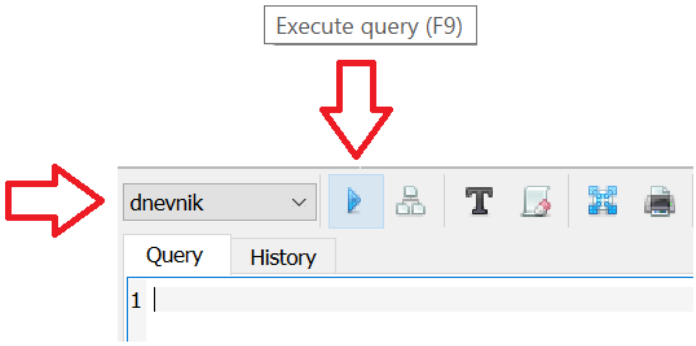
Izvršavanjem upita dobija se sledeći rezultat:

id	ime	prezime	pol	datum_rodjenja	razred	odeljenje
1	Petar	Petrović	m	2006-07-01	1	1
2	Milica	Jovanović	ž	2006-04-03	1	1
3	Lidija	Petrović	ž	2006-12-14	1	1
4	Petar	Milovanović	m	2005-12-08	2	1
5	Ana	Pekić	ž	2005-02-23	2	1

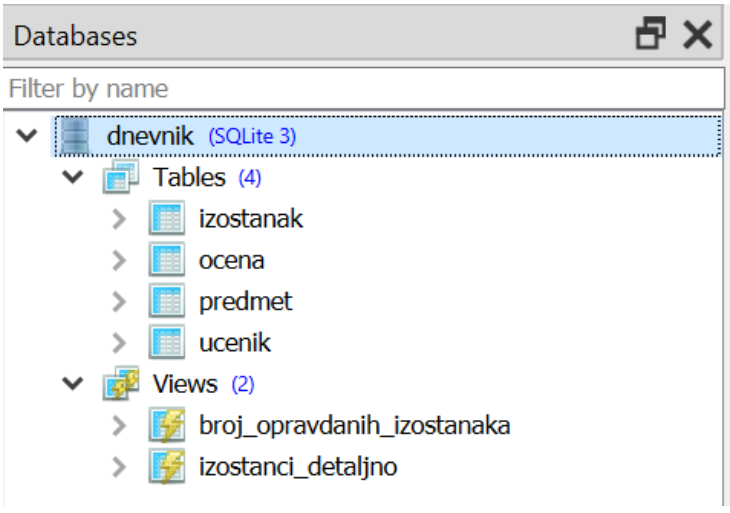
Oblik upita **SELECT** koji smo upravo upoznali sadrži samo obavezne delove, pa je to najkraći mogući oblik ovog upita.

Obradićemo različite složenije oblike upita **SELECT**, koji nam omogućavaju da od postojećih podataka odaberemo samo neke, da ih prebrojimo, da nađemo najmanji ili najveći podatak koji ispunjava neki uslov, da prikažemo jednostavne statistike po grupama podataka i slično.

U sistemu *SQLite Studio* se upiti pišu nakon što se klikne na kreiranu bazu **dnevnik** u prozoru *Databases* i potom izabere komanda menija *Tools* → *Open SQL Editor*. Kada se napiše upit, klikne se na dugme *Execute query (F9)* (plavi trouglić). Kako je prikayano na slici.

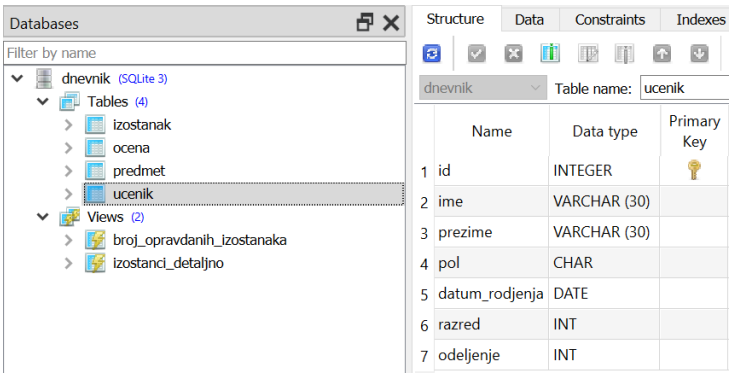


Ukoliko se u prostoru za pisanje upita nalazi više njih, potrebno je obeležiti onaj koji želimo da pokrenemo. Ukoliko imamo više baza podataka, obavezno proveri da li je pored ovog dugmeta naziv baze u kojoj želiš da vršiš upite.

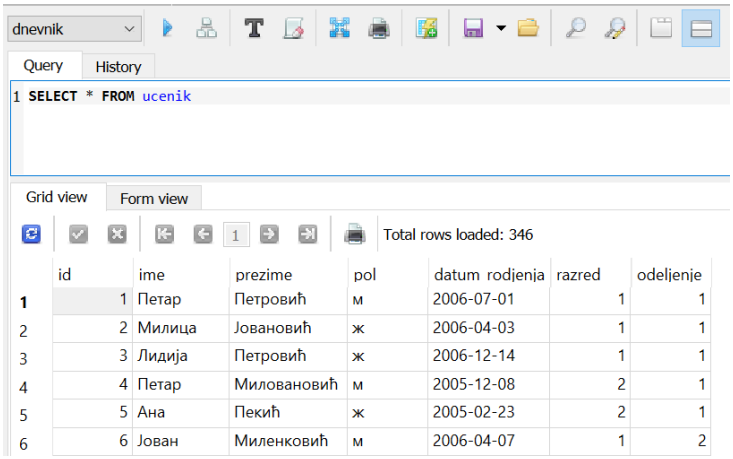


Često će nam kod upita biti potrebno da znamo i tačne nazive kolona. To možemo da vidimo za svaku tabelu pojedinačno tako što kliknemo na nju u prozoru *Databases*, pa se

onda pojavi opis strukture tabele koji sadrži spisak svih kolona.



Spisak kolona možemo da vidimo i kada izvršimo osnovni **SELECT** upit:



3.4. Kreiranje multitabularnih upita

Multitabularni upiti (*multi-table queries*) omogućavaju korisnicima da povežu podatke iz više tabela koristeći operacije kao što su **JOIN**, **INNER JOIN** i **LEFT JOIN**. Ovi upiti su posebno korisni kada je potrebno kombinovati informacije iz različitih delova baze podataka kako bi se dobio potpuniji uvid u podatke.

Multiplikativni upiti predstavljaju vrstu upita u *SQL*-u koji omogućavaju kombinovanje podataka iz više tabela. Ovakvi upiti su posebno korisni kada je potrebno analizirati sve moguće parove ili kombinacije podataka između tabela.

Prilikom kreiranja multiplikativnog upita, bitno je paziti na broj redova u rezultatima, jer ukoliko tabele sadrže veliki broj slogova, broj redova u rezultujućoj tabeli može eksponencijalno rasti. Na primer, upit koji kombinuje tabelu A sa 100 redova i tabelu B sa 50 redova rezultiraće sa 5000 redova u multiplikativnom rezultatu.

U praksi se multiplikativni upiti često kombinuju sa *JOIN* operacijama kako bi se rezultati filtrirali i ograničili na relevantne kombinacije, čime se izbegava nekontrolisano povećanje broja slogova. Ovim se omogućava da se iz multiplikativnog upita dobiju samo oni parovi ili kombinacije podataka koji su korisni za analizu.

Multiplikativni upiti su korisni u situacijama kada je potrebno izvršiti analize poput poređenja svih proizvoda sa svim kupcima, kombinovanja svih uslova sa svim stavkama narudžbine itd. Važno je planirati upit i koristiti dodatne kriterijume filtriranja kako bi rezultati bili pregledni i korisni.

4. Izveštaji

Izveštaji predstavljaju organizovan prikaz podataka iz baza podataka, namenjen analizi, prezentaciji i štampi. Oni se sastoje od informacija koje se preuzimaju iz tabela ili upita, kao i dodatnih elemenata poput oznaka, naslova, grafika i formata koji se definišu prilikom dizajna izveštaja.

Tabele ili upiti koji pružaju osnovne podatke nazivaju se *record source* izveštaja. Ako se sva polja nalaze u jednoj tabeli, ta tabela može direktno služiti kao izvor podataka. U slučaju da su potrebna polja iz više tabela, koriste se upiti, koji mogu već postojati ili biti kreirani posebno da zadovolje potrebe konkretnog izveštaja.

Izveštaji omogućavaju pregled podataka na pregledan i čitljiv način, često uključujući sortiranje, grupisanje, filtriranje i agregacije.

Pored analitičke funkcije, izveštaji služe i kao dokumentacija i alat za donošenje odluka, jer omogućavaju vizuelizaciju važnih podataka i njihovih odnosa.

4.1. Kreiranje izveštaja

Kreiranje izveštaja može se izvršiti na više načina. Jedan od najbržih je korišćenje alatke *Report* koja automatski generiše izveštaj na osnovu izabranog izvora podataka.

Ova metoda omogućava brzo pregledanje osnovnih podataka, dok finalni izgled izveštaja može biti dodatno uređivan u prikazu *Layout* ili *Design*, kako bi se postigla bolja preglednost i estetski efekat.

4.1.1 Kreiranje izveštaja pomoću čarobnjaka

Čarobnjak za izveštaje (*Report Wizard*) omogućava korisniku da precizno odredi koja polja će biti uključena u izveštaj, kako će podaci biti grupisani i sortirani, kao i da koristi polja iz više tabela ili upita, pod uslovom da su prethodno definisane relacije između tabela.

1. Na kartici *Create*, u grupi *Reports*, izaberite opciju *Report Wizard*.
2. Pratite uputstva čarobnjaka, birajući polja, način grupisanja i sortiranja podataka.
3. Na poslednjoj stranici kliknite *Finish* da biste generisali izveštaj.

Kada je izveštaj generisan, moguće je pregledati kako će izgledati prilikom štampe i koristiti opciju zumiranja (*Zoom*) za detaljan pregled. Izveštaj se može sačuvati i naknadno menjati, čime se omogućava stalno ažuriranje podataka iz izvora zapisa i prilagođavanje izgleda potrebama korisnika.

4.1.2 Saveti za dizajn i upotrebu izveštaja

Prilikom kreiranja izveštaja važno je:

- koristiti relevantan *record source* koji obuhvata sva potrebna polja,
- primeniti logičko grupisanje i sortiranje kako bi podaci bili pregledni,
- koristiti dodatne elemente dizajna, kao što su naslovi, oznake i grafika, za bolju vizuelnu interpretaciju,
- proveriti da su podaci ažurni, jer izveštaj prikazuje stanje u trenutku otvaranja izveštaja.

Kombinovanjem automatskog čarobnjaka i manuelnog dizajn moda omogućava se brzo kreiranje funkcionalnih i estetski dopadljivih izveštaja, koji se mogu koristiti za analize, prezentacije i dokumentaciju.

4.1.3 Izveštaji u *Access*-u

Izveštaji u *Access*-u su organizovani u odeljke koji definišu izgled i ponašanje podataka prilikom štampe ili prikaza.

Razumevanje funkcije svakog odeljka omogućava precizno kreiranje preglednih i funkcionalnih izveštaja. U prikazu *Design* moguće je pregledati strukturu izveštaja i njegovih odeljaka, što olakšava postavljanje kontrola, agregacija i drugih elemenata.

Glavni odeljci izveštaja uključuju:

- **Zaglavlje izveštaja (*Report Header*)**: Štampa se samo jednom, na početku izveštaja. Koristi se za informacije koje bi se obično pojavljivale na naslovnoj stranici, kao što su logotip, naslov ili datum. Kada se u ovom odeljku postavi izračunata kontrola koja koristi agregatnu funkciju *Sum*, zbir se računa za ceo izveštaj.
- **Zaglavlje stranice (*Page Header*)**: Štampa se na vrhu svake stranice. Koristi se, na primer, za ponavljanje naslova izveštaja na svakoj stranici.
- **Zaglavlje grupe (*Group Header*)**: Štampa se na početku svake grupe zapisa. Idealno je za odštampavanje imena grupe, npr. naziv proizvoda u izveštaju grupisanom po proizvodima. Agregatne funkcije u ovom odeljku primenjuju se samo na trenutnu grupu.
- **Detalj (*Detail*)**: Štampa se za svaki red u izvoru zapisa. Ovo je glavni odeljak u kojem se postavljaju kontrole koje prikazuju podatke.
- **Podnožje grupe (*Group Footer*)**: Štampa se na kraju svake grupe zapisa i služi za prikaz zbirnih informacija ili sažetih statistika za grupu.
- **Podnožje stranice (*Page Footer*)**: Štampa se na kraju svake stranice. Koristi se za brojeve stranica, datum ili druge informacije o stranici.
- **Podnožje izveštaja (*Report Footer*)**: Štampa se samo jednom, na kraju izveštaja. Idealan za zbirne vrednosti, ukupne iznose ili sažetke podataka za ceo izveštaj.

Razumevanje i pravilno korišćenje ovih odeljaka omogućava:

- pregledne i logički organizovane izveštaje,
- pravilno grupisanje i agregaciju podataka,
- postavljanje kontrola na mesta koja određuju tačno gde se i kada vrednosti izračunavaju,
- prilagođavanje izveštaja različitim potrebama korisnika, kao što su analize, prezentacije ili štampa.

Praktično iskustvo pokazuje da je pravilno korišćenje odeljaka ključno za efikasno kreiranje izveštaja koji su i funkcionalni i estetski dopadljivi. Ova organizacija omogućava korisnicima da odmah identifikuju važne informacije, da grupišu podatke po kriterijumima i da dobiju pregled ukupnih ili zbirnih vrednosti.

4.1.4 Pregled izveštaja

Postoji više načina za prikaz izveštaja. Izbor metoda zavisi od toga šta želite da uradite sa izveštajem i njegovim podacima:

- Ako je cilj napraviti privremene promene u tome koji podaci se pojavljuju u izveštaju pre štampanja, ili ako želite da kopirate podatke iz izveštaja u drugi dokument, koristi se *Report View*.
- Ako želite mogućnost da promenite dizajn izveštaja dok posmatrate podatke, koristi se *Layout View*.
- Ako je potrebno samo videti kako će izveštaj izgledati kada se odštampa, koristi se *Print Preview*.

Ovi prikazi omogućavaju fleksibilnost pri radu sa izveštajima, bilo da se vrše brze provere podataka, prilagođavanje dizajna ili priprema za štampanje.

4.2. Postavljanje kontrola i izračunavanja u izveštajima

Kontrole u *Access*-u predstavljaju objekte koji prikazuju podatke, omogućavaju rad sa informacijama i poboljšavaju korisnički *interfejs*, uključujući oznake, slike i druge grafičke elemente. Kontrole se mogu svrstati u tri osnovna tipa: povezane (*bound*), nepovezane (*unbound*) i izračunate (*calculated*).

4.2.1 Povezane kontrole

Povezana kontrola je kontrola čiji je *data source* polje u tabeli ili upitu. Ove kontrole se koriste za prikaz vrednosti iz baze podataka, uključujući tekst, brojeve, datume, vrednosti *Yes/No*, slike i grafike. Najčešći tip povezane kontrole je *text box*.

4.2.2 Nepovezane kontrole

Nepovezana kontrola nema izvor podataka i koristi se za prikaz statičkih informacija, linija, pravougaonika, naslova ili slika. Na primer, oznaka koja prikazuje naslov izveštaja je nepovezana kontrola. One omogućavaju vizuelno obogaćivanje izveštaja i organizovanje informacija nezavisno od baze podataka.

4.2.3 Izračunate kontrole

Izračunata kontrola koristi izraz umesto polja kao *data source*. Izraz može sadržati operatore (npr. =, +, *, /), imena kontrola, polja iz izvora zapisa ili konstantne vrednosti. Na primer, izraz:

```
= [Cena po jedinici] * 0,75
```

izračunava cenu stavke sa 25% popusta množenjem vrednosti iz polja *Cena po jedinici* sa konstantom 0,75. Izračunate kontrole omogućavaju fleksibilne kalkulacije i agregacije podataka unutar izveštaja, bez potrebe za modifikacijom osnovnih tabela ili upita.

4.2.4 Postavljanje kontrola u izveštaju

Prilikom kreiranja izveštaja, preporučljivo je prvo dodati i rasporediti sve povezane kontrole, jer one čine osnovu većine izveštaja. Nepovezane i izračunate kontrole se potom dodaju kako bi se dopunio dizajn i omogućilo izračunavanje dodatnih vrednosti.

Povezivanje kontrole sa poljem može se ostvariti prevlačenjem polja iz okna *Field List* u izveštaj. Okno *Field List* prikazuje polja iz izvora zapisa izveštaja (tabele ili upita). Alternativno, ime polja može se uneti direktno u kontrolu ili u svojstvo *ControlSource* u listi svojstava kontrole. Lista svojstava definiše karakteristike kontrole, uključujući ime, izvor podataka i format.

Korišćenje okna *Field List* ima dve prednosti:

- Povezana kontrola automatski dobija prateću oznaku koja preuzima ime polja ili definisani natpis, što olakšava označavanje i štedi vreme.
- Povezana kontrola nasleđuje većinu svojstava polja iz osnovne tabele ili upita (kao što su *Format*, decimalna mesta i ulazna maska), osiguravajući doslednost prikaza podataka.

Nepovezana kontrola se takođe može povezati sa poljem naknadno, tako što se svojstvo *ControlSource* postavi na ime polja. Ova fleksibilnost omogućava dizajniranje izveštaja koji kombinuju statičke i dinamičke elemente, prilagođene potrebama krajnjih korisnika.

4.3. Kreiranje multitabularnih izveštaja

Multitabularni izveštaji omogućavaju prikaz podataka iz više tabela ili upita u jednom izveštaju, čime se olakšava analiza i upoređivanje informacija iz različitih izvora. Ovakvi izveštaji se kreiraju tako što se kao *record sources* koriste tabele ili prethodno definisani *queries*, a između njih se uspostavljaju relacije kako bi se podaci ispravno povezali.

Kada se pravi multitabularni izveštaj, važno je definisati strukturu i raspored podataka. Zaglavlja i podnožja mogu se koristiti za grupisanje zapisa po ključnim kategorijama, dok se u detaljnom odeljku izveštaja prikazuju konkretni podaci iz više tabela.

Upotreba *subreports* omogućava uključivanje dodatnih detalja iz povezanih tabela unutar glavnog izveštaja, čime se dobija pregledan i informativan izveštaj.

Kontrole u izveštaju, uključujući povezane, nepovezane i izračunate kontrole, koriste se za prikaz vrednosti iz različitih izvora. Pravilnim povezivanjem kontrola sa poljima iz tabela i upita, moguće je automatski izračunati i prikazati agregirane vrednosti, kao što su *sum*, *average* ili *count*.

Dizajn multitabularnog izveštaja treba da bude jasan i pregledan: raspored kolona, boje, obrubi i druge vizuelne karakteristike pomažu korisniku da brzo identifikuje ključne podatke. Ovakvi izveštaji su posebno korisni za praćenje višedimenzionalnih informacija, kao što su lageri po magacinima, prodaja po proizvodima ili finansijski pokazatelji po odeljenjima.

5. Vizuelizacija podataka baze

Vizuelizacija podataka baze predstavlja prikaz i interpretaciju informacija iz baze na način koji olakšava razumevanje, analizu i donošenje odluka.

Podaci iz tabela ili upita često se vizuelizuju kroz interaktivne komponente korisničkog *interfejsa*, grafikone, tabele i obrasce koji omogućavaju pregled, filtriranje i manipulaciju podacima.

5.1. Komponente za povezivanje Windows aplikacije sa bazom podataka

Povezivanje Windows aplikacije sa bazom podataka obuhvata komponente i mehanizme koji omogućavaju uspostavljanje i održavanje veze između aplikacije i izvora podataka.

U .NET okruženju, vizuelne komponente se često oslanjaju na strukture kao što su *BindingSource*, *DataTable*, *DataSet* i *DataView*, koje omogućavaju automatsku sinhronizaciju podataka između baze i elemenata korisničkog *interfejsa*.
BindingSource predstavlja posrednika koji povezuje izvor podataka sa više kontrola, omogućavajući dvosmerno ažuriranje i obaveštavanje o promenama.

Pored toga, Windows aplikacije se mogu povezivati sa bazama podataka korišćenjem različitih interfejsa kao što su *ODBC* ili specifični drajveri za određene DBMS-ove, a *Visual Studio* i druge razvojne platforme nude alate koji olakšavaju konfigurisanje veze,

izvršavanje upita i upravljanje rezultatima direktno u kodu ili kroz vizuelne komponente koje podržavaju *data binding*.

5.2. Vizuelne komponente za prikazivanje i modifikaciju podataka baze

Vizuelne komponente predstavljaju kontrolne elemente korisničkog interfejsa koje omogućavaju prikaz i modifikaciju podataka iz baze. Najčešće se koriste tabele, liste, obrasci, padajuće liste i grafikoni koji su povezani sa izvorima podataka putem mehanizama *data binding*.

Na primer, komponenta *DataGridView* omogućava prikaz višestrukih zapisa baze u tabelarnom obliku, dok kontrole poput *TextBox* ili *ComboBox* služe za prikaz pojedinačnih vrednosti ili izbor iz skupa podataka.

Ove komponente omogućavaju ne samo prikaz podataka, već i njihovu modifikaciju, jer se promene u vizuelnim kontrolama mogu automatski propagirati nazad u izvor podataka. To znači da korisnik može unositi, ažurirati ili brisati podatke direktno, bez potrebe za direktnim pisanjem *SQL* upita. Vizuelni alati za razvoj aplikacija kao što je *Visual Studio* dodatno podržavaju automatsko generisanje i povezivanje ovih komponenti sa izvorima podataka.

5.3. Komponente za navigaciju

Komponente za navigaciju omogućavaju korisniku da se kreće kroz podatke i različite delove aplikacije. U Windows aplikacijama, tipične komponente za navigaciju uključuju *MenuStrip*, *ToolStrip*, *Toolbar* ili *Navigation Pane*, koje omogućavaju otvaranje različitih obrazaca, filtriranje podataka, prelazak između tabela ili modula aplikacije.

Na primer, u programu *Microsoft Access*, *Navigation Pane* prikazuje sve objekte baze podataka (tabele, obrasce, izveštaje) i omogućava brzi pristup njihovom otvaranju i upravljanju.

Dodatno, vizuelni elementi kao što su dugmad, kartice (*tabs*) i paneli omogućavaju intuitivno upravljanje sadržajem i prelazak između različitih funkcionalnosti aplikacije, čime se poboljšava korisničko iskustvo i olakšava rad sa podacima.

Literatura:

- https://petlja.org/sr-Latn-RS/kurs/4654/0
- https://www.bazepodataka.matf.bg.ac.rs/
- https://poincare.matf.bg.ac.rs/ smalkov/nastava.urbp.php
- https://poslovnainformatika.rs/access/referencijalni-integritet/
- https://poslovnainformatika.rs/access/kreiranje-upita/?cn-reloaded=1
- https://edukacija.rs/it/baze-podataka/sql-naredbe-setovi
- https://petlja.org/sr-Latn-RS/kurs/4654/