SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Renata Hanžek

Sustav za upravljanje bazama podataka

ZAVRŠNI RAD

Varaždin, 2018.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Renata Hanžek

Matični broj: 43-262/14 R

Studij: Primjena informacijske tehnologije u poslovanju

Sustav za upravljanje bazama podataka

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Kornelije Rabuzin

Varaždin, rujan 2018.

Renata Hanžek

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Prvi dio završnog rada obuhvatit će teoriju o sustavima za upravljanje baze podataka. Objasnit ću definiciju sustava za upravljanje baze podataka, koje su njegove najvažnije funkcije i prednosti. Također ću navesti osnovne zadatke te izvršavanje jednostavnih i složenih upita. Nadalje, navest ću i opisati tri poznata sustava za upravljanje bazama podataka. U praktičnom djelu prikazat ću izrađenu bazu podataka u sustavu Microsoft Access te ću opisati svaki postupak izrade i korištenja te baze.

Sadržaj:

1. Uvod

Od samog početka korištenja računala, obrada različitih vrsta podataka bila je jedan od osnovnih zadataka. Kada želimo imati kvalitetne informacije o svim segmentima našeg poslovnog ili čak privatnog života trebali bi na određen način organizirati sve podatke koji će nam pružati informacije koje su od velike važnosti u trenutku kada su nam potrebne. Ti podaci za svaki pojedini element bi se trebali organizirati tako da se mogu smjestiti u posebne tablice sa istovrsnim zaglavljem, a te tablice se mogu povezati preko određenih zajedničkih elemenata.

Svaka se tablica sastoji od redaka i stupaca. Presjek svakog reda i stupca je polje. Stupce još drugim riječima nazivamo i atributima, a svi atributi zajedno čine relacijsku shemu. S obzirom da u jednoj tablici može biti više redova, svaki red moramo jednoznačno identificirati kako bi ih mogli razlikovati. Zbog toga jedan ili više stupaca određujemo kao primarni ključ tablice. Primarni ključ mora biti jedinstven i mora uvijek imati neku vrijednost, dakle, ne smije biti null (prazno polje). Postoji i vanjski ključ koji je zapravo primarni ključ neke druge tablice.

Skup više tih tablica koje služe jednom zajedničkom cilju, skupa sa njihovim veznim elementima nazivamo bazom podataka. Postoji više načina na koji bi mogli definirati pojam baza podataka. Jedan od njih je spomenut u knjizi Marija Radovana (1993:1) gdje navodi slijedeću definiciju :

> „*Baza podataka je skup međusobno povezanih podataka koji tvori zajedničku podatkovnu osnovu svih aplikacija nekog informacijskog sustava. Baza podataka je, s jedne strane temelj integralnosti (povezanosti, cjelovitosti) informacijskog sustava, a s druge strane omogućava neovisnost pojedinačnih aplikacija sustava od unutarnje organizacije podataka u bazi podataka*.“

Postoji više modela podataka na kojima se svaka baza temelji. Najpoznatije su relacijske baze podataka koje se danas i najčešće koriste. Postoje još i hijerarhijske, mrežne, temporalne i objektne baze podataka.

Međutim, kako bismo uopće i mogli raditi s bazom podataka, potreban nam je sustav za upravljanje bazom podataka (DBMS). Najpoznatiji su Oracle, DB2, MS SQL Server te Microsoft Access.

U nastavku ću prvo govoriti općenito o sustavu za upravljanje bazama podataka, što znači taj pojam te kako se uopće koristi. Nakon toga ću prikazati svoj praktičan dio gdje sam izradila bazu podataka u Microsoft Access-u te ću opisati postupak izrade i korištenje te baze.

Za svoju bazu podataka prikupljala sam podatke o poznatim nogometnim klubovima. S obzirom da živimo u doba kada je nogomet nezamisliv pojam, ne samo za muškarce nego i za pojedine žene, odlučila sam stvoriti bazu podataka koje bi olakšala i pojednostavila pristup podacima spremljenima u tim bazama.

2. Sustav za upravljanje bazama podataka

Sustav za upravljanje bazama podataka (engl. DBMS - DataBase Management System) je programska podrška koja omogućava rad s bazama podataka odnosno definiranje baza podataka (spremište podataka), upis podataka u bazu, ispis podataka iz baze i obradu podataka smještenih u bazi podataka. Da bi mogao izvršavati ove zadatke mora biti zasnovan na nekom modelu podataka. Također, on mora podržavati programske jezike za definiranje strukture i uvjeta integriteta baze podataka te selekciju i izmjene sadržaja baze podatak, te mora posjedovati mehanizme za upravljanje transakcijama, zaštitu od neovlaštenog korištenja, zaštitu od uništenja, upravljanje distribuiranim dijelovima baze podataka, itd.

DBMS u skladu s traženom logičkom strukturom oblikuje fizički prikaz baze. On obavlja u ime klijenta sve operacije s podacima. Također je u stanju podržati razne baze, od kojih svaka može imati svoju logičku strukturu, ali u skladu s istim modelom. Dalje, on brine i za sigurnost podataka, te automatizira administrativne poslove s bazom.

Najvažnije funkcije sustava za upravljanje bazama podataka su funkcija za definiranje baze podataka, funkcija za manipuliranje podacima u bazi podataka te upravljačke funkcije koje su podijeljene u tri kategorije:

1. Funkcija za definiranje baze podataka - ostvaruje se standardnim jezikom za rad s bazom podataka kao što je SQL kod relacijskih baza podataka ili zasebnim jezikom kod mrežnih i hijerarhijskih baza podataka. U prvom ili drugom slučaju jezikom se opisuje shema baze podataka .

2. Funkcija za manipuliranje podacima u bazi podatka - ostvaruje se zasebnim jezikom kao što je SQL u relacijskoj bazi podataka. Također se može ostvariti dodavanjem naredbi za manipuliranje u tradicionalnom programskom jeziku kao što je npr. SQL.

3. Upravljačke funkcije - podijeljene su u tri kategorije. Prva je funkcija sigurnosti baze podatka zaštite od neovlaštenog korištenja. Pomoću te funkcije se korisnicima propisuje koje operacije nad bazom podataka mogu obavljati. Druga funkcija je funkcija očuvanja integriteta u bazi podataka kao što je npr. zaštita od mogućih

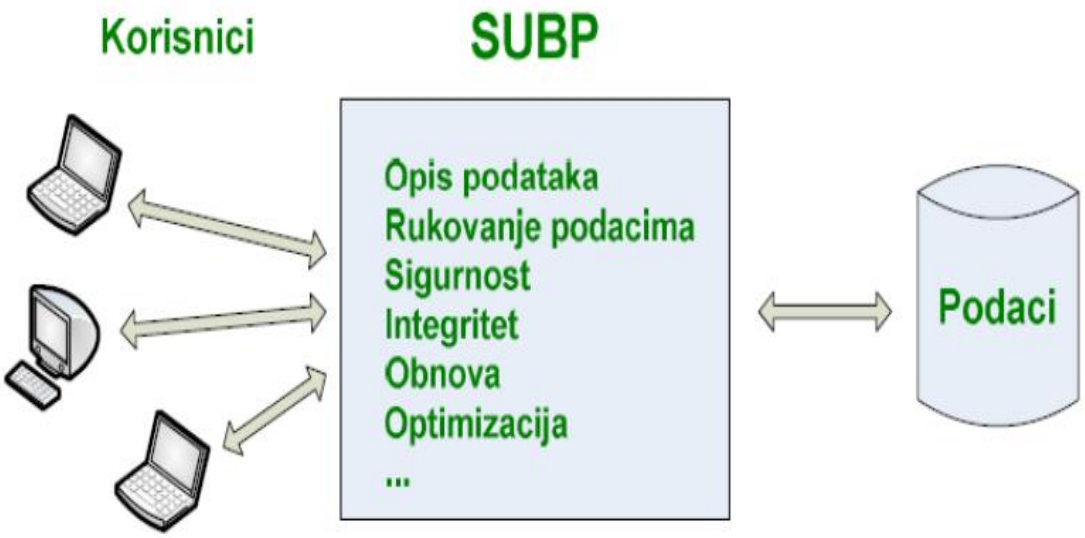
oštećenja (baza podataka se štiti uzimanjem sigurnosnih kopija podataka a nakon oštećenja obavlja se postupak oporavka). Posljednja funkcija je funkcija statističkog praćenja rada baze podataka.

Neke važnije prednosti DBMS-a su skladištenje podataka s minimumom redundance te pouzdanost podataka pri mogućim hardverskim i softverskim ispadima. Također je prednost DBMS-a što je pouzdano konkurentno korištenje od strane više korisnika. Nadalje, logička i fizička nezavisnost programa od podataka kao i jednostavno komuniciranje sa bazom podataka pomoću jezika bliskih korisniku, odnosno “upitnih jezika”.

Sustav za upravljanje bazama podataka oblikuje fizički prikaz baze u skladu s traženom logičnom strukturom. Također, on obavlja u ime klijenata sve operacije s podacima te se brine za sigurnost podataka.

Svaki DBMS ima osnovne zadatke koji se postavljaju nad njim (slika1), a to su:

1. opis i rukovanje podacima pomoću posebnog jezika (SQL)
   - definicija podataka, izmjena, dodavanje i brisanje podataka,
2. zaštita integriteta podataka
   - ne smije se dogoditi unos i/ili izmjena podataka koja narušava konzistentnost baze podataka,
3. visoki nivo sučelja prema korisniku
   - aplikacije moraju biti neovisne o lokaciji i fizičkoj strukturi podataka,
4. osiguravanje obnove podataka u slučaju djelomičkog ili potpunog razrušenja baze podataka
   - ponovna uspostava nasvježijeg mogućeg konzistentnog stanja, povrat podataka,
5. onemogućavanje štetnog međujelovanja u višekorisničkom radu
   - ne smije se dogoditi nekonzistentnost zbog istovremenih operacija nad istim podacima,
6. osigurati sigurnost baze podataka
   - svaki korisnik ima svoje ovlasti,
7. osigurati učinkovito izvođenje operacija nad podacima
   - optimizacija upita.



Slika 1: Zadaci sustava za upravljanje baza podataka

Svaki DBMS ima barem jedan alat za kreiranje i interaktivno izvršavanje SQL naredbi i pregled rezultatima kako bi uopće mogao izvršavati SQL naredbe interaktivo.

Postavljanje upita u DBMS-U može biti jednostano ili složeno. Svi upiti u SQL-u pa tako i oni najjednostavniji, postavljaju se naredbom SELECT. Rezultat izvođenja naredbe SELECT shvaća se kao nova, bezimena i privremena relacija koja ispisuje na zaslonu ili prosljeđuje u aplikacijski program. Kao što jednostavni upiti ispisuju podatke iz jedne tablice, tako složeni upiti vraćaju podatke iz više tablica kao rezultat upita. Naravno da su jednostavni upiti važni, no često nam trebaju podaci iz dvije, tri ili više tablica. Dozvoljeno je odjednom raditi više relacija, te kombinirati podatke iz njih. Jedna SELECT nareba može se ugnjezditi unutar druge, tako da rezultat prve naredbe služi kao dio uvjeta drugoj narebi.

U DBMS-u se izvršavaju transakcije. Transakcija je aktivnost ili niz aktivnosti koje izvršava jedan korisnik ili aplikacijski program, a koja čita ili ažurira sadržaj baze podataka. To je logička radna jedinica baze podataka i ona se mora provesti kao nedjeljiva cjelina jer inaće neće biti izvedena uopće.

Svaka transakcija bez obzira na vrstu se obavezno sastoji od sljedeća dva elementa:

1. početak transakcije,
2. završetak transakcije koji može označiti potvrdu transakcije (sve naredbe koje transakcija sadrži uspješno izvršene) i odustajanje od transakcije (došlo je do najmanje jedne greške u izvršavanju transakcije, pa bazu podataka moramo vratiti u onakvo stanje kakvo je bilo prije započinjanja transakcije.

Postoji nekoliko vrsta transakcija a to su automatske, eksplicitne, Batch-scoped transakcije, implicitne i distribuirane transakcije. Automatska transakcija je zapravo transakcija što se vidi u dnevniku transakcija i njihov naziv je automatske transakcije jer se same potvrđuju. Svaka eksplicitna transakcija počinje nakon naredbe BEGIN TRANSACTION i traje do eksplicitne definicije završetka dok implicitne transakcije ne započinju naredbom BEGIN već one započinju prvom naredbom u konekciji u kojoj su uključene implicitne transakcije. Batch-scoped transakcije su primjenjive kod višestruko aktivnih rezultirajučih skupova. Posljednja transakcija je distribuirana transakcija koja se koristi na distribuiranim sustavima. Distribuirani sustav ima više dijelova baze podataka na različitim serverima koji su razmješteni na lokacijski udaljenim mjestima i povezana su mrežam.

Zaštita i sigurnost baza podataka je vrlo često mnogo kompleksniji problem nego što je zaštita i sigurnosti drugih informacijskih sustava. Zbog toga sigurnosni mehanizmi ugrađeni u sustave za upravljanje bazama podataka su kompleksniji. Korištenje pogleda za uspostavu sigurnosti je jedan od sigurnosnih mehanizma. S aspekta sigurnosti, aktivne baze podataka moraju posjedovati mehanizme zaštite samih pravila od neovlaštenog čitanja i promjene.

Iako sustavi za upravljanje bazama podataka ne podržavaju sigurnostne mogućosti kod drugih sustava, ispravno postavnjanje postojećih mogučnosti može mnogo podiči sigurnosnu razinu zaštičenosti podataka te ukloniti veliki broj ranjivosti. Ugrađivanje sigurnosnih elemenata izravno u DBMS-ove i njihova ispravna primjena jedini su pravi način za uklanjanje ranjivosti, a ti elementi su :

1. dodjeljivanje primjerenih ovlasti i dozvola pristupa
   - ovo načelo temelji se na dozvoli pristupa samo onim podacima baze i funkcionalnostima DBMS-a koji su korisnicima neophodno potrebni, obzirom na njihov status i opis posla

- u svrhu podizanja računalne sigurnosti, ne preporučava se izravno dodjeljivanje ovlasti pojedinim koriničkim računima,

2. primjeru efektivnih korisničkih računa i zaporki
   - korisničke račune, nužne za pristup bazi podataka, potrebno je definirati u skladu s tradicionalnim metodama upravljanja korisničkim računima a to podrazumijeva promjenu izvorno postavljenih zaporki, onemogućenje korisničkog računa nakon određenog broja neuspjelih prijava, ograničenje pristupa podacima, onemogućenje neaktivnih korisničkih računa te upravljanje životnim ciklusom korisničkih računa,

3. primjerene metode nadzora i logiranja
   - jedan od ključnih elemenata zaštite DBMS-ova je nadzor koji treba biti usklađen s njihovom primjenom
   - pažjivo postavljen sustav nadzora omogučava uštede vremena i ne utječe značajno na performanse nadziranog DBMS-a,

4. korištenje enkripcije
   - obzirom na podatke koji se kriptiraju i na razinu na kojoj se kriptiranje obavlja, postoje različite vrste enkripcija primjenjivih u zaštiti baza podataka
   - jedna od mogućnosti je korištenje enkripcije za zaštitu podataka tijekom prijenosa
   - drugi je način primjena enkripcije na podatke u mirovanju, ali ni tada nije potrebno kriptirati sve podatke, već samo najosjetljivije,

5. nadzor nad pritupom tablicama
   - kontrola pristupa tablicama je vjerojatno najzanemarivaniji element zaštite baza podataka zbog toga što je njena implementacija složena i zahtjeva suradnju sistemskog administratora i razvojnog programera baze podataka
   - primjeri ovakve kontrole su onemogućavanje čitanja tablice u istoj sjednici u kojoj je u nju obavljen upis ili dozvoljavanje čitanja samo određenog tipa tablica.

2.1. SQL Server

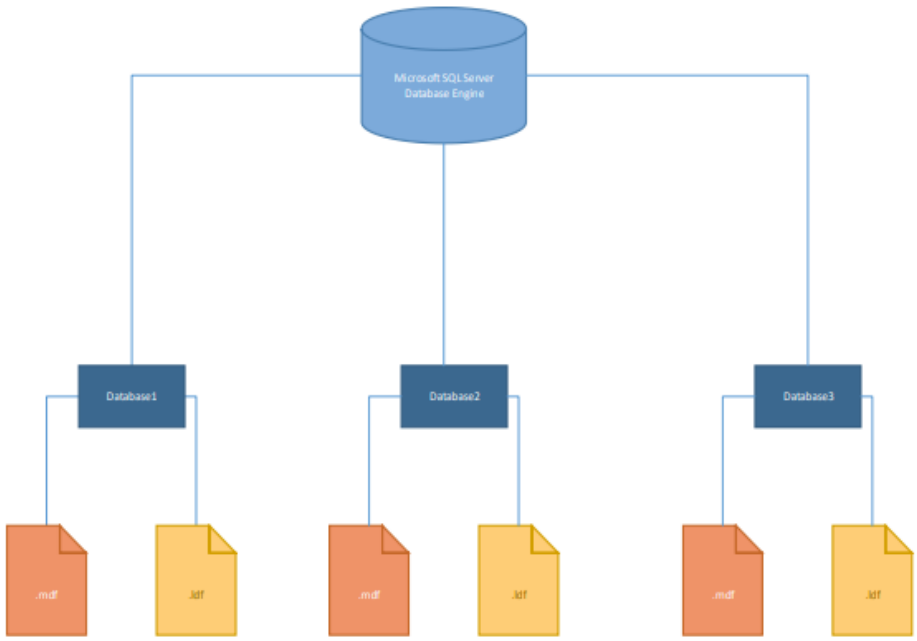
SQL Server je jedan od sustava za upravljanje bazama podataka. On u sebi predstavlja proizvod koji ujedinjuje snagu i fleksibilnost velikih baza podataka, uz istovremenu lakoću administracije kako smo već navikli u Windowsima. Može se koristiti u različite svrhe kao što su poslovna inteligencija ili za skladištenje podataka.

On nam nudi mogućnost prijenosa baze podataka sa jednog na više fizičkih servisa, koji se s aspekta korisnika ponašaju kao jedan. Također, dozvoljava korisniku da upite postavlja koristeći običan engleski jezik.

SQL server donosi mogućnost dostupa vrijednim poslovnim informacijama svim zaposlenicima, posebno onima o kupcima. Takve informacije sada mogu biti dostupne svim zaposlenicima i to na raznim tipovima uređaja, primjerice mobilnim uređajima putem Mobile Report Publishera.

Kada aplikacija postavlja upit, modificira i dodaje podatke u SQL Server bazu podataka, ona koristi Structured Query Language (SQL). SQL je standardni jezik koji se koristi za komunikaciju bazama podataka na SQL Serveru.

Svaka SQL Server baza podataka je pohranjena unutar minimalno dvije datoteke operacijskog sustava. Prva je primarna podatkovna datoteka koja sadrži podatke o objektima unutar baze podataka poput tablica, indeksa ili procedura. Druga datoteka je dnevnik transakcija koji sadrži podatke potrebne za obnovu transakcija i dovođenje baze u konzistentno stanje. Primarna podatkovna datoteka se može organizirati pomoću dodatnih, korisnički definiranih sekundarnih datoteka. Na taj način se može izbjeći problem povećanja baze podataka gdje bi primarna datoteka prelazila najveću moguću dopuštenu veličinu definiranu datotečnim sustavom.



Slika 2: Arhitektura SQL Server baze podataka

Također, SQL Server baze podataka vodi i dnevnik pogrešaka (SQL Server Error Log) u kojem se pohranjuju sistemski i korisnički definirani događaji. Prilikom svakog pokretanja SQL Servera stvara se nova error log datoteka u koju se pohranjuje dnevnik pogrešaka.

Neke od karakteritik SQL Servera su da štiti podatke i u stanju mirovanja i pokreta te gradi kritične aplikacije za online obradu transakcija koja omogućuje poboljšane performanse i dostupnost. Također, transformira podatke u vrijedne i dostupne uvide te dostavlja izvještaje na bilo koji uređaj u mreži ili van nje. Napravljena je stabilna platforma s alatima koji omogućavaju rad sa većim obujmom posla.

Prednosti korištenja SQL Servera su:

1. široko korišten alat za upravljanje bazom,
2. skup korisničkih alata za upravljanje podacima,
3. velike mogućnosti manipulacije podacima i prijenos u skladište podataka,
4. može rukovati velikom količinom podataka,
5. mogućnost rada na Internetu;

Nedostaci korištenja SQL Servera su:

1. rukovanje samom bazom još uvijek nije jednostavno,
2. SQL Server je namijenjen za veće tvrtke, iako postoje inačice za manje tvrtke i za edukaciju: MSDE (Microsoft SQL Server Desktop Edition) ili SQL Server Express;

Primarni jezik koji Microsoft SQL Server koristi je Transact SQL, što znači da se mogu koristiti osnovni upiti (naredbe SELECT i sl.), kao i petlje za mijenjanje programskog toka (IF/ELSE).

Pohranjivanje podataka izvršava se preko obične baze podataka, koja se sastoji od jednostavnih tablica sa stupcima i redovima. SQL server podržava razne tipove podataka, kao što su int, float, decimal, char, varchar i sl. Osim navedenih, SQL Server omogućava korištenje korisnički definiranih tipova podataka.

SQL Server dolazi u više različitih verzija, tako da će svaki korisnik bez obzira na veličinu baze koje ima, naći odgovarajući server za svoje potrebe. On se smatra najbržom i najpouzdanijom bazom podataka, no još ima puno mogućnosti za nadogradnju tog sustava.

2.2. Oracle

Oracle je također jedan od sustava za relacijsko upravljanje bazama podataka. On je zajedno sa SQL Serverom sustav koji je namijenjen za velike tvrtke gdje se podaci ne mjere stotinama i tisućama slogova već milijunima. Ona pored baze podataka još uključuje i cijeli skup pomoćnih alata i aplikacija kao što su e-mail i web serveri. Moram također napomenuti da Oracle baze podataka nisu namjenjene masovnom tržištu.

Oracle predlaže klaster. Klaster je ujedinjenje više manjih računala u jednu logičku cjelinu koja gledana iz vana djeluje kao jedno veliko računalo. Ovo praktički znači da koristeći takvu tehnologiju , podužeća više ne moraju ulagati u skupu opremu za koju ne znaju da li će ikada biti upotrijebljena u potpunom kapacitetu. Oracle također poboljšava pouzdanost ovakvog sistema jer je do sada sistem zavisio od jednog računala, dok sada u slučaju pada jednog ili više računala unutar klastera ostala računala unutar tog klastera dalje rade.

Oracle omogučuje enkripciju korisničkih zaporki tijekom mrežne komunikacije. Ako se ova mogučnost uključi na klijentskom i poslužiteljskom računalu, Oracle koristi prilagođeni DES (eng. Data Encription Standard) algoritam za enkripciju zaporki prije slanja. Također, ona omogučuje kontrolu složenosti korisničkih zaporki, njihovog roka trajanja i ponovnog korištenja.

Neke karakteristike Oracle-a su:

1. sigurnost,
2. lakše korištenje,
3. smanjenje redudanse,
4. konkurentno korištenje,
5. brz razvoj aplikacija,,
6. smanjenje broja grešaka;

Pod sigurnost podrazumijevamo zaštitu od neovlaštenog pristupa podacima (svaki korisnik mora dobiti ovlaštenje za pristup bazi kao i ovlaštenje za različite nivoe pristupa podacima u okviru te baze). Nadalje, možemo precizno definirati ovlaštenje za unos, ispravak i brisanje podataka na najnižem nivou knjiženja i mogućnost definiranja ovlaštenja isključivo za uvid podataka na višem nivou izvedenih podataka.

Svaki direktor ili šef može odredi nivo dostupnosti podataka pojedinom radniku ili da mu oduzme ovlaštenja. Također, postoji mogućnost načina zaštite od uništenja podataka te automatski oporavak podataka (nepotpunim završetkom programa, nestankom struje ili odustajanjem započete transakcije podaci se automatski, bez ikakve intervencije programera ili administratora, vraćaju na nivo početka transakcije).

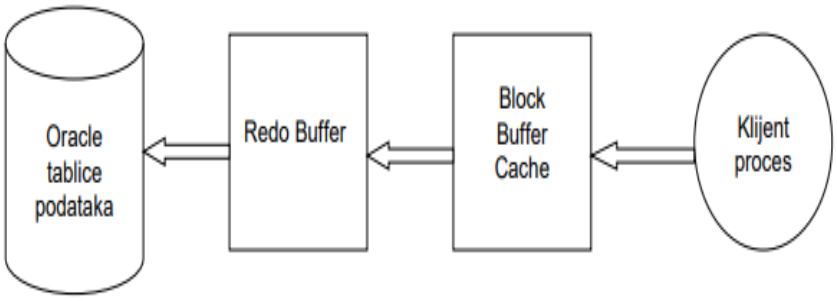
Za lakše korištenje:

1. ima manje programiranja,
2. postoje gotovi alati,
3. direktan je pristup korisnika do podataka,

4. različiti pregledi su urađeni nad istim podacima,
5. grafičko se prikazuje podatke,
6. moguće je povezivanje s drugim alatima.

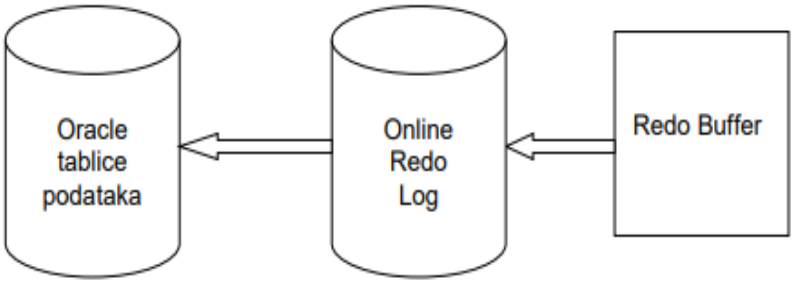
Oracle sustav čine dva glavna dijela. Prvi dio je baza podataka koju čine različite vrste datoteka. Najvažnije su datoteke podataka, no postoji i desetak drugih vrsta datoteka, od kojih su posebno važne Redo Log datoteke, koje se dijele na Online i Archive. Online Redo Log datoteke služe za oporavak baze u slučaju pada sustava (npr. programske greške ili nestanaka struje), dok Archive Redo Log datoteke služe za uspostavu prijašnjeg stanja, zajedno sa backup datotekama (na trakama ili diskovima) u slučaju kvara medija. Drugi dio je instanca (jedna ili više) baze podataka, koju čine memorijske strukture i procesi u memoriji.

Klijentski procesi (klijentski proces može biti proces na drugom računalu, proces na istom računalu na kojem se nalazi i Oracle SUBP, ali može biti i neki proces unutar Oracle SUBP-a) nikad ne dobivaju podatke direktno sa diska. Svi podaci koji se čitaju sa diska smještaju se u Block Buffer Cache. Također, kada klijentski proces mijenja podatke, ne mijenja ih direktno na disku, već se promjene prvo spremaju u Block Buffer Cache. Nakon spremanja promjena u Block Buffer Cache može se izvršiti upis koji se radi preko Redo Buffer-a kao što prikazuje slika 3.



Slika 3: Izmjena podataka kroz Redo Buffer

Osim toga, mijenjanje podataka ne ide tako da se podaci odmah upisuju u tablice podataka, već se iz Redo Buffer-a prvo upisuju u Online Redo Log datoteku.



Slika 4: Zapisivanje u Online Redo Log

Online Redo Log čine minimalno dvije datoteke (a preporučljivo je da postoje barem tri) koje se koriste u krug. Kada se prva napuni, Oracle počinje pisati u drugu, a kada se druga napuni Oracle se vraća na prvu. Kada su podaci zapisani u Online Redo Log može doći do pada sustava (to nije kvar medija) prije nego Oracle te podatke upiše u prave tablice podataka. No nakon što se Oracle sustav ponovno podigne, pročitat će Online Redo Log i upisati podatke u prave tablice.

Online Redo Log, kako mu i samo ime kaže, sadrži samo sliku za REDO. Podaci potrebni za UNDO nalaze se u jednom posebnom prostoru na disku, koji se naziva UNDO tablespace. Oracle zapisuje staro stanje redaka u UNDO tablespace uvijek kada radi izmjenu podataka (INSERT / UPDATE / DELETE).

UNDO tablespace, osim što služi za eliminiranje (iz tablica podataka) promjena koje nisu COMMIT-irane, kod oporavka sustava nakon pada, služi i za omogućavanje višekorisničkog rada. Postojanje UNDO tablespace-a omogućava da mijenjanje podataka ne utječe na čitanje podataka, tj. mijenjanje podataka ne spriječava da se ti podaci istovremeno i čitaju.

3. Primjer baze podataka

Baza podataka je organizirana zbirka podataka. Namijenjena je za pohranjivanje, analizu i pretraživanje grupe srodnih i povezanih podataka. Podaci su dostupni istovremeno raznim korisnicima i aplikacijskim programima. Upisivanje, promjena, brisanje i čitanje podataka obavlja se pomoću posebnog softvera, a to je sustav za upravljanje bazom podataka. Korisnici i aplikacije ne moraju pri tome poznavati detalje fizičkog prikaza podataka, nego se referenciraju na neku idealiziranu logičku strukturu baze.

“Može se slobodno reći da baza podataka nije ništa drugo nego skup povezanih, organiziranih podataka te da korisnici (u pravilu) bazu podataka doživljavaju kao skup (povezanih) tablica. “ (Uvod u SQL, Kornelije Rabuzin,2011.)

Tijekom dizajniranja baze podataka najprije trebamo dobro razmisliti koje bi podatke trebalo spremiti u bazu. Primjer koji navodi Kornelije Rabuzin (2011:13) je da ako imamo tablicu kupaca, onda je preporuka da u tablici imamo samo podatke o kupcima, a ne i podatke o njihovim narudžbama i proizvodima koje su naručili.

Pošto u ovo doba svi žele pronaći način kako pronaći sve podatke na jednom mjestu, smatram da je izrada baze podataka na računalu idealno rješenje. Najveća prednost je ta što je uvelike smanjena potrošnja papira i tonera te je lakše i brže snalaženje u bazi podataka.

S obzirom da sam se odlučila napraviti bazu podataka u Microsoft Access sustavu, u ostalom djelu rada ću pokazati postupak izrade vlastite baze. Prikupila sam podatke o nogometnim klubovima. Na temelju toga sam napravila i posebnu tablicu u kojoj se vidi godina osnivanja pojedinih nogometnih klubova, mjesto osnivanja, država te igralište kluba. Nadalje, prikazala sam i ukupan broj golova u prethodnoj godini i postotak posjeda lopte. Napravila sam još i tablice posebno za trenere pojedinih klubova i kapetana. Na temelju tih podataka možemo u samo nekoliko koraka kreirati upit sa podacima koji su nama bitni, napraviti izvješće o određenim podacima iz tablica ili formu.

Moja je aplikacija namijenjena prvenstveno lakšem pristupu podataka. Važno je napomenuti da se u nju mogu i dalje unositi podaci. Npr, ako dođe do promjene trenera ili kapetana nekog nogometnog kluba možemo lako promijeniti podatke.

Nadalje, u aplikaciji sam postavila i određena ograničenja unosa te je tako rizik od pogrešnog upisa podataka je sveden na minimum. Osoba koja je zadužena za unos podataka može efikasnije i brže obavljati svoj posao. Ako se dogodi da upiše pojedini podatak u krivi stupac, aplikacija ga upozorava na počinjenu grešku.

3.1. Model baze podataka (tablice i veze)

Model baze podataka predstavlja skup definicija koje određuju kako bi trebalo oblikovati i zapisivati podatke te definira operacije koje se mogu izvoditi s podacima. Postoji nekoliko modela podataka koji su se koristili kroz povijest, a poznati su mrežni (temelji na usmjerenim grafovima) i hijerarhijski (temelji se na odnosima „podređeni-nadređeni“). Tu su još i deduktivni, temporalni i objektni. Međutim, najpoznatiji je relacijski model podataka koji se danas najviše koristi. U nastavku ću opširnije objasniti modele.

Relacijski model je zasnovan na matematičkom pojmu relacije. I podaci i veze među podacima prikazuju se tablicama koje se sastoje od redaka i stupaca.

Kod mrežnog modela je baza predočena mrežom koja se sastoji od čvorova i usmjerenih lukova. Čvorovi predstavljaju tipove zapisa (slogova podataka), a lukovi definiraju veze među tipovima zapisa.

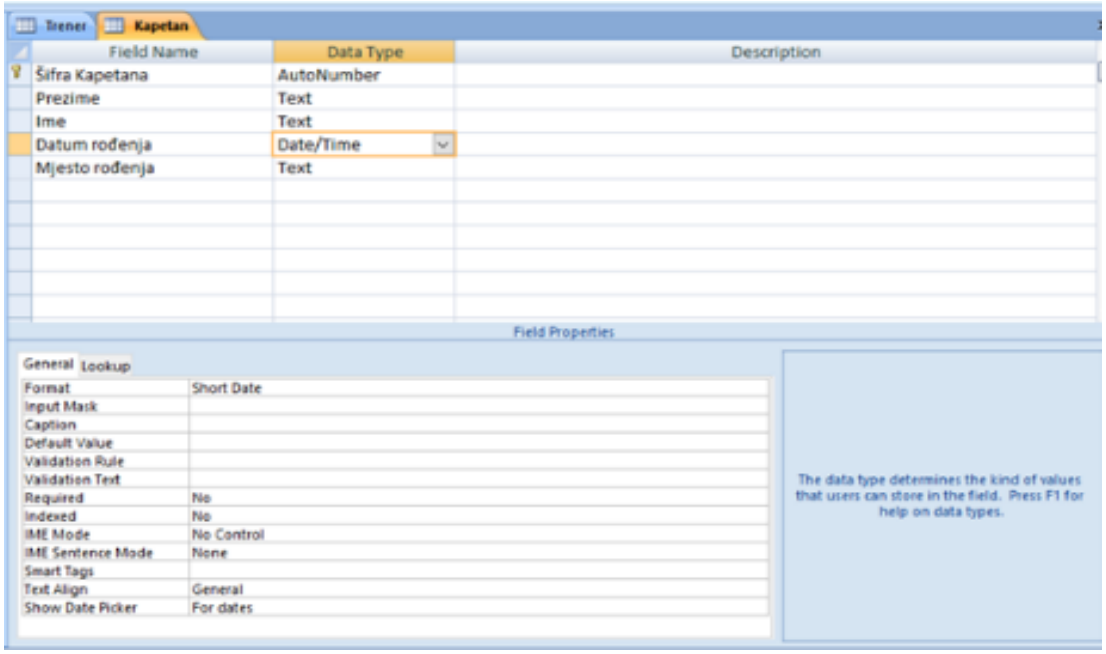
Hijerarhijski model je poseban slučaj mrežnog modela. Baza mu je predočena jednim stablom (hijerarhijom) ili skupom stabala. Svako stablo sastoji se od čvorova i veza “nadređeni-podređeni” između čvorova. Čvorovi su tipov zapisa, a odnos “nadređeni-podređeni” izražava hijerarhijske veze među tipovima zapisa.

Objektivni model je inspiraran objektivno orijentiranim programskim jezicima. Baza je predočena kao skup trajno pohranjenih objekata koji se sastoje od svojih internih atributa (podataka) i metoda (operacija) za rukovanje tim podacima. Svaki objekt pripada nekoj klasi. Između klasa se uspostavljaju veze nasljeđivanja itd.

Hierarhijski i mrežni model bili su u uporabi u 60-im i 70-im godinama 20. stoljeća. Od 80-ih godina prevladava relacijski model. Očekivani prijelaz na objektivni model za sada se nije dogodio, tako da današnje baze podataka uglavnom i dalje možemo poistovjetiti s reacijskim bazama, iako se danas sve više koriste NoSQL baze podataka. Svi prije spomenuti poznati sustavi za upravljanje bazama podataka, koji su danas u širokoj uporabi, podržavaju isključivo relacijski model.

3.1.1. Tablice

Tablice su temelj svake baze podataka. Sastoje se od redova i stupaca. Naziv svakog stupca zovemo atributom, dok su svi atributi jedne tablice entiteti, odnostno, objekti o kojim prikupljamo podatke.



Slika 5: Prikaz tipa podataka

U svojoj bazi podataka imam ukupno 7 tablica. U tablici „Trener“ popisala sam sva imena i prezimena trenera pojedinih klubova, datum rođenja te mjesto rođenja. U tablici „Kapetan“ napravila sam isto kao i u prethodnoj tablici samo što sam unijela podatke o kapetanima a ne o trenerima. U tablici „Osnivanje“ navedeni su svi podaci o osnivanju klubova a to su šifra osnivanja, šifra godine, mjesto osnivanja, država, igralište, i godina osnivanja kluba. Tablica „Godina“ sadrži šifru godine i godinu osnivanja. Tablica „Igrači“ sadrži neke igrače pojedinih klubova koje sam navela. Ta tablica sadrži šifru igrača, ime, prezime te klub. Tablica „Status“ pokazuje za svaki pojedini klub status o danim ukupnim golovima i postotak posjeda lopte u prethodnoj godini. I posljednja tablica je tablica „Klub“ koja sadrži sve klubove koje sam odabrala, trenere i kapetane tih klubova, godinu osnivanja kluba te broj golova koje je taj klub zabio u prethodnoj godini. Slika 6 prikazuje abecedni popis svih tablica.

Slika 6: Popis tablica

3.1.2. Veze

Kako bi naši podaci uvijek bili točni i konzistentni, naša baza podataka mora biti redovito ažurirana sa svim promjenama koje smo napravili na bilo kojem mjestu u bazi podataka. Zbog toga tablice moramo povezati.

Da bi stvorili veze između tablica, trebamo odrediti primarni ključ. Bitno je napomenuti da on uvijek mora sadržavati neku vrijednosti (ne smije biti null) te svaki red u tom stupcu mora biti jedinstven. U drugoj tablici mora postojati stupac čija vrijednost odgovara vrijednosti prve tablice. To nazivamo vanjskim ključem. Na taj način nismo ponavljali sve podatke u drugoj tablici, nego uz pomoć vanjskog ključa možemo doći do podataka iz prve tablice s obzirom

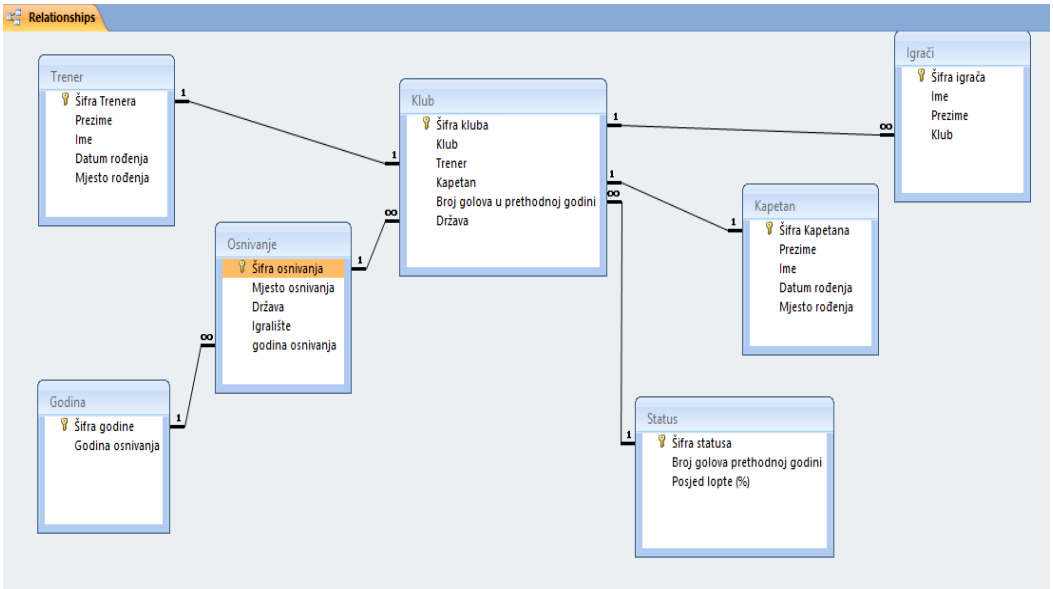
da je svaki red jedinstveno određen pa tako i prepoznatljiv. Nadalje, kad jednom definiramo vanjski ključ, sustav nam ne dopušta da upišemo šifru iz stupca određenog primarnim ključem koja ne postoji. To je najlakše spriječiti tako da napravimo „Lookup“ na podatke iz primarne tablice. Na taj način u drugoj tablici možemo odabrati samo one šifre koje postoje.

Vezama možemo odrediti njihov red, način vezivanja objekta te način sudjelovanja objekta u vezi. Red veze predstavlja broj tipova objekata koji tvore vezu. Ako vezu tvore objekti istog tipa, tada je ta veza unarna veza. Sukladno tome, ako je veza između objekata dvaju tipova, tada je to binarna veza. Imamo i ternarne kao i veze višeg reda. Prema tome, između dviju tablica može postojati veza jedan – jedan (1:1), jedan – više (1:M) ili više – više (M:N).

U vezi jedan – jedan, jedan slog iz prve tablice odgovara samo jednom slogu iz druge tablice koje su međusobno povezane. Kod veze jedan – više, jedan slog iz prve tablice može biti povezan s više slogova iz druge tablice. U takvim vezama vanjski ključ dodajemo na stranu više. Vezu više – više možemo zamisliti kao dvije veze jedan – više koje su međusobno povezane novom zajedničkom tablicom. U toj novoj tablici primarni ključ je najčešće sastavljen od oba vanjska ključa.

Objekti u vezi mogu sudjelovati na dva načina: obavezno ili opcionalno. Ako jedan objekt mora sudjelovati u vezi, onda je način sudjelovanja tog objekta obavezan. Međutim, ukoliko objekt ne mora sudjelovati u vezi, tada je njegov način sudjelovanja opcionalan.

U svojoj bazi podataka imam uglavnom veze jedan – više (Slika 7). Kako bih pobliže objasnila te veze, navest ću primjer iz svoje baze. U jednom klubu možemo imati više igrača, a više igrača možemo imati u jednom klubu. Dakle, te dvije tablice tvore vezu jedan – više.



Slika 7: Veze

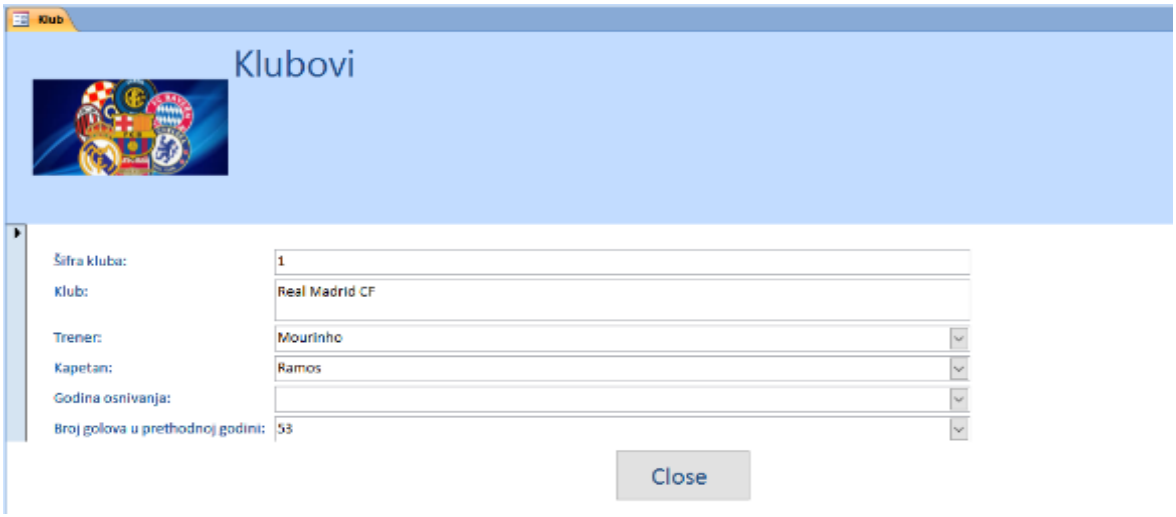
3.2. Obrasci

Obrazac ili forma je jedna od komponenti baze podataka koja je namijenjena interakciji s korisnikom. U formama nisu prikazane tablice kao takve, već su svi podaci iz tablica vizualno oblikovani i personalizirani prema željama korisnika. Forme nam služe za dodavanje i brisanje podataka, ali u vizualno uređenom izgledu.

Uz pomoć forme možemo stvoriti i tzv. Switchboard, tj. formu iz koje možemo otvoriti druge forme, pokrenuti upit ili tablicu. Formu možemo kreirati na nekoliko načina:

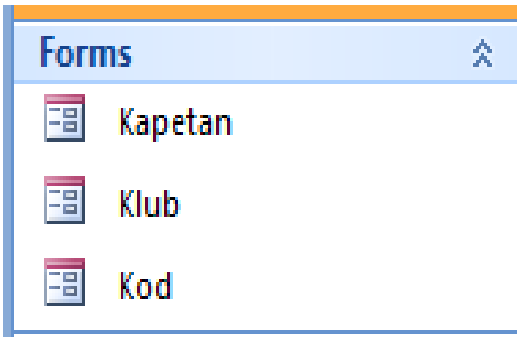
1. pomoću Form Wizarda – čarobnjaka pomoću kojeg možemo izraditi osnovne forme ili podbforme,
2. Form Designa u kojem sami stvaramo forme i dodajemo kontrole po našoj želji,
3. pomoću Pivot Table koji stvara formu sa Excelovom tablicom,
4. Datasheet Forme koja automatski stvara formu kao tablicu, itd.

Korisnik sam može prilagoditi izgled forme kao npr. font, pozadinu, boje, labele i slično. Može dodati logo, slike, gumbe, okvire, razne linije i text boxove, kreirati grafikone ili pak podesiti zaglavlje i podnožje forme. Forme se mogu povezati sa drugim objektima kao što su web stranice i datoteke, a svi podaci u formi mogu se pregledavati ili izmijeniti uz pomoć padajuće liste (Slika 8) ili list boxa.



Slika 8: Primjer padajuće liste u obrascu

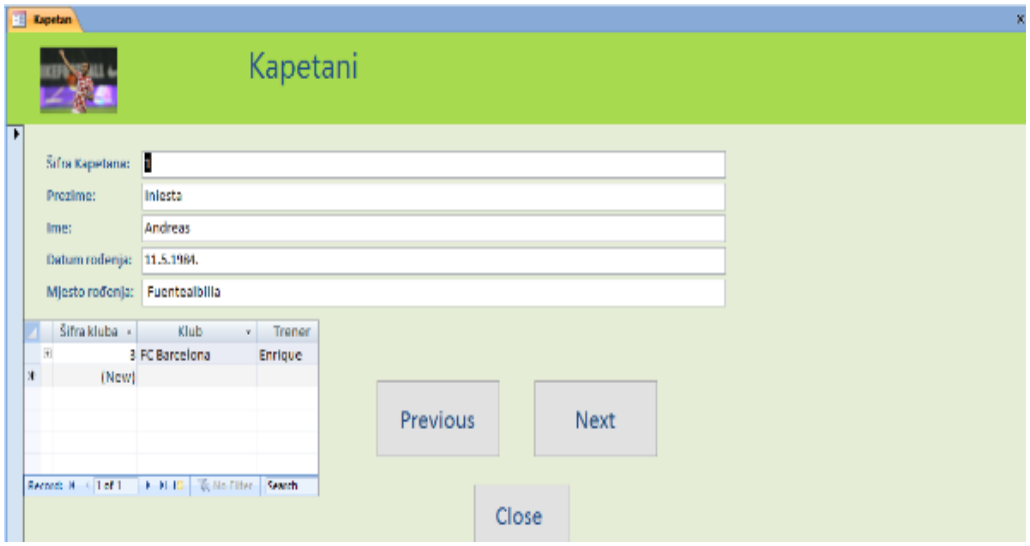
U svojoj bazi podataka izradila sam jednostavne forme. Ukupno imam 3 forme (Slika 9).



Slika 9: Popis formi

3.2.1. Obrazac „Kapetan“

Primjer obrasca koji sam izradila je obrazac pod nazivom „Kapetan“ (Slika 10). U Design Viewu sam podesila izgled prema vlastitoj želji. Odredila sam izgled forme, dodala logo koji asocira na tu formu te podesila boju. U ovoj formi korisnik ima uvid u sve kapetane te podatke o svakom pojedinom kapetanu kao što su Šifra kapetana, Prezime, Ime, Datum rođenja te Mjesto rođenja.Pomoću gumba „Next“ vidimo podatke za ostale kapetane, a gumbom „ Previous“ vraćamo se natrag na prijašnje kapetane. Te na kraju imamo još gumb „Close“ s kojim možemo zatvoriti ovaj obrazac.



Slika 10: Obrazac "Kapetani"

3.3. Upiti

Upite koristimo za pregledavanje, dodavanje, mijenjanje ili brisanje podataka. Upiti su vrlo praktični za filtriranje i sažimanje podataka te za izvršavanje proračuna s podacima jer možemo kombinirati podatke iz više tablica. Pomoću upita na vrlo brz način možemo dobiti određene podatke na jednom mjestu koji su nam potrebni, a izravnim pogledom na tablicu to možda nebi bilo moguće. Mogu nam služiti i kao priprema podataka za izradu izvješća ili obrasca. Dakle, upit je zapravo zahtjev za rezultatima ili akcijom podataka.

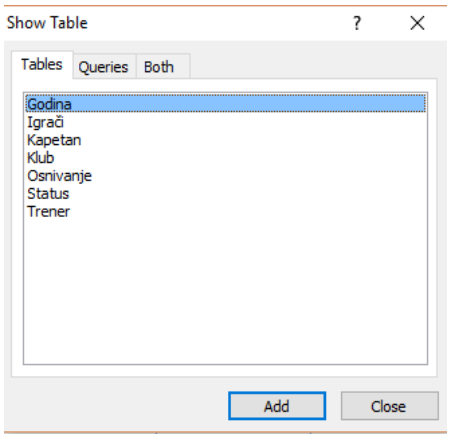
Upite možemo kreirati pomoću Query Wizarda ili pomoću Query Designa. Kod kreiranja upita možemo birati želimo li jednostavan upit ili unakrsni upit (Crosstab Query), želimo li pronaći duplikate ili neuparive upite (Find Unmatched Query Wizard).

Postoje dvije vrste upita – upiti za odabir te akcijski upiti. Upiti za odabir su oni upiti koje koristimo za dohvat podataka iz tablice ili za izvođenje izračuna. Akcijski upiti su oni upiti koji dodaju, mijenjaju ili brišu podatke.

U kriterij upita možemo dodavati i slijedeće operatore: = (jednako), < > (različito od), < (manje od), <= (manje ili jednako od), > (veće od), >= (veće ili jednako od). Možemo koristiti i logičke operatore: AND (i), OR (ili) i NOT (ne). Na taj način možemo odrediti prikaz samo onih podataka koji zadovoljavaju naše uvjete.

3.3.1. Upit Klubovi sa >30 golova

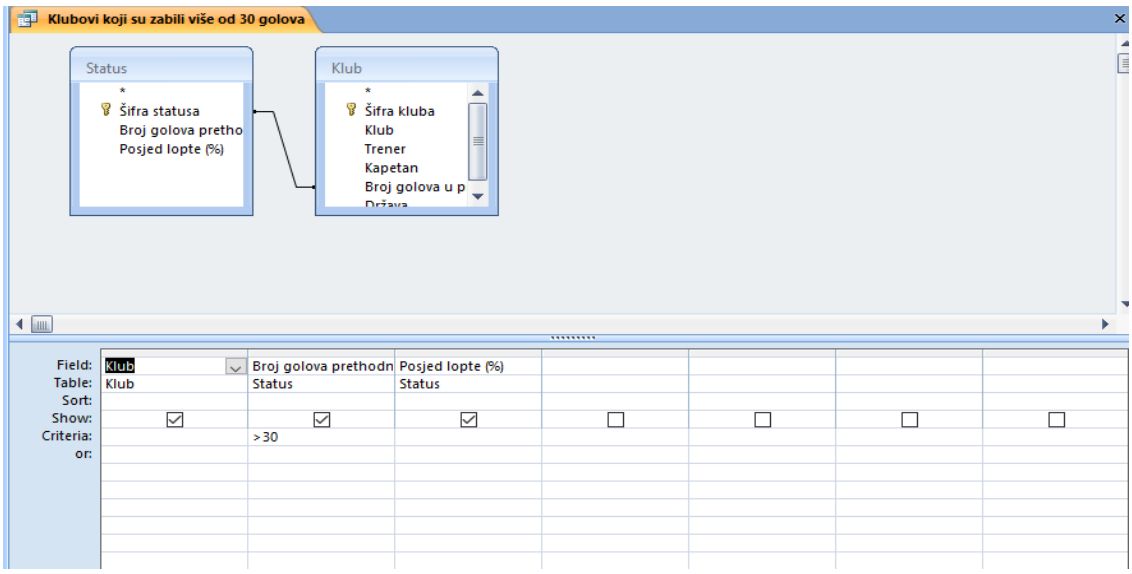
Na mojem primjeru upit vraća sve one klubove koji imaju više od 30 zabijenih golova. Taj upit kreirala sam pomoću Query Design-a. Nakon što sam kreirala upit, otvara nam se pregled u Design view-u gdje možemo dodati tablice koje su nam potrebne kako bi napravili upit po želji.



Slika 11: Pregled u Design view-u

U konkretnom slučaju ja sam odabrala tablicu status i klub. Stavke koje želim da mi se pojavljuju u upitu su: Klub, Broj golova prethodne godine i Posjed lopte (%). Kod stavke

„Criteria“ postavila sam kriterij, odnosno uvjeti da želim sve one klubove koji imaju zabijene golove u prethodnoj godini veće od 30.



Slika 12: Klubovi sa >30 golova

Rezultat takvog upita je tablica koja sadrži samo one podatke koji zadovoljavaju kriterij koji smo unijeli kod pokretanja (Slika 13).

Klubovi koji su zabili više od 30 golova

Klub	Broj golova	Posjed lopte
Real Madrid CF	53	61,00%
FC Barcelona	35	51,00%
GNK Dinamo Zagreb	35	51,00%
Manchester City FC	38	56,00%
FC Bayern München	32	60,00%
Manchester United F	32	60,00%
Atlético de Madrid	36	58,00%

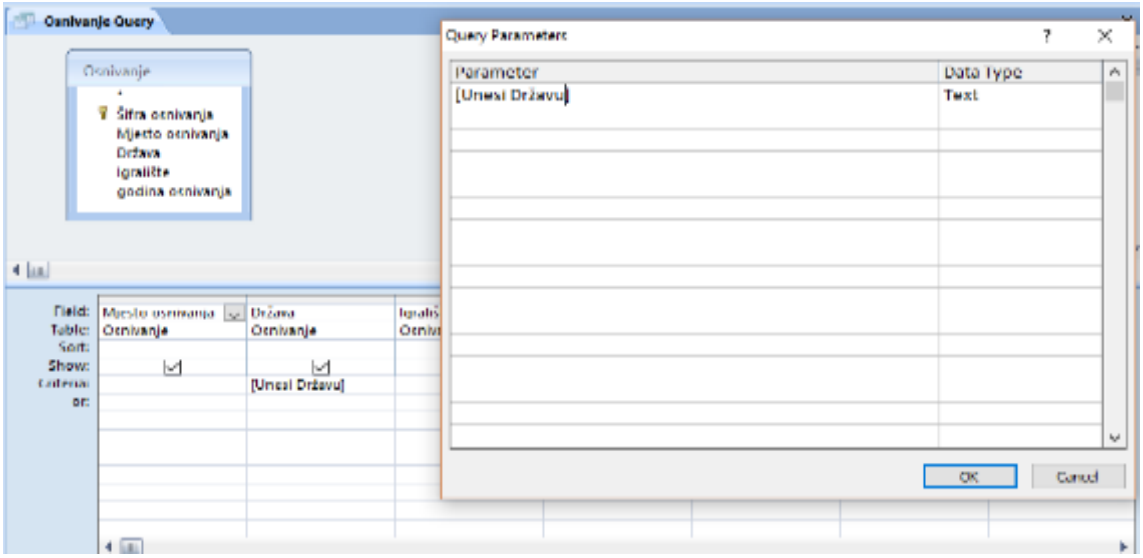
Slika 13: Upit klubova koji imaju > 30 zabijenih golova

3.3.2.Upit igrališta neke države

Drugi primjer upita radila sam pomoću Query Wizarda. Ovaj upit vraća popis igrališta koja se nalaze unutar neke države. Ponovo nam se otvara područje gdje biramo koje tablice i atribute želimo da nam se prikazuju u upitu. U konkretnom slučaju ja sam odabrala tablicu

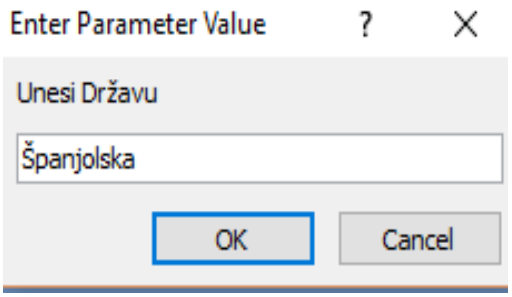
osnivanje. Atribute koje želim imati u ovom upitu su: Mjesto osnivanja, Godina osnivanja, Država te Igralište.

Također, vrlo važno je napomenuti da sam u ovom upitu postavila parametar koji traži da prvo upišemo Državu i zatim nam upit vrati sva Igrališta unutar neke Države koju smo upisali. Evo primjera na slici:



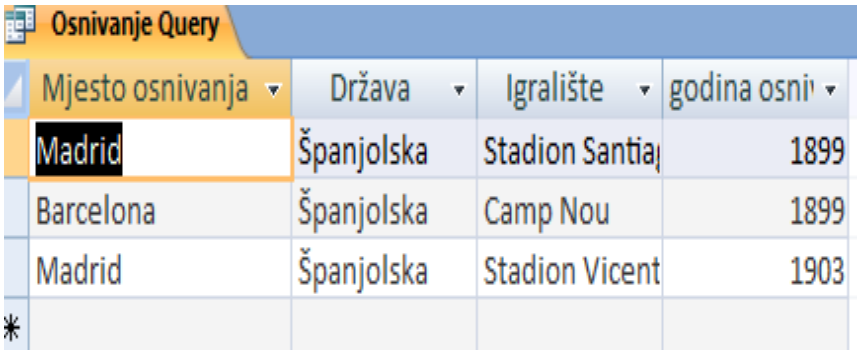
Slika 14: Postavljanje parametra

Nakon što smo definirali sve uvjete, možemo pokrenuti upit. Ovaj upit prvo nas traži da unesemo Državu. U konkretnom slučaju ja sam upisala „Španjolska“.



Slika 15: Unos parametra

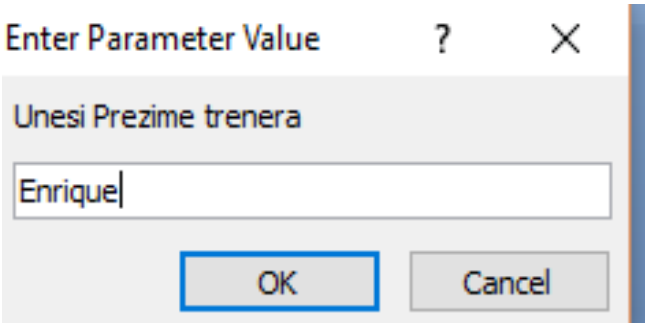
Nakon toga, možemo vidjeti podatke koje nam je upit vratio (Slika 16).



Slika 16: Rezultat upita

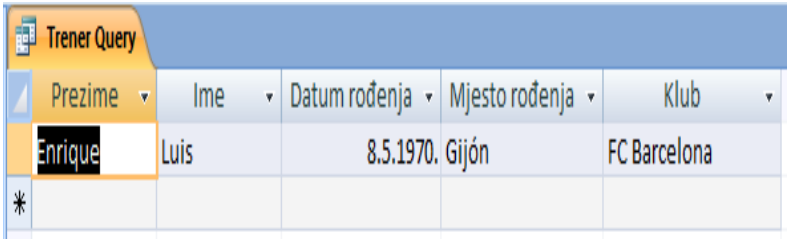
3.3.3. Upit o treneru

Upit o treneru radila sam na sličan način kao i na prethodni. Koristila sam isto parametar gdje trebamo upisati prezime Trenera. U ovom slučaju napisala sam “Enrique” (Slika 17).



Slika 17: Unos parametra

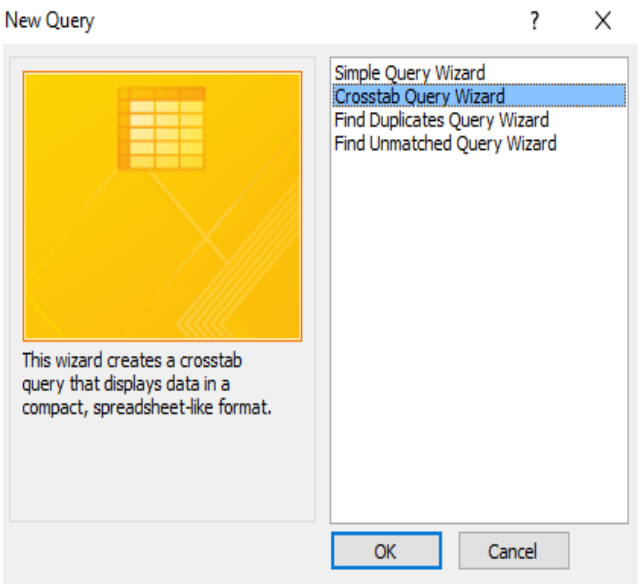
No, tada će nam se pojaviti samo jedan redak o treneru koji smo odabrali te njegovi ostali podaci koje smo odabrali. Rezultat upita se može vidjeti na sljedećoj slici (Slika 18):



Slika 18: Rezultat parametra

3.3.4. Crosstab upit

Crosstab upit se drugim riječima još naziva i unakrsni upit. Takva vrsta upita koristi podatke iz samo jedne tablice ili upita. Služi za pojednostavljen prikaz i lakše snalaženje i uspoređivanje podataka. Može se izraditi pomoću Query Wizarda. Klikom na Query Wizard otvori nam se prozor gdje možemo odabrati kakav upit želimo. Ja sam odabrala Crosstab Query Wizard.



Slika 19: Odabir crosstab upita

U liniju reda možemo odabrati jedan ili više stupaca iz tablice, dok u liniju stupaca možemo odabrati samo jedan. Tada odabiremo polje koje će se analizirati te vrstu izračuna (count, sum, average...). U svojoj sam bazi izradila jedan Crosstab upit. Uzela sam podatke iz tablice „Igrači“.

U liniji reda sam stavila Klubove, dok su u liniji stupaca igrači. Zadatak upita je bio da prikaže koji igrač igra u kojem klubu te da presjek klubova i igrača označi s brojem 1. U drugom stupcu možete još vidjeti da sam stavila sumu igrača koji igraju u pojedinom klubu.

Klub	Total Of Šifr	Carlos	Cristiano	Diego	Ivan	Jamez	Lionel	Manuel	Thomas	Toni
Real Madrid Cl	3		1			1				1
FC Bayern Mür	2							1	1	
FC Barcelona	2				1		1			
Chelsea FC	1			1						
AC Milan	1	1								

Slika 20: Crosstab upit

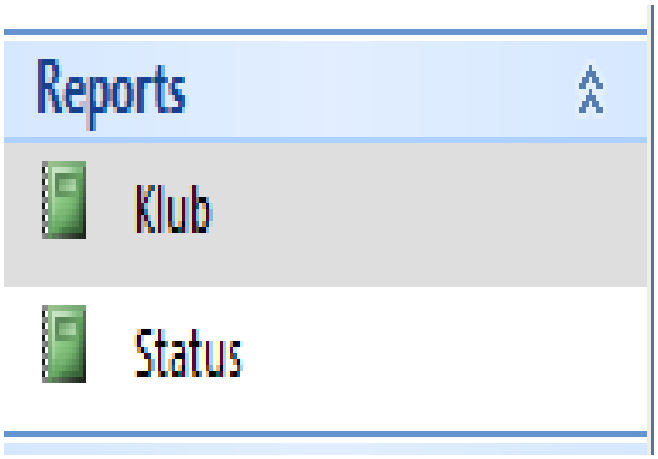
3.4. Izvještaji

Izvještaje koristimo kada želimo određene podatke pripremiti za ispis. Možemo ih kreirati pomoću Report Wizarda ili pomoću Report Designa gdje cijeli izvještaj radimo ispočetka. Također, možemo odabrati i tablicu ili upit i u kartici Create kliknuti na izvještaj. Otvara nam se gotov izvještaj spreman za ispis.

Međutim, u izvještajima ne možemo mijenjati, brisati ili dodavati podatke. Ali, možemo mijenjati dizajn, stavljati slike, računati prosjeke ili sume i slično. Izračun na kraju izvještaja može biti od velike koristi korisniku koji to ne mora sam raditi na kalkulator. Kako bi mogli računati, potrebno je prvo napraviti tekstualno polje (eng. Text box). U listi svojstva (eng. Property Sheet) moramo otvoriti svojstva tog tekstualnog polja te u njegov kontrolni izvor (eng. Control Source) pomoću izrađivača izraza (eng. Expression builder) upisujemo izraz, tj. formulu po kojoj želimo dobiti rezultat. To možemo napraviti i na jednostavniji način: odaberemo polje za koje želimo dobiti zbroj, prosjek ili nešto drugo. Zatim kliknemo na gumb Totals i odaberemo željenu operaciju. Stvara nam se novo tekstualno polje u koje se upisuje rezultat.

Izvještaje možemo napraviti na temelju podataka iz samo jedne tablice ili upita ili više njih zajedno. Ako smo odabrali više izvora podataka, tada čarobnjak ispituje odnose među njima te utvrđuje kakav bi mogao biti pogled između njih, tj. koju razinu grupiranja je moguće ostvariti. U čarobnjaku možemo odabrati način na koji podatke želimo sortirati – uzlazno ili silazno. Također, u izvještaj je poželjno dodati i broj stranice te datum. U Print Previewu možemo odabrati veličinu papira te orijentaciju stranice (Portrait ili Landscape).

U svojoj bazi podataka imam 2 izvještaja (Slika 21). Kreirala sam ih pomoću Report Wizarda.



Slika 21: Popis izvještaja

U mojem slučaju izvještaj sam kreirala na postojećoj tablici, odnosno na način da sam prvo označila tablicu po kojoj želim napraviti izvještaj. Ovakav izvještaj pokaže sve klubove te trenere, kapetane, godinu osnivanja te broj golova u prethodnoj godini tih klubova.

Također na dnu stupca „Broj golova u prethodnoj godini“ je suma svih golova zajedno. U zaglavlju izvještaja postoji datum i vrijeme kada se izvještaj pregledava te na samom dnu imamo gumb „Close“. Klikom na njega omogućava nam zatvaranje ovog izvještaja. Prikaz ovog izvještaja možete vidjeti na slici 22.

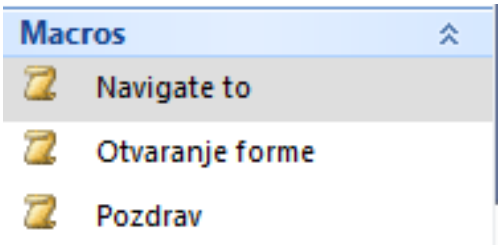
Slika 22: Izještaj klubova

3.5. Makro naredbe

Makro naredbe se koriste za izvršavanje određene serije akcija. Mogu biti samostalne ili u sklopu izvještaja i formi. Kada kliknemo na gumb Makronaredbe, otvara se navigacijsko okno u koje upisujemo akcije, argumente i komentare. Ispod toga biramo vrstu i ime objekta te uvjete koji su za svaku naredbu drugačiji. Klikom na gumb Run, pokreće se makro naredba.

Makro naredba je jednostavniji način na koji možemo odrediti aplikaciji koje naredbe treba izvršiti. Teži način za to je pomoću VBA (eng. Visual Basic for Applications). Međutim, s obzirom da je većini ljudi lakše odrediti naredbe bez kodova, češće se koriste makro naredbe.

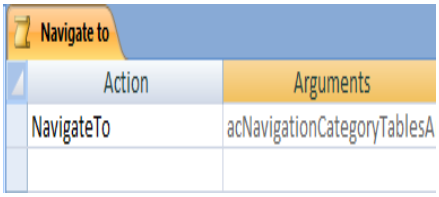
U svojoj bazi podataka izradila sam tri makro naredbe (Slika 23).



Slika 23: Makro naredbe

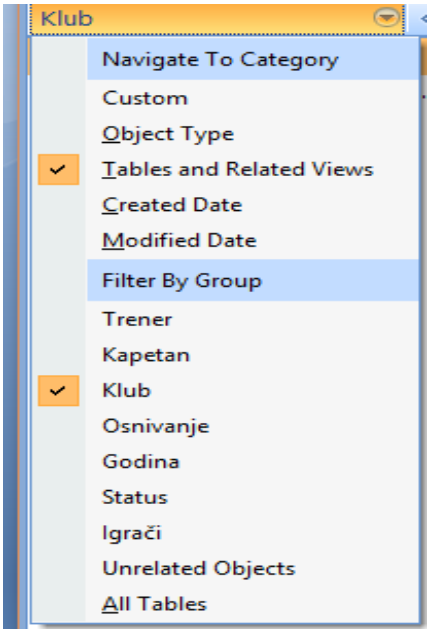
3.5.1. Navigate to

Prva makro naredba koju sam napravila je naredba Navigate to. Kada pokrenemo tu makro naredbu, ona nas prebacuje na željeno mjesto u bazi podataka. U popisu akcija odabrala sam naredbu Navigate to (Slika 24).



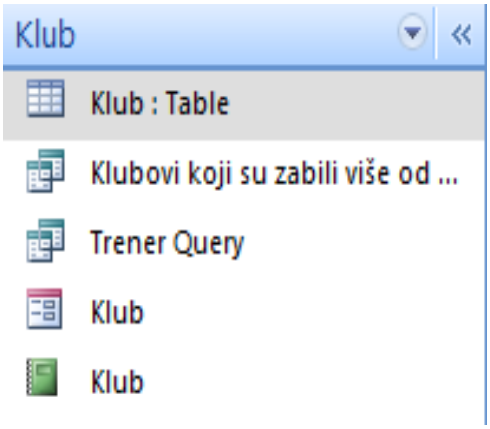
Slika 24: Navigate to

U argumentima naredbe sam odabrala da mi se prikažu sve tablice i povezani sadržaj te sam odabrala grupu „ Klub“.



Slika 25: Orjentiranje i sortiranje

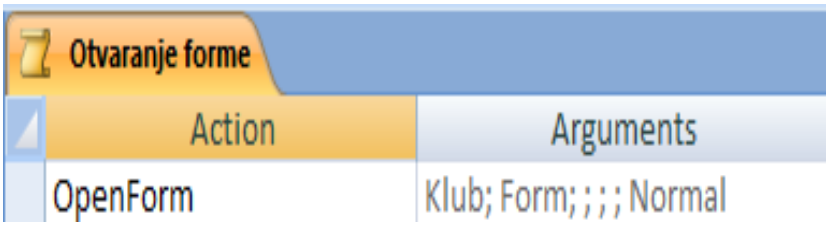
Pritiskom na gumb Run (Pokreni), aplikacija mi prikazuje sve tablice, upite, forme i izvještaje koji su povezani s odabranom grupom (Slika 26).



Slika 26: Navigate to rezultat

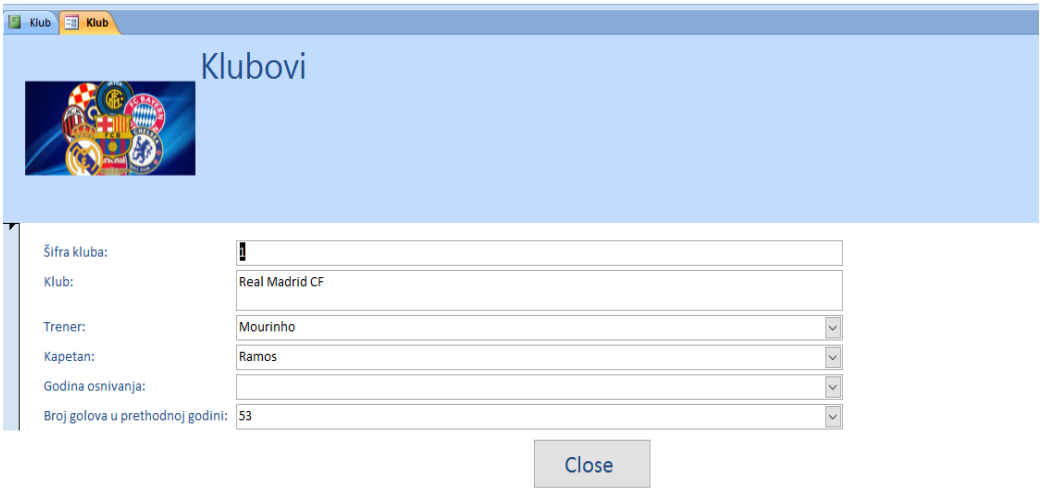
3.5.2. Open form

Druga makro naredba koju sam izradila je Open form. U popisu akcija odabrala sam akciju Open form.



Slika 27: Open form

Pod argumentima akcije odabrala sam naziv forme „Klub“ te pogled u kojem će se forma otvoriti kao Form. Prozor sam odabrala kao normalan. Nakon što pokrenem naredbu, otvara se forma „Klub“ (Slika 28).



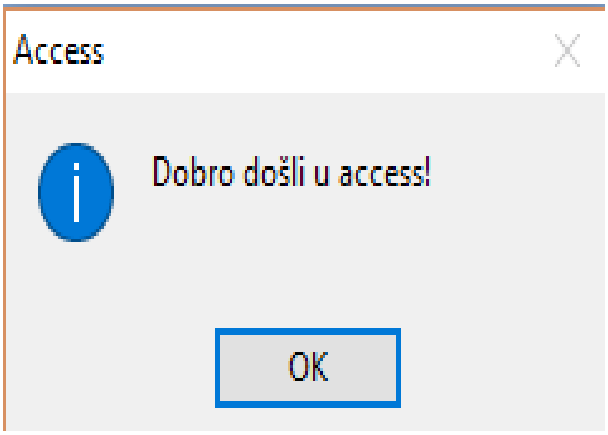
Slika 28: Open form „Klub“

3.5.3. Poruka

Treću makro naredbu koju sam izradila je naredba Poruka. Akciju koju sam odabrala je akcija MsgBox.

Slika 29: MsgBox

U argumentima akcije sam kao poruku upisala „Dobro došli u access!“. Zvuk poruke sam isključila, dok sam kao tip poruke odabrala informaciju. Kao naslov poruke sam stavila „Access“. Nakon pokretanja naredbe, otvara se prozor koji je prikazan na slici 30.



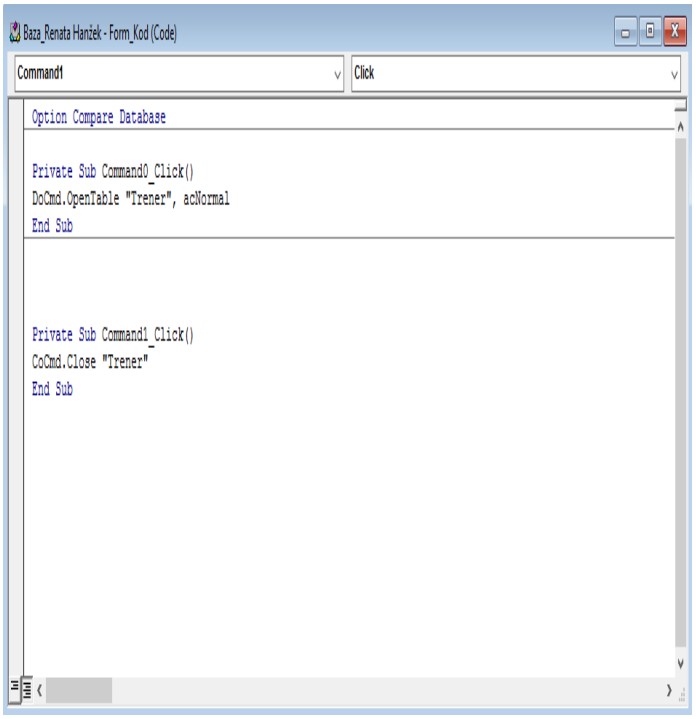
Slika 30: Poruka

3.6. VBA

VBA je verzija programskog jezika Visual Basica koja je izrađena za programiranje u Microsoft Officeu i drugim aplikacijama. Korisnik ručno unosi naredbe koje bi aplikacije trebale izvoditi. Bitno je napomenuti da VBA ima daleko manje mogućnosti za razliku od standardnog Visual Basica.

U svojoj sam bazi pomoću VBA kreirala dva gumba. Pomoću jednog od njih otvara se Tablica „Trener“, a pomoću drugog se ta tablica zatvara. Do VBA sam došla na način da sam napravila gumb u Design Viewu i u Property Sheetu sam otvorila karticu pod nazivom Event. Odabrala sam da se naredba izvrši na jednostruki klik. Koristila sam naredbu DoCmd

pomoću koje sam jednom gumbu odredila otvaranje tablice (DoCmd.OpenTable), a drugom gumbu zatvaranje tablice (DoCmd.Close).



Slika 31: Kod u VBA

4. Zaključak

Danas u svijetu postoje mnoge tehnologije za rad sa spremljenim podacima. No bitan je pravi izbor tehnologije koji je bitan za rad konkretne aplikacije. U većini slučajeva baze podataka su vrlo korisne. Bazu podataka možemo još nazvati i organiziranom zbirkom podataka. Da bi baza podataka funkcionirala njome mora upravljati određeni sustav za upravljanje bazama podataka (DBMS). On nam omogućava rad s bazama podataka odnosno definiranje baze podataka, upis podataka u bazu, ispis podataka iz baze i obradu podataka smještenih u bazi.

Kako bih bolje objasnila DBMS i upravljanje bazom podataka izradila sam vlastitu bazu podataka svjetski poznatih nogometnih klubova u sustavu MS Access. Ova aplikacija namijenjena je lakšoj i bržoj upotrebi u trgovini te s ciljem kako bi se što više smanjilo korištenje papira te tonera. Na taj bi se način smanjili troškovi poduzeća.

U MS Accessu možemo izraditi bazu podataka u kojoj se podaci automatski u cijeloj bazi ažuriraju ovisno o tome jesmo li koji slog promijenili, dodali ili obrisali. U tu se svrhu u bazi podataka, osim tablica, mogu izraditi i upiti, izvještaji, makronaredbe i forme. Za one ambicioznije, MS Access podržava i verziju Visual Basica namijenjenu aplikacijama – VBA u kojoj se pomoću programskog koda mogu upisati naredbe koje je potrebno izvršiti.

U svojoj sam bazi izradila sedam tablica koje sam povezala vezama jedan – više. Izradila sam i obrasce pomoću kojih možemo mijenjati sadržaj tablica, ali u vizualno ljepšem okruženju. Pomoću upita i kriterija izdvojila sam određene podatke. Nakon toga izradila sam izvještaje koji nam prikazuju kako bi to izgledalo ispisano na papiru. Uz pomoć makro naredbi, aplikacija je brže i lakše izvela naredbe koje sam odabrala.

Na taj sam način kreirala cijelu aplikaciju koja je vrlo korisna i jednostavna za upotrebu te na taj način olakšava svakodnevne poslove zaposlenicima u poduzeću informatičkom opremom.

5. Literatura

Knjige:

1. Rabuzin Kornelije (2014). *SQL – napredne teme*. Varaždin
2. Rabuzin Kornelije (2011). *Uvod u SQL.* Čakovec: Zrinski d.d.
3. Radovan Mario (1993). *Baza podataka.* Zagreb: Informator
4. Malenković Mirko i Rabuzin Kornelije (2016). *Uvod u baze podataka.* Varaždin

Internetski izvori:

1. Germanijak.hr (2015). *Poznat popis najboljih 500 klubova svijeta*. Preuzeto s http://www.germanijak.hr/vijesti/poznat-popis-najboljih-500-klubova-svijeta-rijeka-ispred-dinama-bayern-izmedju-spanjolaca/5638
2. Socerboard (2012-2018). *Nogomet.* Preuzeto s http://www.scoreboard.com/hr/nogomet/
3. Real Madrid (2018). *Squad*. Preuzeto s http://www.realmadrid.com/en/football/squad
4. FCBarcelona (2018). *Players*. Preuzeto s https://www.fcbarcelona.com/football/first-team/staff/players
5. Freeservers (2017). *Baza podataka.* Preuzeto s http://tecajevi.freeservers.com/isbaza.htm
6. Element (1996-2018) *Uvod u baze podataka*. Preuzeto s https://element.hr/artikli/file/1710
7. Znanje.org (2001-2005) *Oracle*. Preuzeto s http://www.znanje.org/knjige/computer/access-/access_01/oracle.htm

6. Popis slika