Izrada baze podataka za potrebe upravljanja radnim nalozima

Krajcar, Gordan

Politehnika Pula
Visoka tehničko-poslovna škola

ZAVRŠNI RAD

Izrada baze podataka za potrebe upravljanja radnim nalozima

Kolegij: Primjena elektroničkih računala

Student: Gordan Krajcar

Mentor: Kristian Matas, predavač

Pula, rujan 2019.

Sažetak:

Baza podataka (*Database*) i Sustav za upravljanje bazom podataka predstavljaju informacijske sustave kojima je cilj obavljanje operacija nad velikom količinom sustavno pohranjenih podataka na korisniku prilagođen način. Prilikom oblikovanja baze podataka koristimo metode i korake izrade koji nam omogućuju realizaciju optimalnog i funkcionalnog rješenja za pohranu, dohvaćanje i operacije nad podacima. Jedan od takvih sustava je i MS Access koga mnogi nazivaju samo RAD (*Rapid Application Development*) alatom, dok ga drugi opisuju kao nešto mnogo više od sustava za upravljanje bazom podataka. U koracima kojima opisujemo praktični dio rada približiti ćemo se objema tvrdnjama.

Abstract:

The database and the databese management system are information systems designed to perform operations on large amount of systematically stored dana in a user-friendly manner. When designing a database, we use methods and design steps that allow us to implement an optimal and functional solution for storing, retrieving and performing data operations. One such system is MS Access, which many refer to just a RAD (Rapid Application Development) tool, while others describe it as much more than a database management system. In the steps where we describe the practical part of the work, we will address both these claims.

Ključne riječi:

Baza podataka, Sustav za upravljanje bazama podataka, MS Access, metodologija modeliranja informacijskih sustava, E-V dijagram, relacijski model podataka, SQL

Keywords:

Database, Database Management System, MS Access, methodology of information systes modeling, E-R diagram, relational dana model, SQL

SADRŽAJ:

1. UVOD

1.1 Opis i definicija problema

Poduzeće „Altus“ d.o.o. bavi se između ostaloga i pružanjem dimnjačarskih usluga.

Dok je sam princip čišćenja i korištenja dimovodnih kanala stoljećima ostao manje-više nepromijenjen, današnji modeli birokracije zahtijevaju sve veći dio radnog vremena.

Pregledom dokumenata utvrđeno je da za svaki očišćeni dimovodni kanal, transparentno poslovanje zahtijeva ispunjavanje tri različita obrasca sa kombinacijama istih podataka, a to su:

1) Stručni dimnjačarski nalaz
2) Radni nalog
3) Račun

Gradska uprava davatelja dimnjačarskih usluga obvezuje uvjetom stavke ugovora o koncesiji da vodi evidencijsku listu korisnika i čišćenja dimovodnih kanala, te da jednom godišnje preda pisano izvješće Javnoj vatrogasnoj postrojbi za evidenciju i analizu.

Informatička obrazovanost dimnjačarskog kadra je podosta niska i većina se ovih obrazaca ispisuje ručno. Ovakvim se načinom rada, od ukupnog radnog vremena koristi 20-30% vremena na ispunjavanje obrazaca.

1.2 Cilj i svrha rada

Cilj ovog rada je proširiti znanje iz područja baza podataka, te ga implementirati u rješavanje zadane problematike. Prvenstveno se to odnosi na sistematizirano opće objašnjenje postupaka i koraka pri izradi baze podataka, te korištenje tih istih postupaka za izradu konkretne baze podataka za potrebe poduzeća „Altus“ d.o.o.

Ideja je izraditi bazu podataka koja će biti prilagođena potrebama i informatičkoj obrazovanosti korisnika. Realizirani model baze podataka trebao bi korisnika voditi od početnog unosa podataka, pa do ispisa izvješća potrebnih za poslovanje u obliku stručnog

dimnjačarskog nalaza i radnog naloga, smanjujući mu na minimum mogućnost greške pri korištenju.

1.3 Hipoteza

U današnjem poslovanju informatička pismenost je neophodna za opstanak na tržištu. Korištenjem informatičkih sustava skraćujemo vrijeme potrebno za organizaciju i finalizaciju radnih zadataka. Nevelikim ulaganjima u informatičke sustave možemo instantno povećati produktivnost poslovanja, i pri tome štedjeti naše najvažnije resurse – ljude.

1.4 Metode rada

Za vrijeme izrade ovog završnog rada korištene su slijedeće metode rada:

- Metoda analize (proučeni su svi sakupljeni podaci o izradi baze podataka, analiziran način poslovanja i rješenja sličnih sustava)
- Metode sinteze (rezultati analize potreba poslovanja „Altus“ d.o.o., te znanja o izradi i implementaciji baza podataka objedinjeni su u funkcionalno rješenje)

1.5 Struktura rada

Ovaj završni rad je podijeljen u dva glavna dijela. U prvom dijelu se upoznajemo sa pojmom baze podataka, sustava za upravljanje bazom podataka, različitim vrstama modela podataka. Obrađujemo također raslojavanje baze podataka na tri razine, te se upoznajemo sa koracima kojima se ostvaruje izrada baze podataka. Prikazan je i osvrt na dvije metodologije izrade informacijskih sustava, MIRIS metodiku i CASE*Method.

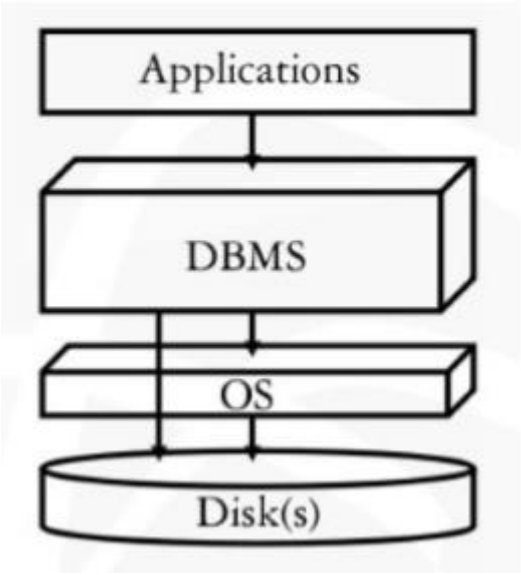
U drugom dijelu opisujemo realizaciju koraka pri izgradnji baze podataka za potrebe Altus d.o.o.-a, te primjerima izrade objekata baze i primjerima manipulacije podataka putem SQL-a, VBA editora i funkcija Macro Buildera, izvedenim u programu Microsoft Access.

2. Baze podataka i sustav za upravljanje bazom podataka

Baze podataka se više desetljeća razvijaju kao odgovor na problematiku trajnog pohranjivanja velikih količina podataka u vanjskoj memoriji računala, te manipuliranje i efektivno provođenje izmjena nad tim istim podacima.

Trajno pohranjivanje podataka je jedna od glavnih opcija korištenja računala, te se razvijala od početaka razvoja računala. Pohrana podataka podržana je u svim programskom jezicima, te se koristeći njima može stvarati trajne kolekcije podataka na disku i razviti aplikacije koje se koriste tim podacima.

Govoreći o bazama podataka mislimo na višu razinu od rada s podacima koji podržavaju klasični programski jezici. Bazični princip tehnologije baza podataka je da pojedinačne aplikacije ne stvaraju svoje vlastite datoteke na disku, već sve aplikacije koriste zajedničku objedinjenu bazu podataka. Aplikacija pritom ne pristupa direktno podacima na disku već na posredan način preko specijaliziranog programa koji je zadužen za opsluživanjem podacima iz baze podataka. Takvog posrednika između baze podataka i korisničke aplikacije nazivamo sustavom za upravljanjem bazama podataka.



Slika 1: Shema pristupa podacima

Izvor: http://uni-mo.sve-mo.ba/~goran/nastava/BAZE_Nastava.pdf, str. 14 (3.2.2019.)

„Baza podataka (engl.Database) predstavlja neredundantni skup podataka o stanju sustava strukturiran na način opisan u shemi baze podataka. Baza podataka je skup podataka o svim pojavljivanjima entiteta, svih veza i svih njihovih atributa opisanih u shemi baze podataka.“[1]

Bazom podataka nazivamo skup međusobno povezanih podataka koji su, pohranjeni u vanjskoj memoriji računala, istovremeno dostupni različitim korisnicima i aplikacijskim programima. Različite manipulacije podacima, npr. čitanje, brisanje, ubacivanje, promjena podataka, se obavljaju posredstvom sustava za upravljanje bazom podataka SUBP (engl. Database management system, DBMS).

Slika 2: Funkcije sustava za upravljanje bazom podataka

Izvor: http://adria.fesb.hr/~zmiletic/Baze%20podataka/BP_predavanja%20-%20K.Klarin.pdf, str 5 (3.2.2019.)

Sustav za upravljanjem bazom podataka oblikuje fizički prikaz baze u skladu s traženom logičkom strukturom – u stanju je podržati razne baze koje imaju vlastitu logičku strukturu, ali u skladu s istim modelom.

Temeljne funkcije svakog informacijskog sustava, pa tako i sustava za upravljanje bazom podataka su:

1. Prikupljanje podataka
2. Upis podataka u bazu podataka
3. Obrada podataka

[1] Mile Pavlić, „Oblikovanje baza podataka“, Odjel za informatiku-Sveučilište u Rijeci,2011., str 13

4. Prikaz i ispostavljanje podataka iz baze podataka
5. Čuvanje podataka

Svoju prvu temeljnu funkciju informacijski sustav ostvaruje prihvatom i upisom podataka u bazu podataka, no temeljna funkcija organizacijskog sustava je korištenje navedenih ulaza u sustav kao osnovu za realizaciju poslovnih procesa sustava, koji stvaraju nove informacije, te kao takve predstavljaju nove ulaze u druge procese ili predstavljaju izlaze iz sustava.

Izvođenje poslovnih sustava organizacijskog sustava je druga temeljna funkcija informacijskog sustava. Organizacijski sustav je ekonomičniji i efikasniji što je više dijelova poslovnog sustava automatizirano i ne izvode ih direktno ljudi, već informacijska tehnologija uz ljudsko upravljanje i kontrolu.

Za takav funkcionalni proces potrebno je izvješćivanje, koje možemo odrediti kao proces oblikovanja podataka iz baze podataka u oblik podataka modificiran prema potrebama korisnika ili dokumentacije.

Informacijski sustav svojim radom obavlja temeljne funkcije organizacijskog sustava ako se proces poslovnog sustava može u potpunosti ili djelomično automatizirati i ugraditi u informacijski sustav.

Danas kada govorimo o bazama podataka gotovo isključivo se to odnosi na relacijske baze podataka i sustave za upravljanje relacijskim bazama podataka. Na tržištu postoje mnogi modeli sustava upravljanja bazama podataka od različitih proizvođača koji nisu međusobno kompatibilni. Realizirana su i učinkovita softverska rješenja prijenosa velikih količina podataka iz jednog sustava baze podataka u drugi.

Naziv alata	Autor	Web
CA-Datacom	Computer Associates	http://www.ca.com/us/products/product.aspx?id=1237
Ingres	University of California, Berkeley	http://www.ingres.com/
DB2	IBM	http://www-01.ibm.com/software/data/db2/
Oracle	Oracle	http://www.oracle.com/index.html
Informix	IBM	http://www-01.ibm.com/software/data/informix/
MySQL	Ulf Michael Widenius	http://www.mysql.com/about/
PostgreSQL	Michael Stonebraker University of California at Berkeley Computer Science Department	http://www.postgresql.org/docs/current/static/intro-whatis.html
Model 204	Computer Corporation	http://www.cca-int.com/prodinfo/m204.html
NonStop SQL	Tandem Computers	http://h20338.www2.hp.com/NonStopComputing/cache/76708-0-0-225-121.html
SQL Server	Microsoft	http://www.microsoft.com/sqlserver/2008/en/us/
SQLBase	Centura Software Corporation	http://www.unify.com/Products/Data_Management/default.aspx
Impromptu	Cognos Inc.	http://www-01.ibm.com/software/data/cognos/products/series7/impromptu/
Teradata	Teradata	http://www.teradata.com/t/
Progress	Progress Software Corporation	http://web.progress.com/en/index.html
Access	Microsoft	http://www.microsoft.com
dBASE	Ashton-Tate Corporation	http://en.wikipedia.org/wiki/Ashton-Tate

Tablica 1: Sustavi za upravljanje bazom podataka

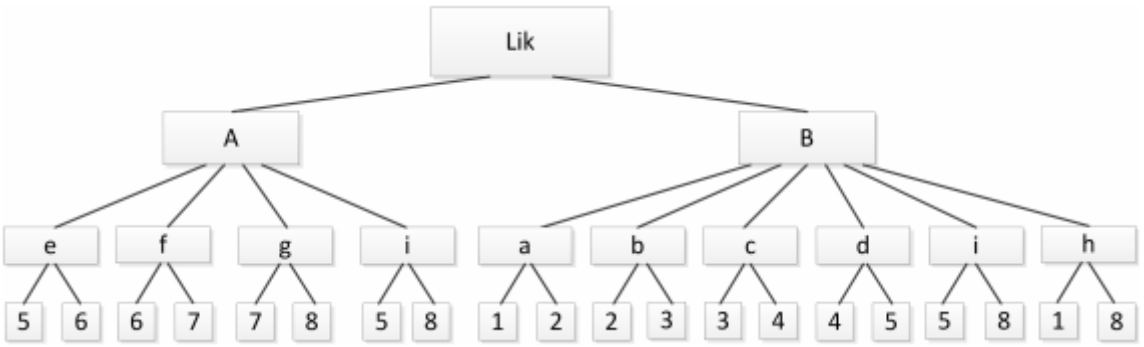
Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str 16 (3.2.2019.)

2.1 Modeli podataka za bazu podataka

Model podataka je sastavljen od skupa objekata, operacija i pravila cjelovitosti te je njime definirana logička struktura baze podataka.

2.1.1 Hijerarhijski model podataka

Razdoblje pojave hijerarhijskog modela podataka su 60-te godine prošlog stoljeća. Implementacija ovog modela nije se pojavila nakon definiranja modela, već se model definirao na osnovu IBM-ovog IMS sustava (Information Management System).



Slika 3: Hijerarhijski model podataka

Izvor: http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf, str 5 (16.2.2019.)

Konstrukcija hijerarhijskog modela temeljena je na zapisima koji se sastoje od polja, gdje se uređeni skup zapisa naziva stablo, a sama baza podataka je sastavljena od skupa stabala. Korijen stabla sastoji se od više zapisa-djece, gdje dijete može imati samo jednog roditelja (odnos roditelj-dijete). Ovdje možemo uočiti prvu manu hijerarhijskog modela podataka – nemogućnost ostvarivanja relacije više naprema više, odnosno ostvarivanje samo relacije jedan-više.

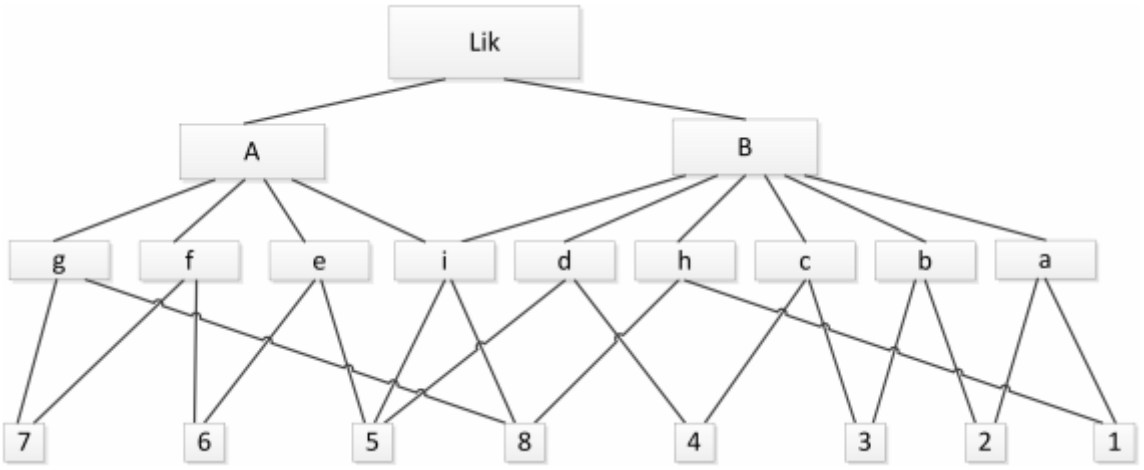
Veliki nedostatak ovog modela je redundancija podataka koja se akumulira pregledom prema korijenu stabla. Svako kompleksno brisanje i ažuriranje podataka zahtijeva poznavanje fizičkih veza između zapisa.

Jezik za manipulaciju podacima u hijerarhijskoj bazi sastoji se od skupa operatera pomoću kojih se obrađuju podaci u stablima, operateri se sastoje od operacija lociranja korijena stabla, pomicanja na slijedeće stablo, te pomicanja na slijedeći zapis.

Iako današnja zastupljenost hijerarhijskih baza podataka nije velika, one imaju svoje prednosti, kao što su brzo spremanje i dohvaćanje podataka.

2.1.2 Mrežni model podataka

Na osnovu mrežnog modela 1971. godine nastao je prvi standard na području baza podataka. Iako sličan hijerarhijskom modelu, u mrežnom modelu podataka svako dijete može imati više roditelja. Time se ostvaruju relacije jedan naprema više, i više naprema više, a moguće je i uspostavljanje veze između različitih hijerarhija zapisa.



Slika 4: Mrežni model podataka

Izvor: http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf, str 6 (16.2.2019.)

Osnovni elementi mrežnog modela su zapisi, gdje jedan zapis može imati više roditelja, a svaki roditelj može imati više djece (zapisa). Ovakvom realizacijom relacija redundancija podataka se uvelike smanjuje.

U prednostima mrežnog modela podataka nalazimo brzinu pristupa podacima, dobro upravljanje integritetom i podatkovnu neovisnost, no nedostaci su još uvijek velika kompleksnost sustava i zahtjevna implementacija modela.

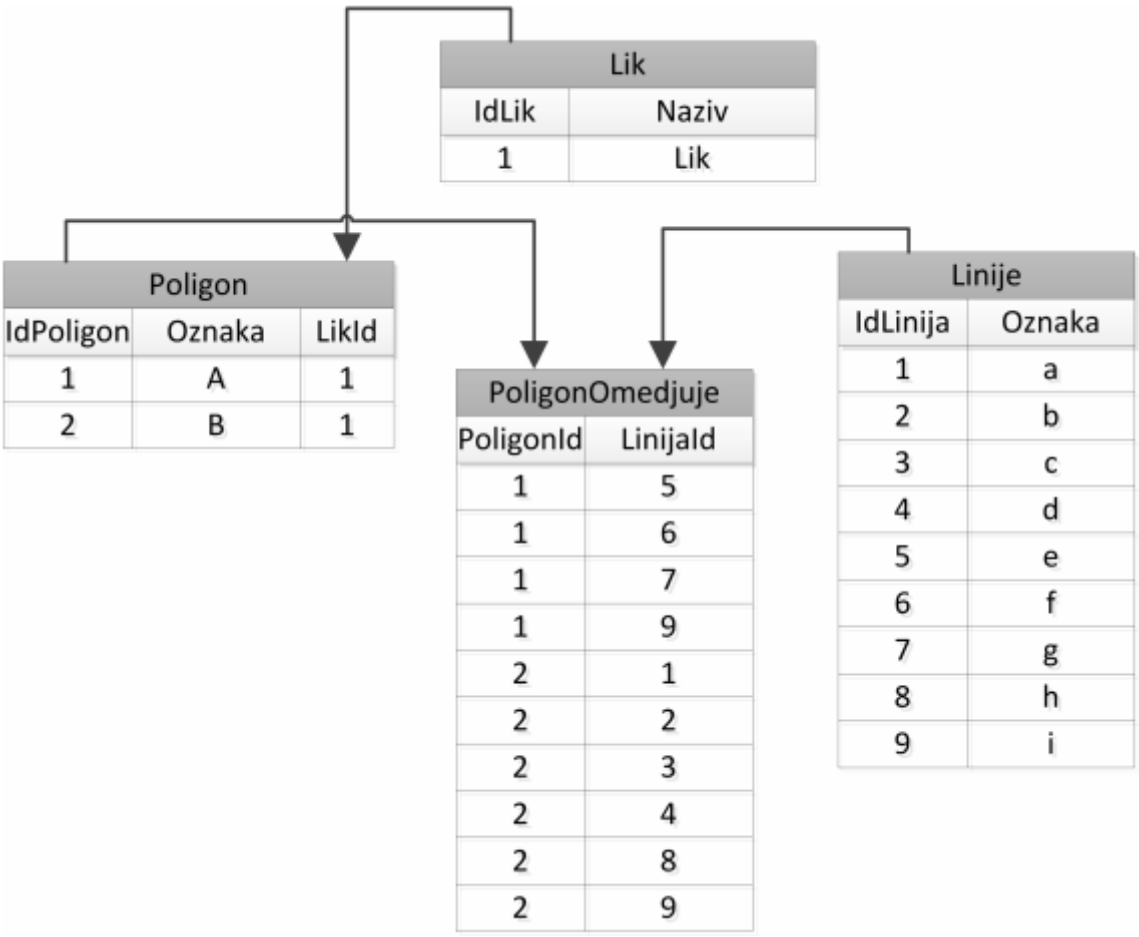
2.1.3 Relacijski model podataka

Otac relacijskog modela podataka je engleski matematičar Edgar Frank Codd. On je za vrijeme rada u IBM-u 1970. godine izdao članak „A relational model of data for large shared data banks“. Želja mu je bila implementirati relacijski model u IBM, no oni su ostali dosljedni u korištenju provjerenog, ali zastarjelog hijerarhijskog modela u sustavu IMS. Oracle je tako postala prva tvrtka sa implementiranim relacijskim modelom podataka, a i danas je sinonim za baze podataka. Codd-ov relacijski upitni jezik Alpha pritom nije prihvaćen, već je odabran nerelacijski jezik SEQUEL, koji je kasnije preimenovan u SQL radi podudaranja imena sa zaštićenim nazivom zrakoplovne kompanije.

Proizvođači su godinama krivo ili djelomično implementirali relacijski model podataka, prezentirajući ga kao relacijski. Codd-ov odgovor na to je objava dvanaest pravila koja sustav za upravljanje bazom podataka mora poštivati da bi ga se nazivalo relacijskim:

1. Predstavljanje informacija (engl. The infomation rule)
2. Pravilo pristupa (engl. The guaranteed access rule)
3. Tretiranje nepoznatih vrijednosti (engl. Systematic treatment of null values)
4. Dinamički online katalog (engl. Active online catalog based on the relational model)
5. Pravilo sveobuhvatnog jezika (engl. The comprehensive data sublanguage rule)
6. Pravilo pogleda (engl. The view updating rule)
7. Pravila ažuriranja skupova (engl. High-level insert, update, and delete)
8. Nezavisnost fizičkih podataka (engl. Physical data independence)
9. Nezavisnost logičkih podataka (engl. Logical data independence)
10. Nezavisnost integriteta podataka (engl. Integrity independance)
11. Distribuirana zavisnost (engl. Distribution independence)
12. Pravilo o nenarušavanju (engl. The nonsubversion rule)
13. Nulto pravilo (eng. Rule 0) – da bi sustav za upravljanje bazama podataka bio relacijski mora koristiti isključivo relacijske mogućnosti baze podataka kod upravljanja

Danas se sustav za upravljanje bazom podataka naziva relacijskim ako poštuje šest od trinaest navedenih pravila.



Lik	
IdLik	Naziv
1	Lik

Poligon		
IdPoligon	Oznaka	LikId
1	A	1
2	B	1

PoligonOmedjuje	
PoligonId	LinijaId
1	5
1	6
1	7
1	9
2	1
2	2
2	3
2	4
2	8
2	9

Linije	
IdLinija	Oznaka
1	a
2	b
3	c
4	d
5	e
6	f
7	g
8	h
9	i

Slika 5: Relacijski model baze podataka

Izvor: http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf, str 11 (16.2.2019.)

Podaci se u relacionom modelu spremaju u tablice, a pri tom se za svaku tablicu definira primarni ključ koji točno određuje n-torku relacije, odnosno red u tablici. Stupovi tablice su raspoređeni u vrijednosti atributa. Veza između tablica se ostvaruje fizičkim upisom primarnog ključa u konceptualno povezanu tablicu, te ga se tamo naziva stranim ključem.

Temelj relacionog modela podataka je matematička teorija skupova, te se tako i kao operacije nad podacima koriste matematičke operacije iz teorije o skupovima (oduzimanje, zbrajanje, presjek, unija te Kartezijev produkt).

Relacijski model podataka je aktualno najzastupljeniji; bez obzira što je brzina ovog modela najsporija, pokazao se najfleksibilniji i pogodan za široku primjenu.

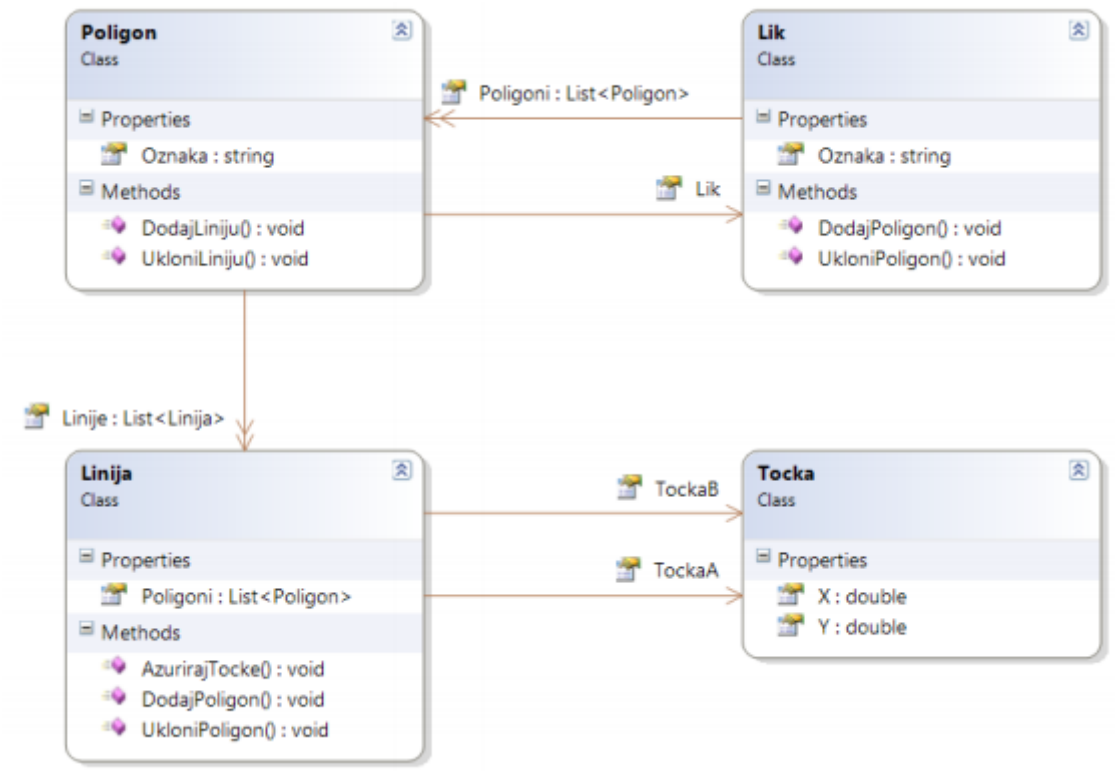
2.1.4 Objektno orijentiran model podataka

Ideja objektno orijentiranog modela podataka se pojavila u isto vrijeme kad i relacijski model podataka, prototipovi su se razvijali 80-tih godina, a komercijalni alati su izašli na tržište početkom 90-tih godina. Objektni model podataka podržava semantiku objekata koja je zastupljena u objektno orijentiranim programskim jezicima.

Objektno orijentirani model podataka pokazao se mnogo brži od relacijskog modela podataka za aplikacije koje zahtijevaju spremanje složenijih podataka sa mnogo relacija, kao što su npr. prostorni podaci, te u posljednje vrijeme osvaja velika područja primjene.

Osnovna svojstava modela potrebna da bi ga se definiralo kao objektno orijentirani model podataka:

1. Apstrakcija – pojednostavljenje kompleksnih objekata iz realnog svijeta tako da ostanu separirane bitne karakteristike i ponašanje pojedinačnog objekta
2. Enkapsulacija – implementacija koja se realizira željenim ponašanjem objekta, pritom se odvaja sučelje objekta od same implementacije ponašanja objekta
3. Modularnost – formiranje modularnih cjelina koje se mogu ponašati neovisno ili mogu komunicirati sa drugim modulima
4. Nasljeđivanje – definiranje objekta na osnovi drugog, već definiranog objekta, gdje se nasljeđuju sve metode i atributi i eventualno dodaju novi atributi i modificira ponašanje naslijeđenih metoda
5. Poliformizam – objekt može predstavljati više od jednog tipa oblika – ovo se smatra najvažnijim svojstvom objekno orijentiranih baza podataka



Slika 6: Objektne baze podataka

Izvor : http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf, str 8 (16.2.2019.)

Prilikom definiranja svakom pojedinačnom objektu se dodjeljuje jedinstveni identifikator GUID (engl. *Globally unique identefier*) na osnovu kojega sustav gradi relacije i preko kojega se vrši identifikacija objekata baze

Veliki nedostatak, s obzirom na aktualno stanje na tržištu baza podataka, objektno orijentiranih baza podataka je nekompatibilnost integracije sa relacijskim bazama podataka, te standardizacija upitnog jezika istih.

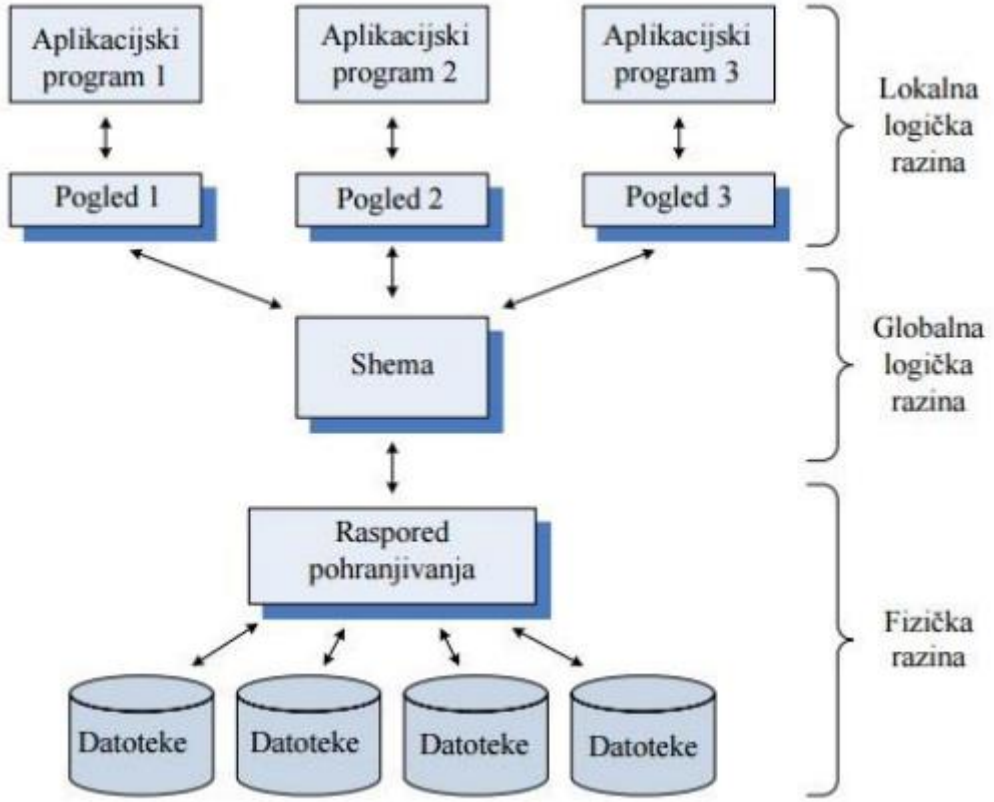
U ovom ćemo radu postaviti fokus na relacijske baze podataka, realizaciju i korištenje istih.

3. Arhitektura baze podataka

Pod arhitekturom baze podataka smatramo pogled na bazu podataka segmentiran u tri sloja, odnosno u tri razine apstrakcije:

1. Fizička razina

2. Globalna logička razina
3. Lokalna logička razina



Slika 7: Arhitektura baze podataka

Izvor: https://bib.irb.hr/datoteka/895660.1-tomislav-bobinac-dipl-rad.pdf, str. 3 (28.2.2019.)

Baze podataka predstavljaju višu razinu rada s podacima uspoređujući ih sa klasičnim programskim jezicima. Ovakav zaključak se nameće kada promotrimo ciljeve koje tehnologija baza podataka nastoji, i uvelike uspijeva ispuniti.[2]

1. Fizička nezavisnost podataka – razdvojena logička definicija baze od stvarne fizičke građe(prijepis podataka na druge diskove ne utječe na rad aplikacija)
2. Logička nezavisnost podataka – razdvojena globalna logička od lokalne logičke razine za jednu aplikaciju (uvođenje novog zapisa ili veze ne utječe na rad aplikacije)
3. Fleksibilnost pristupa podacima – korisnik može slobodno po vlastitom izboru koristiti podatke i stvarati veze među njima

[2] Ciljevi tehnologije baze podataka preuzeti iz R.Mangar, „Baza podataka“

4. Istovremeni pristup podacima – omogućeno istovremeno korištenje podacima većem broju korisnika
5. Zaštita od neovlaštenog korištenja – mogućnost ograničavanja korištenja baze podataka, odnosno regulacija ovlaštenja korisnika
6. Zadovoljavajuća brzina pristupa – obavljanje operacija s podacima dovoljnom brzinom za potrebe određene aplikacije
7. Mogućnost podešavanja i kontrole – sve veća baza podataka zahtijeva sve veće održavanje, odnosno korigiranje i prilagođavanje promjenama. Ove funkcije obavlja administrator baze podataka

„ Arhitektura baze podataka se sastoji od tri sloja: fizičke, globalne logičke razine i lokalne logičke razine.

Fizička razina se odnosi na fizički prikaz i raspored podataka na jedinicama vanjske memorije. Sama fizička razina se može podijeliti na više razina, od sasvim konkretnih staza i cilindara na disku, do pojmova datoteke i zapisa.

Globalna logička razina odnosi se na logičku strukturu cijele baze. Opis globalne jezične definicije naziva se shema. Shema je tekst ili dijagram koji definira logičku strukturu baze i u skladu je sa zadanim modelom.

Lokalna logička razina odnosi se na logičku predodžbu o dijelu baze kojega rabi pojedina aplikacija. To je aspekt sa strane korisnika ili programera. Opis jedne lokalne jezične definicije naziva se pogled. To je tekst ili dijagram kojim se imenuju i definiraju svi lokalni tipovi podataka i veze među tim tipovima, u skladu sa modelom.

Fizička neovisnost podataka se postiže time što se pravi razlika između fizičke i globalne logičke razine, dok se logička neovisnost postiže razlikovanjem lokalne logičke razine i globalne logičke razine. Na taj način, troslojna arhitektura omogućuje ispunjavanje dvaju najvažnijih ciljeva koje se nastoje postići uporabom baze podataka.“ (Manger, 2011.).

4. Planiranje izrade i životni ciklus baze podataka

Projekt uvođenja novih softverskih proizvoda u poslovanje je složeni proces i provodi se uz primjenu pogodnih metoda i alata, te redovito zahtijeva timski angažman kako projektanta sa softverske strane tako i korisnika poslovnih procesa u koje se sustav implementira.

Općenito se takav razvojni ciklus proizvoda u osnovi dijeli na pet faza:

- Utvrđivanje i analiza zahtjeva
- Modeliranje podataka
- Implementacija
- Testiranje
- Održavanje

Kompleksnost i veličina svake od ovih faza uvjetovana je kompleksnošću i veličinom organizacijskog sustava na koje se odnose.

Složene sustave možemo jasno definirati samo ako ih raščlanimo i podijelimo u više manjih podsustava, koje onda rješavamo jedan po jedan podsustav. Za ovaj postupak navesti ćemo dvije metode – procesno i entitetno razlaganje.

„Procesno rastavljanje sustava u podsustave je traženje grupa poslova sustava koji su relativno međusobno neovisni i predstavljaju logičke cjeline.

Entitetno rastavljanje sustava u podsustave je traženje relativno jednostavnih dijelova (objekata) koji su relativno jednostavno povezani.“[3]

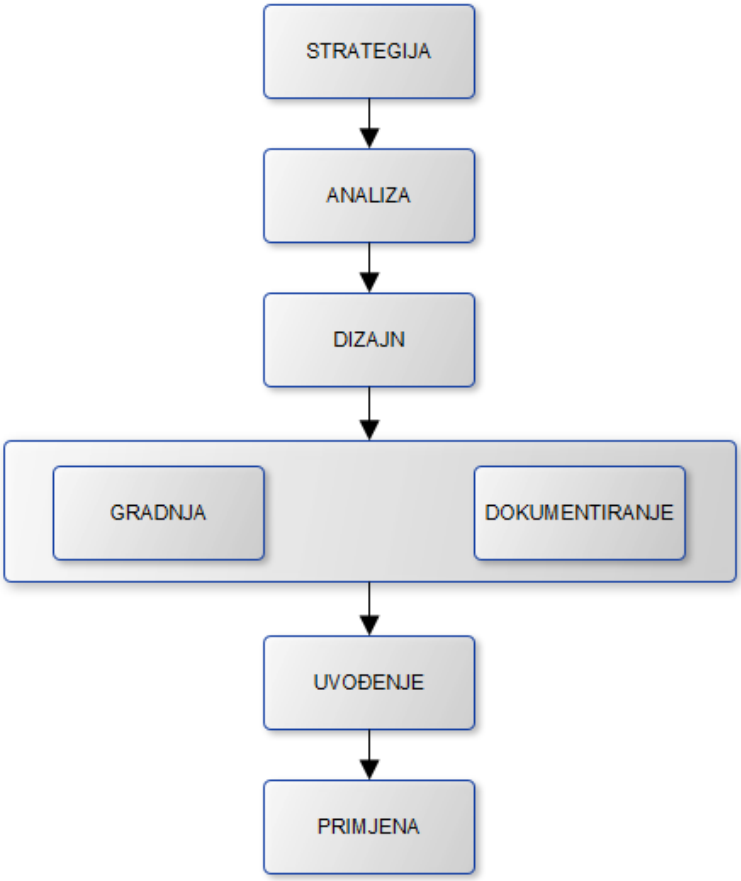
Kada složeni proces rastavimo na grupe potprocesa koje simultano ili uzastopno pridonose realizaciji nekog lokalnog cilja nazivamo ih fazama procesa. Takvim koracima analiziramo i rješavamo jedan po jedan podsustav izvodeći jednu po jednu fazu dok nas posljednja aktivnost ne dovede do konačnog cilja.

Govoreći o informatičkom inženjerstvu, ne postoji jedinstvena metoda za projektiranje svih vrsta informacijskih sustava, nego postoji više vrsta odgovarajućih grupa metoda. Upoznat

[3] M. Pavlić, „Razvoj informacijskih sustava“, 2011.

ćemo se sa dvije metode koje su nastale u želji da se prevladaju problemi metodike klasičnog životnog ciklusa podataka.

CASE*Method – autor ove metodike je R.Baker, a standardizirana je od strane korporacije ORACLE u skladu sa njihovim CASE alatima.[4]



Slika 8: Faze razvoja prema metodici CASE*Method

Izvor: Autor

CASE*Method metodika zastupa područje modeliranja procesa i područje modeliranja podataka.

Standardizirane metode na području modeliranja procesa su[5]:

- Hijerarhijski prikaz poslovnih funkcija
- Prikaz ovisnosti između funkcija

[4] CASE alat (engl. Computer-Aided Software Engineering) – softverski proizvod koji omogućuje automatizaciju procesa izrade sofvera; u sebi sadrži podršku za dizajn, programiranje i testiranje softverskog proizvoda
[5] Prema M.Pavlić,"Razvoj informacijskih sustava", 2011.

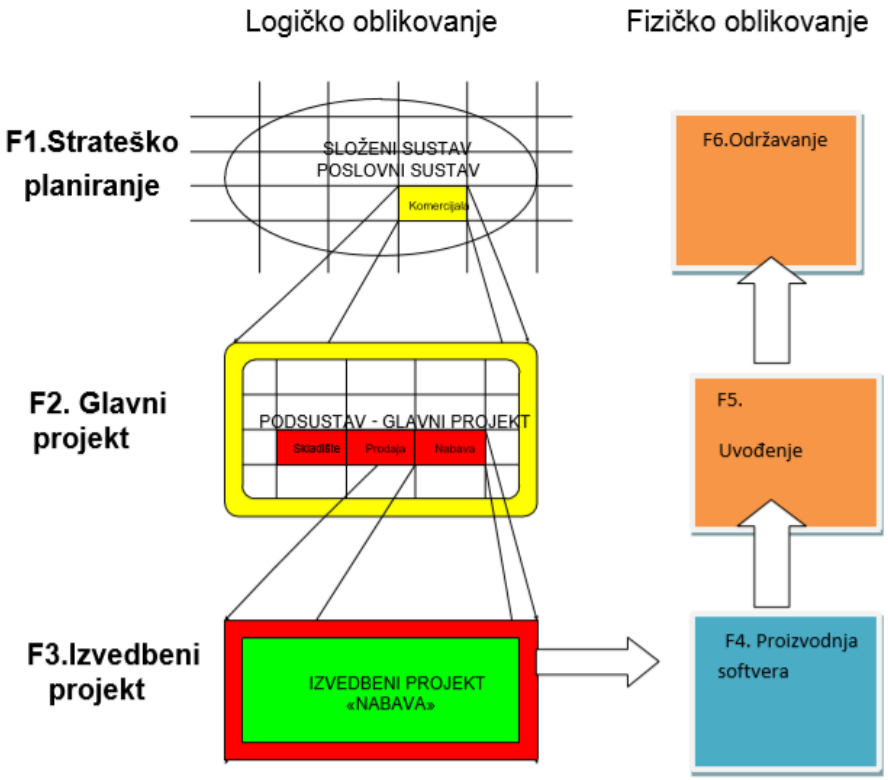
- Povezivanje funkcija s organizacijskim jedinicama, mjestom izvođenja procesa, ulogom funkcija, modelom podataka
- Definicija detaljne logike funkcija
- Modeliranje procesa i crtanje dijagram toka podataka
- Hijerarhijski prikaz procesa
- Modeliranje u stvarnom vremenu
- Strukturna karta
- Prevođenje dijagrama podataka u strukturnu kartu

Standardizirane metode na području modeliranja podataka su:

- Metoda entiteti-veze
- Relacijska metoda
- Entiteti-veze
- Metoda mrežne organizacije podataka
- Metoda hijerarhijske organizacije podataka

Za svaku navedenu metodu propisan je skup obrazaca, ali i razvijena programska podrška. Temelj ove metodike je modeliranje podataka pomoću jedne varijante metode entiteti-veze, te se dobiveni model jednostavno prevodi u relacijski ili neki drugi semantički bogat model podataka.

MIRIS metodika – (MetodologIja za Razvoj Informacijskog Sustava) prva hrvatska metodika. U MIRIS su uključene metode entiteti-veze, SAS metoda druge uključene u većini CASE alata. Ova specijalizirana metodologija propisuje faze razvoja i aktivnosti pojedine faze do potrebne razine detalja informacijskih sustava, pri čemu se cjelokupni razvojni napor sastoji od logičkog (projektiranje informacijskog sustava) i fizičkog oblikovanja (izgradnja informacijskog sustava).



Slika 9: Raščlanjivanje sustava i faze specijalizirane metodologije MIRIS

Izvor: M.Pavlić, „Oblikovanje baza podataka“, 2011., str.24 (6.3.2019.)

Logičko oblikovanje metodikom MIRIS prikazano je u fazama 1, 2 i 3 u tablicama koje slijede.

Sveobuhvatni posao započinje izradom strateškog plana.

Faza 1: STRATEŠKO PLANIRANJE INFORMACIJSKOGA SUSTAVA (SP)
1.1 Analiza: Definiranje i obuka tima, dekompozicija procesa, popis dokumentacije i kretanje kroz sustav
1.2 Podsustavi: Određivanje podsustava i veza
1.3 Prioriteti: Određivanje prioriteta
1.4 Resursi: Definiranje cjelovite infrastrukture
1.5 Plan: Planiranje glavnih projekata i aktivnosti

Tablica 2: Strateško planiranje

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 24 (11.3.2019.)

U fazi Glavnog projekta analizira se poslovanje i kreira model procesa.

Faza 2: GLAVNI PROJEKT (GP)
2.1 PZ: Izrada projektnoga zadatka
2.2 DTP: Intervjuiranje, raščlanjivanje i modeliranje procesa (DTP)
2.3 Procesi GP: Analiza procesa, problema i prijedloga poboljšanja
2.4 Podaci GP: Opisivanje podataka
2.5 Plan GP: Planiranje izvedbenih projekata
2.6 Resursi GP: Definiranje modela resursa glavnoga projekta

Tablica 3: Glavni projekt

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 25 (11.3.2019.)

U fazi Izvedbenog projekta modeliraju se podaci i definira logička arhitektura programskoga proizvoda.

Faza 3: IZVEDBENI PROJEKT (IP)
3.1 DEV: Intervjuiranje, apstrakcija i modeliranje podataka (EV)
3.2 Prevođenje: Prevođenje modela podataka u shemu BP (RM)
3.3 Arhitektura IP: Definiranje arhitekture programskoga proizvoda (APP)
3.4 Operacije IP: Projektiranje operacija nad shemom BP

Tablica 4: Izvedbeni projekt

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 25 (11.3.2019.)

Slijedeće tri tablice odnose se na fizičko oblikovanje metodikom MIRIS.

Cilj četvrte faze je kreiranje potrebnog programskog proizvoda, ili potpuno novog, ili prilagođenog postojećeg koji zadovoljava potrebe definirane u prve tri faze.

Faza 4: PROIZVODNJA SOFTVERA (PS)
4.1: PLANIRANJE PROIZVODNJE
• Planiranje aktivnosti proizvodnje SW
• Određivanje izvršitelja za pojedine zadatke i određivanje rokova
• Određivanje i kreiranje produkcijske, testne i razvojne okoline
4.2: OBLIKOVANJE BAZA PODATAKA
• Prevođenje logičkoga modela podataka u fizički model sheme baze podataka
• Kreiranje razvojne okoline za svakoga pojedinog programera
• Punjenje sheme baze podataka u razvojnoj okolini iz postojeće produkcijske BP
• Dodavanje novih koncepata iz modela podataka u razvojnu shemu BP (tip entiteta i dr.)
• Kreacija razvojne baze podataka
• Inicijalno punjenje testne baze podataka
4.3: RAZVOJ PROGRAMSKOGA PROIZVODA
• Izrada glavnog izbornika (aplikacijskoga stabla) ili dorada već postojećega
• Izrada ekrana za pregled redaka svake tablice po jednom ili više ključeva
• Izrada ekrana za operacije nad jednim retkom tablice (unos, izmjena, brisanje i pregled)
• Izrada programskih modula različitih vrsta i namjena: obračuna, procedura, funkcija kontrola, *look-upova* nad tablicama (s prvim testiranjem modula)
• Izrada izvještaja (s prvim testiranjem modula)
4.4: TESTIRANJE U TESTNOJ OKOLINI
• Prijenos razvijenih programskih modula u testno okruženje
• Spajanje novih modula s postojećim
• *Back-up* verzija softvera
• Testiranje prototipa softvera nad testnom bazom podataka
• Ažuriranje planova proizvodnje softvera
• Izvođenje prema potrebi aktivnosti iz ranijih skupina aktivnosti i ponovno testiranje
4.5: TESTIRANJE I ISPRAVLJANJE U RADNOJ OKOLINI
• Prijenos razvijenih programskih modula u radno okruženje
• Spajanje novih modula s postojećima
• *Back-up* verzija softvera
• Punjenje baze podataka
• Testiranje prototipa softvera nad produkcijskom bazom podataka koje obavlja programer
• Ažuriranje planova proizvodnje softvera
• Izvođenje prema potrebi aktivnosti iz ranijih skupina aktivnosti i ponovno testiranje
4.6: KORISNIČKO TESTIRANJE
• Prezentacija softvera korisniku
• Testiranje od strane korisnika
• Izrada popisa primjedbi korisnika
• Ažuriranje planova proizvodnje softvera
• Izvođenje prema potrebi aktivnosti iz ranijih skupina aktivnosti i ponovno testiranje
• Izrada zapisnika o testiranju i prihvaćanju faze uvođenja

Tablica 5: Proizvodnja softvera

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 25, 26 (11.3.2019.)

Gotovi proizvod se tada uvodi u poslovni sustav po koracima faze 5.

Faza 5: UVOĐENJE (UVO)
5.1 Instalacija gotovoga softvera na produkcijsko okruženje (kod korisnika)
5.2 Izrada uputa
5.3 Prezentacija gotovoga softvera
5.4 Obuka
5.5 Završne konverzije
5.6 Završno testiranje
5.7 Početak primjene nove aplikacije
5.8 Uspostava novoga sustava i potpisivanje primopredajnoga zapisnika

Tablica 6: Uvođenje

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 26 (11.3.2019.)

U fazi 6 se nadgleda rad sustava i povratne informacije korisnika, te se po potrebi održavaju dijelovi sustava kako bi zadovoljili potrebe korisnika.

Faza 6: PRIMJENA I ODRŽAVANJE (ODR)
6.1 Podešavanje novoga aplikacijskoga sustava
6.2 Izvješće o procjeni novoga projekta
6.3 Raspodjela odgovornosti korisnika i programera
6.4 Korištenje aplikacijskoga sustava
6.5 Korisničko postavljanje zahtjeva za izmjenama

Tablica 7: Primjena i održavanje

Izvor: https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 26 (11.3.2019.)

Rezultat glavnog projekta je projektna dokumentacija glavnog projekta. U ovoj se fazi po metodici MIRIS primjenjuje SAS (strukturna analiza sustava) za modeliranje procesa i ekspertne metode za modeliranje resursa.

Izvedbeni projekt je potprojekt i dio glavnog projekta, gdje se detaljno pristupa projektiranju dijela sustava. Rezultat izvedbenog projekta je model koji detaljno opisuje dio sustava i dobra je osnova za fizički dizajn. Metoda entiteti-veze je bazična metoda kod detaljnog projektiranja.

Proučivši korake primijenjene kod ove dvije metodike jasnije nam je predočiti si faze razvojnog ciklusa baze podataka.

4.1 Utvrđivanje i analiza zahtjeva

Za potrebe utvrđivanja zahtjeva potrebno je proučiti tokove informacija u organizaciji za koju izrađujemo informacijski sustav. Potrebno je sagledati sve dokumente koji su u opticaju i definirati podatke koji se pohranjuju i veze među njima, jasno odrediti radne procese, sve to obavezno kroz razgovor s korisnicima; te proučiti postojeći softver ako je implementiran.

Što je veća i kompleksnija organizacija, više je različitih grupa korisnika, te se analogno tome neminovno pojavljuju različita tumačenja značenja i svrhe pojedinih podataka i različiti načini njihove upotrebe. Analiza zahtjeva ima za zadatak pomiriti te razlike, eliminirati redundanciju i nekonzistentnost prvenstveno na način da prepozna sinonime i homonime i tim putem uskladi terminologiju.

Analizom zahtjeva mora se obuhvatiti i analiza transakcija, odnosno postupaka i operacija koje će se obavljati s podacima pri korištenju baze podataka jer u pravilu utječu na sadržaj i konačni oblik baze podataka. Važno je odrediti učestalost i opseg transakcija i zahtjeve na performanse sustava prilikom korištenja.

Definiranjem analize zahtjeva sastavljamo dokument koji je obično pisan neformalno u prirodnom jeziku. Rijetko se ovaj dokument odnosi samo na podatke, već najčešće definira i važne transakcije s podacima, nekad i cijele aplikacije. Taj dokument nazivamo specifikacija potreba i on je osnova za daljni rad pri izradi baze podataka.

4.2 Modeliranje podataka

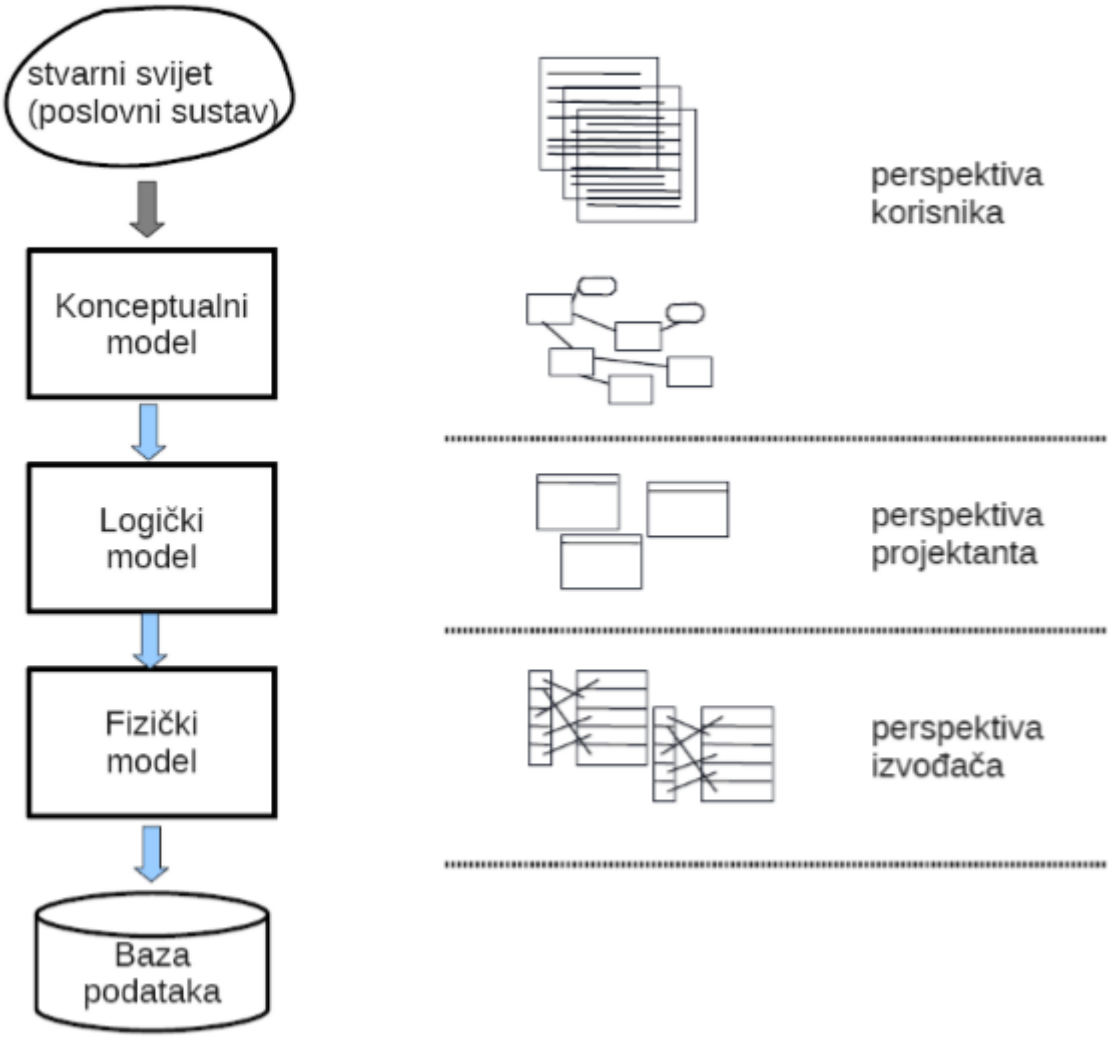
Modeliranje podataka se odnosi na oblikovanje građe baze podataka u skladu sa dobivenom specifikacijom potreba. U analizi zahtjeva se otprilike određuje koje će vrste podataka baza sadržavati i kroz koje će transakcije prolaziti podaci, dok u modeliranju podataka određujemo pogodan način kako da se podaci grupiraju, strukturiraju i međusobno povežu.

Rezultat faze modeliranja podataka mora biti kompletna shema baze podataka, izrađena po pravilima korištenog modela podataka i zapisana na način koji se realizira u odabranom sustavu za upravljanje bazama podataka.

Modeliranje podataka je složen proces u kojem dolazi do izražaja kreativnost u razumijevanju i logičkom povezivanju i definiranju podataka.

Modeliranje podataka u pravilu dijelimo na tri faze:

- Konceptualno modeliranje podataka
- Logičko modeliranje podataka
- Fizičko modeliranje podataka



Slika 10: Faze modeliranja podataka

Izvor: http://uni-mo.sve-mo.ba/~goran/nastava/BAZE_Nastava.pdf, str. 14 (15.05.2019.)

Svaku od navedenih faza i metode korištene u njima pobliže ćemo obraditi u slijedećim poglavljima.

4.2.1 Konceptualno modeliranje podataka

„Konceptualno modeliranje podataka polazi od specifikacija informacijskih zahtjeva, koji čine zahtjeve na strukturu podataka i zahtjeve za korištenje podataka, a rezultira izrađenim konceptualnim modelom podataka.“[6]

Kvalitetno izrađeni konceptualni model podataka ima grupirane podatke koji opisuju jednu činjenicu i neovisni su od podataka druge činjenice.

Postoji više metoda i postupaka kojima se konstruira konceptualni model podataka, mi ćemo opisati model entiteti-veze, i postupke koji se koriste u izradi, te izradu Chenovog dijagrama.

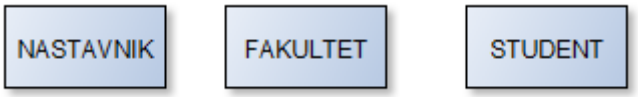
Model entitet-veza je grafički model kojim se stvarni ili zamišljeni svijet prikazuje pomoću odnosa tri osnovna elementa:

- Entiteti
- Veze
- Atributi

Entitet predstavlja nešto što postoji u stvarnom svijetu i posjeduje osobine koje ga opisuju i po kojim se razlikuje od svoje okoline. Entitet je nešto o čemu želimo spremati podatke, može biti objekt ili biće, događaj ili pojava. Entitet je nešto što se može jednoznačno imenovati, a pobliže se opisuje atributima.

Postoje mnoge definicije entiteta, ali nijednu ne možemo smatrati potpunom i konačnom, no sve one nam iz raznih pogleda pobliže definiraju pojam entiteta.

Entiteti se u grafičkom prikazu predstavljaju pravokutnikom proizvoljnih dimenzija.



Slika 11: Grafički prikaz entiteta

Izvor: Autor

[6] Mladen Varga, „Konceptualno, logičko i fizičko modeliranje podataka“, DRIP Zagreb, 1994.,str. 43

Postoje dva tipa entiteta:

- Jaki entitet: tip entiteta čija egzistencija nije vezana za postojanje nekog drugog tipa entiteta
- Slabi entitet: tip entiteta čija egzistencija ovisi o postojanju nekog drugog tipa entiteta; onaj tip entiteta koji ne može postojati u bazi podataka ako neki drugi određeni tip entiteta ne postoji u bazi

Veza predstavlja odnos koji postoji između entiteta. Veza je koncept koji spaja koncepte entiteta a sama nema sadržaj strukturnog tipa kao entitet.

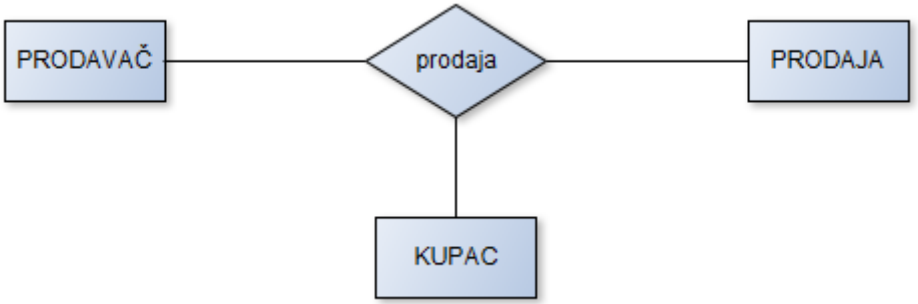
Stupanj veze predstavlja broj entiteta u vezi, tako razlikujemo:

- Binarna veza: je veza dva entiteta

Slika 12: Primjer binarne veze

Izvor: Autor

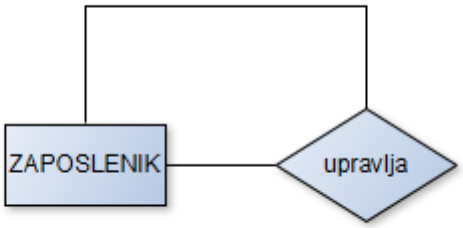
- Ternarna veza: je veza tri entiteta



Slika 13: Primjer ternarne veze

Izvor: Autor

- Unarna veza: je veza gdje isti entitet više puta egzistira u različitim ulogama



Slika 14: Primjer unarne veze

Izvor: Autor

Kardinalnost veze je pojam kojim obuhvaćamo svojstva funkcionalnosti i obaveznosti članstva veze. Razlikujemo više vrsta veze:

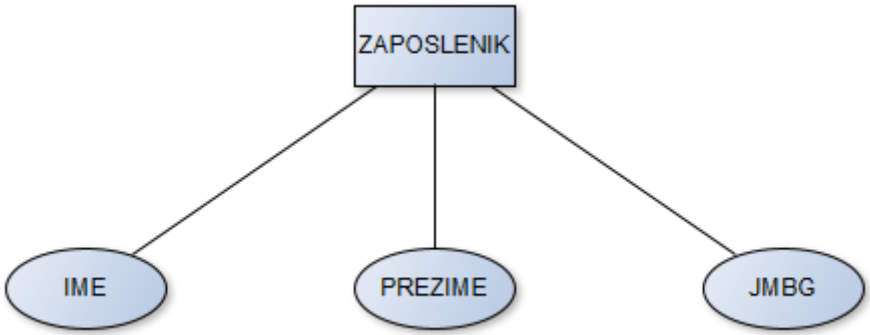
- Jedan naprema jedan (1:1) – jedan primjerak entiteta 1 može biti povezan najviše s jednim primjerkom entiteta 2. Isto tako jedan primjerak entiteta 2 može biti povezan najviše sa jednim primjerkom entiteta 1
- Jedan naprema više (1:M) – jedan primjerak entiteta 1 može biti povezan s više primjeraka entiteta 2, no jedan primjerak entiteta 2 može biti povezan najviše s jednim primjerkom entiteta 1
- Više naprema jedan (M:1) – jedan primjerak entiteta 1 može biti povezan najviše s jednim primjerkom entiteta 2, jedan primjerak entiteta 2 može biti povezan s više primjeraka entiteta 1
- Više naprema više (M:M) – jedan primjerak entiteta 1 može biti povezan s više primjeraka entiteta 2, i jedan primjerak entiteta 2 može biti povezan s više primjeraka entiteta 1

Kardinalnost veze se ne izražava točnim brojem već kao interval u obliku donje i gornje granice, i to oznakama 0, 1 i M. Najčešće kardinalnosti su:

- 0, 1 – jedan primjerak entiteta 1 može biti povezan s nijednim ili najviše s jednim primjerkom entiteta 2
- 1, 1 – jedan primjerak entiteta 1 mora biti povezan s točno jednim primjerkom entiteta 2
- 0, M – jedan primjerak entiteta 1 može biti povezan s nijednim, s jednim ili s više primjeraka entiteta 2

- 1, M – jedan primjerak entiteta 1 mora biti povezan s najmanje jednim, no može i s više primjeraka entiteta 2

Atributi su imenovane karakteristike nekog tipa entiteta koje su nam značajne u izradi baze podataka. Oni pobliže opisuju i definiraju značajke entiteta. Atribut se grafički prikazuje u ovalnom obliku.



Slika 15: Primjer grafičkog prikaza atributa

Izvor: Autor

Prilikom izrade E-V dijagrama, nakon određivanja entiteta i definiranja veza među njima, moramo odrediti koji ćemo atribut proglasiti primarnim ključem tipa entiteta. Takav atribut mora zadovoljavati uvjete jedinstvenosti i neredundantnosti, a ako takav atribut ne postoji morati ćemo ga stvoriti. Najučinkovitije je uvesti novi atribut koji numeracijom jedinstveno definira entitet.

4.2.2 Relacijski model podataka

Nakon što imamo zadovoljavajući grafički prikaz u obliku modela entiteti-veze, vrijeme je da od njega stvorimo relacijski model podataka, koji će predstavljati novu bazu podataka.

U relacijskom modelu se zahtijeva da baza podataka bude sastavljena od skupa pravokutnih tabela koje nazivamo relacije. U takvim tabelama naslov tabele predstavlja tip entiteta (moguće i vezu), dok stupci u tabeli predstavljaju atribute. Vrijednosti jednog atributa su uvijek podaci istog tipa. Jedan red relacije obično predstavlja jedan primjerak entiteta, ili bilježi vezu između dva ili više primjeraka entiteta (spojne tablice ili rješavanje veze više naprema više) te ga nazivamo n-torka.

U nekim se sustavima za upravljanje bazama podataka umjesto matematičkih termina relacija, n-torka, atribut koriste termini tablica, redak, stupac ili termini programskih jezika datoteka, zapis, polje.

TERMINOLOGIJA		
RELACIJSKA	TABLIČNA	KLASIČNE OBRADE
domena	skup vrijednosti	tip
atribut	stupac	polje
n-torka	redak	zapis (record)
primarni ključ	identificirajući stupci	ključ
(normalizirana) relacija	tablica	datoteka
stupanj relacije	broj stupaca	broj polja u zapisu
kardinalnost relacije	broj redaka	broj zapisa u datoteci

Tablica 8: Terminologija relacijske, tablične i klasične obrade podataka

Izvor: Autor

Veze definirane u modelu entiteti-veze se realiziraju u relacijskom modelu pomoću primarnih i stranih ključeva. Atribut ili skup atributa koji smo odredili za primarni ključ jedne tablice postaviti ćemo u povezanu tablicu kao strani ključ, tako ćemo se pomoću vrijednosti tog stranog ključa referirati na vrijednost n-torke prve relacije, odnosno tablice. Pritom se definira veza između relacija. Za vezu 1:1 strani ključ u relaciji je istovremeno primarni ključ te relacije, za vezu 1:M druga tablica ima vlastiti primarni ključ i upisani strani ključ, za vezu M:M potrebno je stvoriti tablicu veze kao entitet, i strane ključeve od dvije povezane tablice proglasiti primarnim ključevima te tablice.

Pri izradi relacijskog modela pridržavamo se općih ograničenja koja važe za bilo koji relacijski model:

- Pravilo integriteta entiteta – niti jedan atribut koji je primarni ključ ili dio primarnog ključa neke relacije ne smije imati null vrijednost, odnosno ostati neodređen
- Pravilo referencijalnog integriteta – ako bazna relacija sadrži strani ključ koji ovu relaciju povezuje sa drugom relacijom preko njenog primarnog ključa, onda svaka vrijednost stranog ključa mora odgovarati nekoj vrijednosti primarnog ključa ili imati vrijednost null

U konstruiranju relacijskog modela podataka provodi se proces normalizacije. Normalizaciju možemo opisati kao proces daljnjeg dotjerivanja relacijske sheme kroz unaprijed definirane normalne forme do zadovoljavajuće relacijske sheme u kojoj je logička redundancija

minimalna. Postupkom normalizacije se uklanjaju anomalije pri izvođenju operacija kao što su dodavanje n-torke, izbacivanje n-torke i anomalije pri ažuriranju.

Postoji sedam normalnih formi no za većinu praktičnih primjera dovoljno je relacije normalizirati do treće normalne forme.

1. Prva normalna forma – relacija je u prvoj normalnoj formi ako su sve vrijednosti njenih atributa atomske

Primjer:

STUDENT

BR_INDEX	IME	SEMESTAR	STUDIJ	VOD_STUDIJA	ŠIF_KOLEGIJA	NAZ_KOLEGIJA	OCJENA
21	Mate	4	Računarstvo	Frane	123	Baze podataka	2
21	Mate	4	Računarstvo	Frane	124	Matematika	4
21	Mate	4	Računarstvo	Frane	267	Fizika	5
77	Ana	7	Elektronika	Jure	678	Polja i valovi	4
77	Ana	7	Elektronika	Jure	589	Teorija sustava	1
77	Ana	7	Elektronika	Jure	245	Modeliranje	4
77	Ana	7	Elektronika	Jure	567	Elektronički sklopovi	2
36	Pero	6	Elektronika	Jure	678	Polja i valovi	2
36	Pero	6	Elektronika	Jure	245	Modeliranje	3

Slika 16: Primjer tablice STUDENT

Izvor:
https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 65 (05.05.2019.)

Na primjeru tablice STUDENT vidimo da ovakva forma uzrokuje gomilanje podataka, tj. redundanciju. Ovu relaciju zato razdvajamo na dvije relacije.

STUDENT1

BR_INDEX	IME	SEMESTAR	STUDIJ	VOD_STUDIJA
21	Mate	4	Računarstvo	Frane
77	Ana	7	Elektronika	Jure
36	Pero	6	Elektronika	Jure
...				

Slika 17: Primjer tablice STUDENT1

Izvor:
https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 65 (05.05.2019.)

PRIJAVA

BR_INDEX	ŠIF_KOLEGIJA	NAZ_KOLEGIJA	OCJENA
21	123	Baze podataka	2
21	124	Matematika	4
21	267	Fizika	5
77	678	Polja i valovi	4
77	589	Teorija sustava	1
77	245	Modeliranje	4
77	567	Elektronički sklopovi	2
36	678	Polja i valovi	2
36	245	Modeliranje	3

Slika 18: Primjer tablice PRIJAVA

Izvor: https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 66 (05.05.2019.)

U procesu svođenja relacije na prvu normalnu formu moramo uzeti u obzir da rastavljanje jedne relacije na dvije ne uzrokuje gubitak podataka ili stvaranje novih podataka koji ne postoje u prvobitnoj relaciji.

2. Druga normalna forma – ako je relacija u prvoj normalnoj formi i svi njeni neključni atributi potpuno i funkcionalno zavise od primarnog ključa

Primjer:

U relaciji PRIJAVA očita je redundancija kod atributa NAZ_KOLEGIJA. Kod ovakve realizacije relacije pojavljuju se anomalije pri korištenju podataka. Podaci o novom kolegiju mogu se dodati u bazu samo ako ga je neki student položio; izbacivanjem n-torke studenta koji je jedini položio kolegij gubimo podatke o tom kolegiju; mijenjanje naziva kolegija potrebno je učiniti za sve studente koji su položili kolegij pojedinačno. Rješenje je i ovdje rastavljanje relacije na dvije nove relacije.

PRIJAVA1

	BR_INDEX	ŠIF_KOLEGIJA	OCJENA
	21	123	2
	21	124	4
	21	267	5
	77	678	4
	77	589	1
	77	245	4
	77	567	2
	36	678	2
	36	245	3
	...		

Slika 19: Primjer tablice PRIJAVA1

Izvor: https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 69 (05.05.2019.)

KOLEGIJ

	ŠIF_KOLEGIJA	NAZ_KOLEGIJA
	123	Baze podataka
	124	Matematika
	267	Fizika
	678	Polja i valovi
	589	Teorija sustava
	245	Modeliranje
	567	Elektronički sklopovi
	...	

Slika 20: Primjer tablice KOLEGIJ

Izvor: https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 69 (05.05.2019.)

3. Treća normalna forma – ako i samo ako je relacija u drugoj normalnoj formi i ako u njoj ne postoji tranzitivna funkcionalna zavisnost atributa o primarnom ključu.

Primjer:

Pogledamo li tablicu STUDENT1 uočavamo tranzitivnu zavisnost atributa VOD_STUDIJA o primarnom ključu BR_INDEKSA. Da bi smanjili redundanciju podataka i unaprijedili efikasnost izmjene podataka ovu relaciju razdvajamo na dvije.

STUDENT2

	BR_INDEX	IME	SEMESTAR	STUDIJ
	21	Mate	4	Računarstvo
	77	Ana	7	Elektronika
	36	Pero	6	Elektronika
	...			

Slika 21: Primjer tablice STUDENT2

Izvor: https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 70 (05.05.2019.)

STUDIJ

	STUDIJ	VOD_STUDIJA
	Računarstvo	Frane
	Elektronika	Jure
	...	

Slika 22: Primjer tablice STUDIJ

Izvor: https://www.racunarstvo550.xyz/Naslovnica/3.%20semestar/Baze%20podataka/Predavanja/Baze%20podataka%20skripta.pdf, str. 70 (05.07.2019.)

4.2.3 Fizičko modeliranje podataka

Glavni rezultat ove faze u modeliranju podataka je stvaranje fizičke sheme, odnosno opis njene fizičke građe. Fizička građa baze podataka se stvara od datoteka i indeksa pohranjenih u vanjskoj memoriji računala, odnosno disku.

Fizička shema je tekst kojeg stvaraju naredbe sastavljene u SQL-u ili nekom drugom programskom jeziku koji komunicira sa sustavom za upravljanje bazom podataka. Pri upisu SQL koda precizno definiramo sve podatke relacijske sheme, uključujući ograničenja i određene veličine podataka.

SQL je jezik u početku razvijen kao upitni jezik pomoću kojega odrađujemo naredbe koje traže neki podatak u bazi podataka. Postepeno razvio se u jezik kojim možemo ne samo pretraživati, već i kreirati i mijenjati podatke i metapodatke – strukture i ograničenja. SQL nazivamo deklarativnim jezikom, odnosno od korisnika zahtijeva specifikaciju onoga što želi, ali ne i specifikaciju procedure postizanja rezultata.

Elementi od kojih se sastoje SQL naredbe:

- Klauzule – ključne riječi koje imaju točno određeno značenje u komandi (preuzete iz engleskog jezika)
- Izrazi – kombinacija identifikatora, operatera i konstanti koji daju jednu vrijednost određenog tipa
- Operatori – oznake koje predstavljaju operaciju nad operandima u izrazu
- Identifikatori – oznake atributa, tablica, varijabli i ostalih objekata
- Konstante – konstantne vrijednosti bilo kojeg tipa

Sintaksa SQL jezika određena je standardom, no u praksi se pojavljuju manja odstupanja u primjerima implementacije.

S obzirom na namjenu u SQL-u razlikujemo:

- DDL – komande za definiranje i manipulaciju podacima (*data definition language*)
- DML – komande za manipulaciju podacima u bazi podataka (*data manipulation language*)
- DCL – komande za upravljanje korisnicima, grupama korisnika, odnosno njihovim pravima i ovlaštenjima (*data control language*)

Za realizaciju fizičke građe baze podataka koristimo DDL naredbe. Ove naredbe mogu se odnositi na akcije CREATE (stvaranje), ALTER (promjena), DROP (brisanje). Ovim naredbama oblikujemo tablice i odnose među njima. Pri korištenju naredbi CREATE i ALTER važna nam je u kreiranju CONSTRAINT klauzula kojom definiramo ograničenja.

DML naredbe nam omogućuju korištenje, i oblikovanje podataka u bazi prema potrebama korisnika baze. Najčešće korištene DML komande su slijedeće

- SELECT – jedna od važnijih i opširnijih komandi. Obavlja operacije selekcije i projekcije.
- UNION – naredba kojom kombiniramo više pogleda, tablica u jednu
- UPDATE – naredba za promjenu vrijednosti atributa n-torke za određenu tablicu na temelju zadanih kriterija
- DELETE – naredba kojom nepovratno brišemo jednu ili više n-torki iz jedne ili više tablica određujući ih klauzulom FROM i kriterijem WHERE

- INSERT INTO – naredba kojom se dodaju nove n-torke u odabranu tablicu, sa navođenjem vrijednosti atributa prema zadanoj sintaksi
- SELECT INTO – naredba specifična za Access, kreira novu tablicu u bazi te je popunjava vrijednostima iz druge tablice, pritom ne prenoseći ograničenja postojeće tablice

SQL je glavni jezik za korištenje pri radu sa sustavima za upravljanje bazama podataka. Postojanje aplikacijskih formi za unos vrijednosti za različite naredbe i klauzule, uvelike pojednostavljuje rad sa bazom podataka, prvenstveno automatskim ažuriranjem sintakse u odgovarajuću, jer većinom male razlike između varijanti SQL-a mogu prouzročiti velike probleme pri realizaciji funkcionalne baze podataka.

5. Primjer izrade baze podataka u Access-u za potrebe „Altus“ d.o.o.

Projekt je realiziran u programu Access, koji je dio Microsoft Office paketa, a predstavlja integrirani alat za razvoj aplikacija zasnovanih na osobnoj bazi podataka. Access se može smatrati RAD alatom za brzu izradu aplikacija za rad sa bazom podataka (*rapid applications development*). Prema Accessu nalazimo podijeljena mišljenja. Iskusni projektanti baza podataka nemaju baš visoko mišljenje o programskim rješenjima u Accessu, te ih iskustvo vodi drugim sustavima za upravljanje bazom podataka. S druge strane Access je efikasan alat za ne toliko komplicirane projekte koji su podložni promjenama pri razdoblju korištenja. Mnogi ga opisuju čak kao nešto više od baze podataka, pri tome aludirajući na to da Access nije sustav za upravljanje bazom podataka po definiciji, već cjeloviti razvojni alat koji nije u dobroj poziciji kada ga uspoređujemo sa većima i kompleksnijim sustavima koji su neusporedivo skuplji. Unatoč kritikama niti jedan sličan RAD alat mu nije preuzeo primat na tržištu. Prednost mu je kompatibilnost sa ostalim programima Office paketa koji se kako znamo uvelike koriste u svim sferama poslovanja.

Access se sastoji od slijedećih dijelova:

- *DBMS-engine* – njime se osigurava konzistencija i integritet podataka. Koristi relacijski model podataka (podaci se spremaju u tablične strukture, za operacije i manipulaciju podacima se koristi SQL)

- *Query designer* – pomoću njega je omogućen dvojni način kreiranja i mijenjanja SQL upita, korištenjem tabličnog alata ili neposrednim pisanjem SQL modula

- *Form designer* – alat pomoću kojega nam je omogućeno vizualno kreiranje programske forme, odnosno dinamičkog sučelja za korištenje podataka u bazi. To se odnosi na više funkcija – prikaz, brisanje, upis ili ažuriranje podataka

- *Report designer* – pomoću njega na vizualan način kreiramo izvještaje prema podacima iz baze podataka, odnosno prema rezultatima upita. Izuzetno je praktičan, posjeduje funkciju pretpregleda, ispisa i transfera u druge podobne formate (PDF).

- Makro editor – alat pomoću kojeg realiziramo na intuitivan način programske rutine, odnosno makro programe. Vrlo su praktični kada se traže programerska rješenja, a ne posjeduje se programersko znanje, iako su performanse lošije nego pri korištenju VBA editora

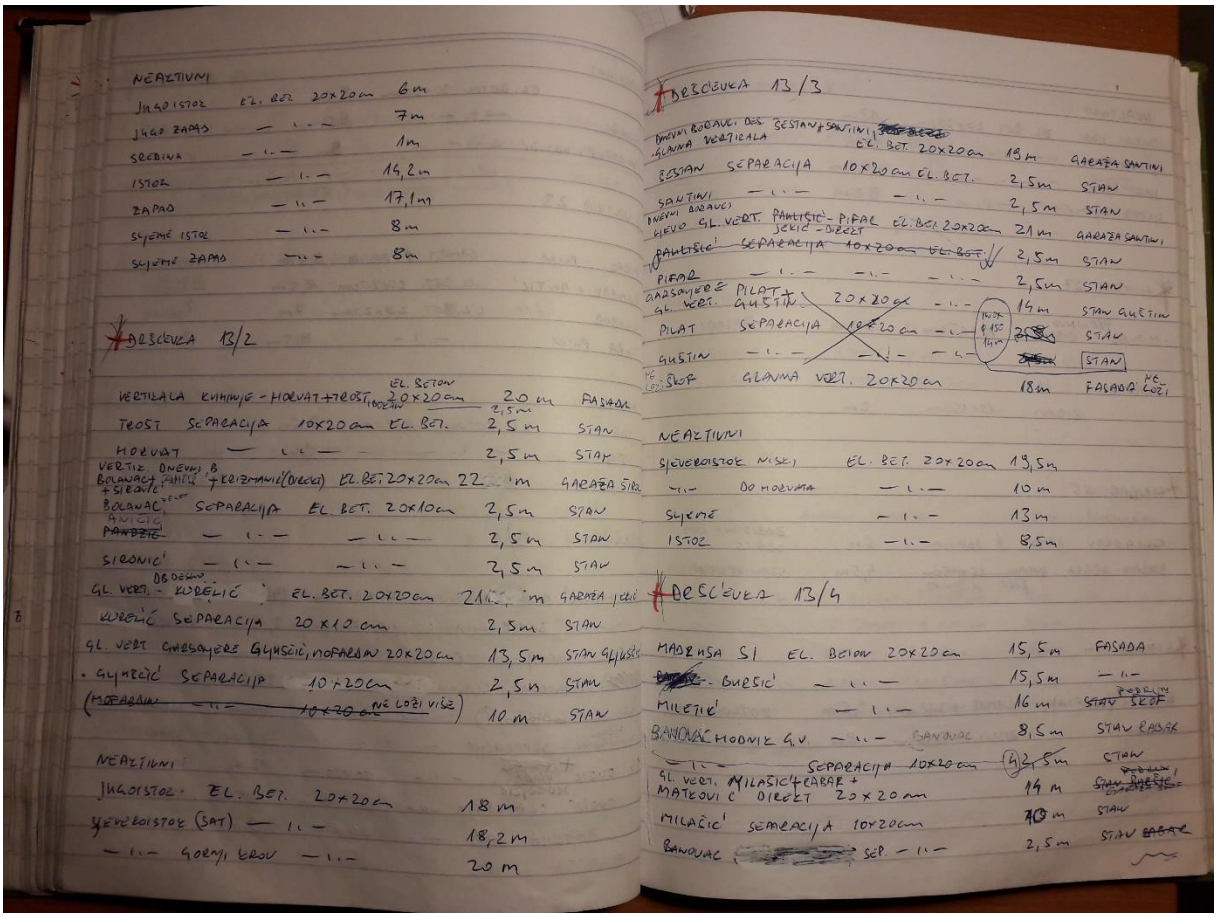
- Visual Basic IDE – interaktivni razvojni editor za programiranje u Accessovoj inačici Visual Basica jezika. To je pravi programski jezik sa razvijenim naprednim osobinama i dodacima za rad sa Access bazom podataka, te mogućnosti korištenja ugrađenog SQL koda. Ovdje se zapravo radi o Visual Basic for applications jeziku (VBA), koji je ugrađeni standard za sve Microsoft Office aplikacije. Korištenjem VBA otvaraju nam se sve mogućnosti realizacije zahtjeva za korištenje baze, i granice mogućnosti su granice vlastitog programerskog znanja.

U ovom smo praktičnom dijelu rada koristili sve navedene dijelove MS Accessa, radi potpunog ispunjavanja zahtjeva korisnika dobivenog prilikom analize potreba i zahtjeva.

5.1 Analiza i specifikacija zahtjeva korisnika

Altus d.o.o. je firma koja između ostaloga obavlja dimnjačarske djelatnosti. Prilikom obavljanje djelatnosti potrebno je ispisati potrebnu papirologiju. Ovaj dio obavlja se ručnim upisivanjem podataka u postojeće formulare, i uzima previše vremena za ponavljajuće upisivanje podataka. Namjera je izradom baze podataka ubrzati i olakšati ispis potrebnih formulara, te implementirati jednostavna rješenja izmjene i dodavanja podataka. Postojeća baza podataka o dimovodnom objektima izvedena je u obliku ručno ispisane bilježnice, koja se ispunjavala paralelno s radom na terenu te je preglednost pri korištenju u najmanju ruku vrlo

upitna. Tome pridonose i izmjene podataka koje su se akumulirale tijekom godina a nisu uvođene sustavno, već se dopisivalo i brisalo podatke po vlastitom nahođenju.



Slika 23: Postojeća baza podataka

Izvor: Autor

Redovna čišćenja dimnjaka obavljaju se dva puta godišnje. Potrebno je za svako obavljeno čišćenje dimovodnog kanala ispuniti Stručni dimnjačarski nalaz, te nakon toga ispuniti Radni nalog temeljem adrese na kojoj se dimovodni kanali nalaze. Sva dokumentacija mora biti numerirana na godišnjoj razini, i sačuvana u evidenciji.

ALTUS d.o.o.
za obavljanje dimnjačarskih djelatnosti
52000 PAZIN, Franine i Jurine 9a
Tel. 098 914 2548
Fax: 052 622 737

STRUČNI DIMNJAČARSKI NALAZ
o ispravnosti dimovodnog kanala

Nalaz broj: _____ Radni nalog broj: _____
Datum: _____
Korisnik: _____
Adresa: _____

PODACI O DIMOVODNOM KANALU

Opis dimnjaka: _____
Visina dimnjaka: _____
Dimnjača: _____
Dimnjak posjeduje ispravan otvor za čišćenje koji se nalazi _____

PODACI O GORIVU

Vrsta: kruto - tekuće - plinovito - kombinirano - ostalo

STANJE DIMOVODNOG KANALA

DIMNJAK JE: **a) ispravan** **b) neispravan**

c) uvjetno ispravan (do otklanjanja nedostataka i naknadnoj vizualnoj kontroli u roku od 15 dana, u protivnom dimnjak se vodi kao **neispravan**).

Ovaj stručni dimnjačarski nalaz izdaje se temeljem članka 38. Zakona o zaštiti od požara (NN br 92/10), članka 8. Odluke o obavljanju dimnjačarskih poslova (Službene novine Grada Pazina br.24/11) i temeljem Zakona o komunalnom gospodarstvu (NN br: 36/95.,70/97.,128/99.,57/00.,129/00.,59/01.,26/03-pročišćeni tekst, 82/04.,110/04., Uredba, 178/04.,38/09.,79/09.,153/09. i 49/11.). Stručni dimnjačarski nalaz služi kao dokaz o ispravnosti ložišno dimovodnih uređaja u svrhu predočenja protupožarnom inspektoru za zaštitu na radu pri nadležnoj policijskoj upravi.

1. Ovaj stručni dimnjačarski nalaz vrijedi 6 mjeseci od dana izdavanja.
2. Dimovodni kanal potrebno je čistiti 2 puta godišnje po gradskoj odluci Članak 17. (Službene novine grada Pazina br 24/11) izmjena je broj Službenih novina.
3. Redovna kontrola ili čišćenje obvezno se provodi najmanje 4 puta godišnje za dimovodne objekte koji se koriste tokom cijele godine Članak 17. (Službene novine grada Pazina br 24/11).

NAPOMENA _____

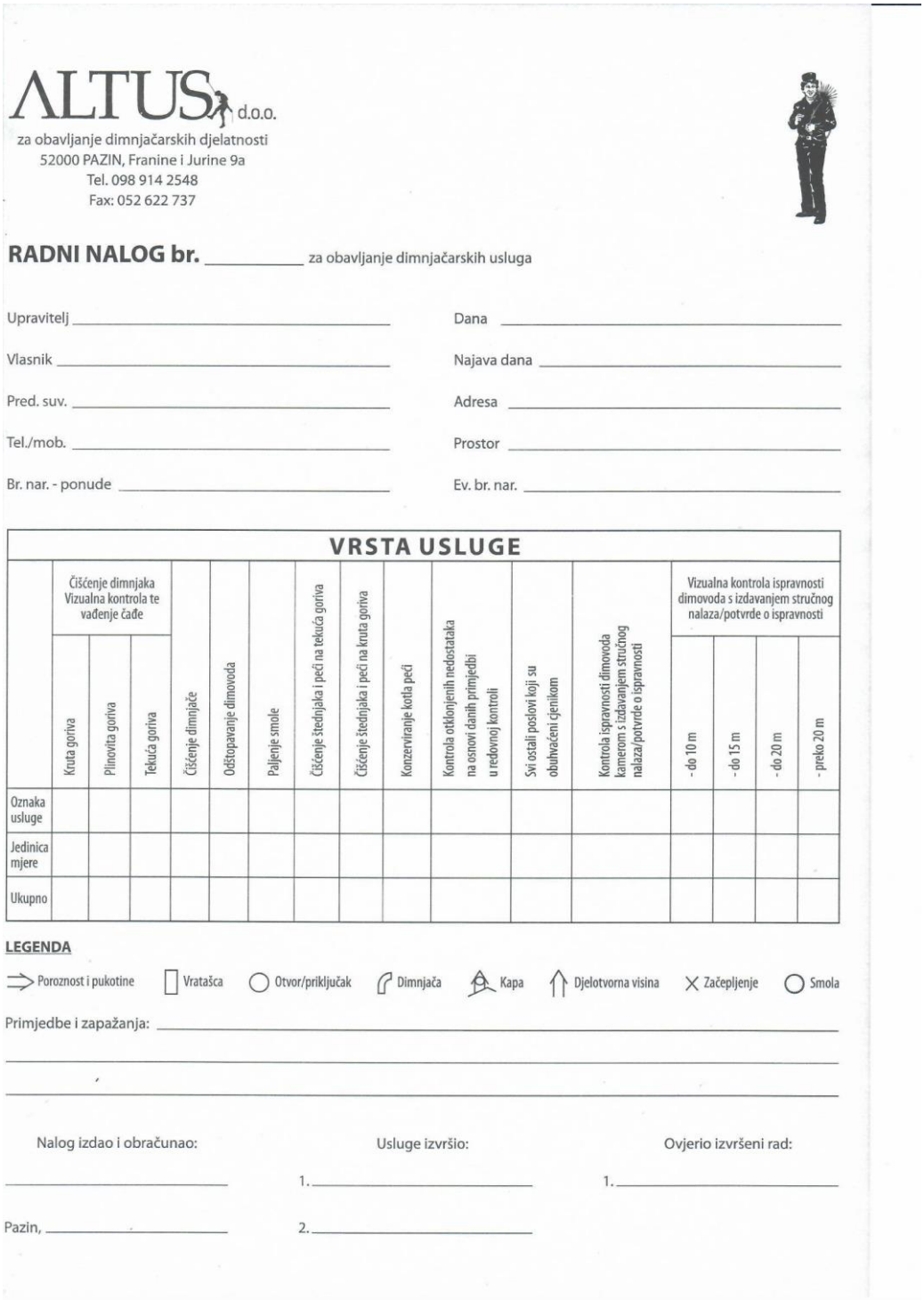
U _____

Potpis i pečat ovlaštenog dimnjačara

Slika 24: Postojeći izgled Stručnog dimnjačarskog nalaza

Izvor: Autor

Stručni dimnjačarski nalaz navodi podatke o pojedinom dimnjaku i čišćenju, te odluku o ispravnosti. Napomene se ispisuju u slučaju nepravilnosti u izvedbi ili radu dimnjaka, te se korisnika upućuje na ispravljanje istih.



ALTUS d.o.o.
za obavljanje dimnjačarskih djelatnosti
52000 PAZIN, Franine i Jurine 9a
Tel. 098 914 2548
Fax: 052 622 737

RADNI NALOG br. _____ za obavljanje dimnjačarskih usluga

Upravitelj _____ Dana _____

Vlasnik _____ Najava dana _____

Pred. suv. _____ Adresa _____

Tel./mob. _____ Prostor _____

Br. nar. - ponude _____ Ev. br. nar. _____

VRSTA USLUGE

	Čišćenje dimnjaka Vizualna kontrola te vađenje čađe			Čišćenje dimnjače	Odštopavanje dimovoda	Paljenje smole	Čišćenje štednjaka i peći na tekuća goriva	Čišćenje štednjaka i peći na kruta goriva	Konzerviranje kotla peći	Kontrola otklonjenih nedostataka na osnovi danih primjedbi u redovnoj kontroli	Svi ostali poslovi koji su obuhvaćeni cijenikom	Kontrola ispravnosti dimovoda kamerom s izdavanjem stručnog nalaza/potvrde o ispravnosti	Vizualna kontrola ispravnosti dimovoda s izdavanjem stručnog nalaza/potvrde o ispravnosti			
	Kruta goriva	Plinovita goriva	Tekuća goriva										- do 10 m	- do 15 m	- do 20 m	- preko 20 m
Oznaka usluge																
Jedinica mjere																
Ukupno																

LEGENDA

Poroznost i pukotine | Vratašca | Otvor/priključak | Dimnjača | Kapa | Djelotvorna visina | Začepljenje | Smola

Primjedbe i zapažanja: _____

Nalog izdao i obračunao: _____

Pazin, _____

Usluge izvršio:
1. _____
2. _____

Ovjerio izvršeni rad:
1. _____

Slika 25: Postojeći izgled Radnog naloga

Izvor: Autor

Radni nalog sadrži podatke o dimnjacima na određenoj adresi i broju, te obavljenim radovima na objektu. Pod pojmom *Prostor* navode se korisnici, a u tablicu *Vrsta usluge* upisuju se zbrojevi vrijednosti za sve dimovodne kanale i obavljene radove na navedenoj adresi. U rubrici *Primjedbe i zapažanja* upisuju se sve napomene navedene u Stručnim dimnjačarskim nalazima koje se odnose na navedenu adresu i radove obavljene navedenog datuma.

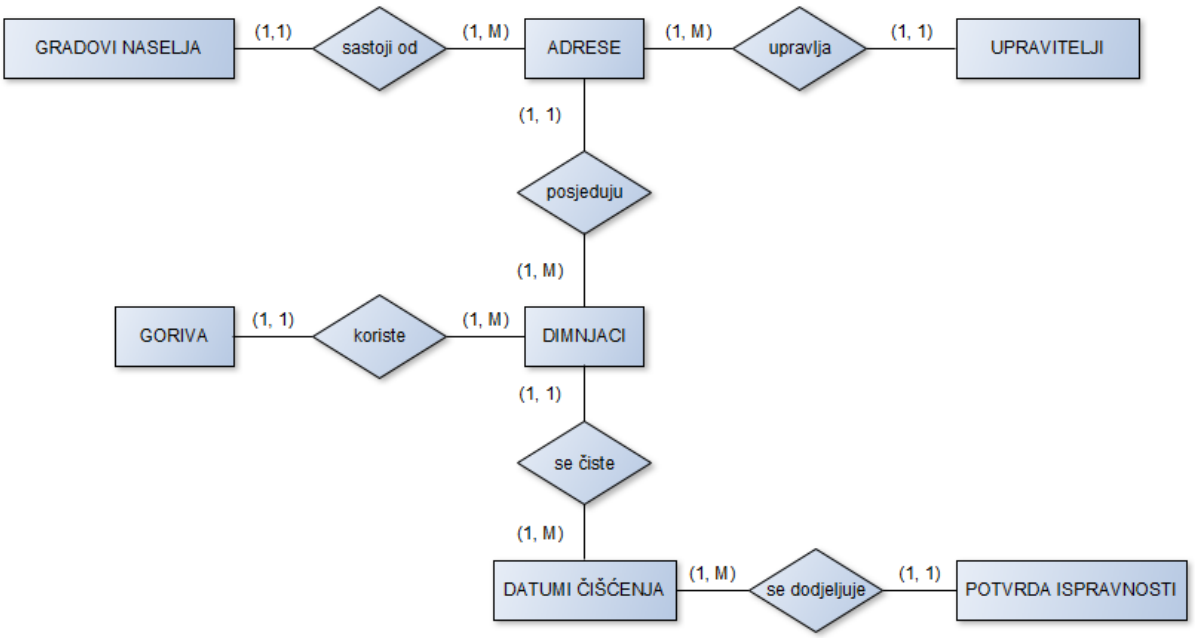
Čišćenja dimovodnih kanala se u pravilu obavljaju po rajonu, tj. ulicu po ulicu, no ima slučajeva gdje se čisti na određenom kućnom broju, a ponekad i pojedinačnog korisnika.

Svi podaci potrebni za ispis navedeni su na formularima, no potrebno je znati i evidencijske brojeve stručnih nalaza na koje se odnosi koji radni nalog radi forme ispostavljanja računa.

Prilikom upisa čišćenja mora postojati mogućnost ažuriranja podataka o korisniku te mogućnost upisa dodatnih radova.

5.2 Izrada E-V modela podataka

Postavljanjem specifikacije zahtjeva i analizom postojećeg stanja dokumentacije naziremo veze i entitete, te pomoću njih konstruiramo ER dijagram.



Slika 26: ER dijagram za potrebe Altus d.o.o

Izvor: Autor

Ovakav dijagram nam zadovoljava sve zahtjeve osim automatiziranog vođenja evidencijskih brojeva. Ovakav dijagram zadovoljava sve zahtjeve za jednogodišnje korištenje, no ulaskom u slijedeću godinu numeracija kreće od početka, odnosno jedan, te bi se u slučaju ovakvog ER dijagrama trebala unositi za svaki izvještaj zasebno. Stoga ćemo konstruirati po dva nova entiteta za pojedinu godinu, i u njima spremati podatke relevantne za izvještaje neovisno od glavne sheme, te za njih nećemo definirati veze, već ćemo proces unosa podataka automatizirati alatima MS Accessa.

Postojeći entiteti sa atributima i dodijeljenim primarnim i stranim ključevima tako su:

GRADOVI NASELJA (Grad naselje ID, Ime grada naselja)

UPRAVITELJI (Upravitelj ID, Upravitelj, Ime kontakta, Telefon)

ADRESE (Adresa ID, Ulica i broj, Vlasnik, Predstavnik stanara, Telefon, Napomena zgrada)

GORIVA (Gorivo ID, Vrsta goriva)

DIMNJACI (Dimnjak ID, Korisnik, Opis, Visina, Vratašca, Aktivan, Do10m, Do15m, Do20m, Preko20m, Adresa ID, Gorivo ID)

DATUMI ČIŠĆENJA (Datumi čišćenja ID, Datum, Napomena, Čišćenje dimnjače, Odštopavanje dimovoda, Paljenje smole, Čišćenje peći na tekuća goriva, Čišćenje peći na kruta goriva, Konzerviranje kotla peći, Kontrola otklonjenih nedostataka, Ostali poslovi, Kontrola ispravnosti dimovoda kamerom, Dimnjak ID, Potvrda ispravnosti ID

POTVRDA ISPRAVNOSTI (Potvrda ispravnosti ID, Stanje dimovoda)

SDN2019 (SDN2019 ID, Ulica i broj, Korisnik, Datum, Napomena, Čišćenje dimnjače, Opis, Visina, Vratašca, Vrsta goriva, Kruto, Tekuće, Plinovito, Ime grada naselja, Stanje dimovoda, Odštopavanje dimovoda, Paljenje smole, Čišćenje peći na tekuća goriva, Čišćenje peći na kruta goriva, Konzerviranje kotla peći, Kontrola otklonjenih nedostataka, Ostali poslovi, Kontrola ispravnosti dimovoda kamerom, Vlasnik, Predstavnik stanara, Telefon, Upravitelj, Do10m, Do15m, Do20m, Preko20m, Napomena korisnik)

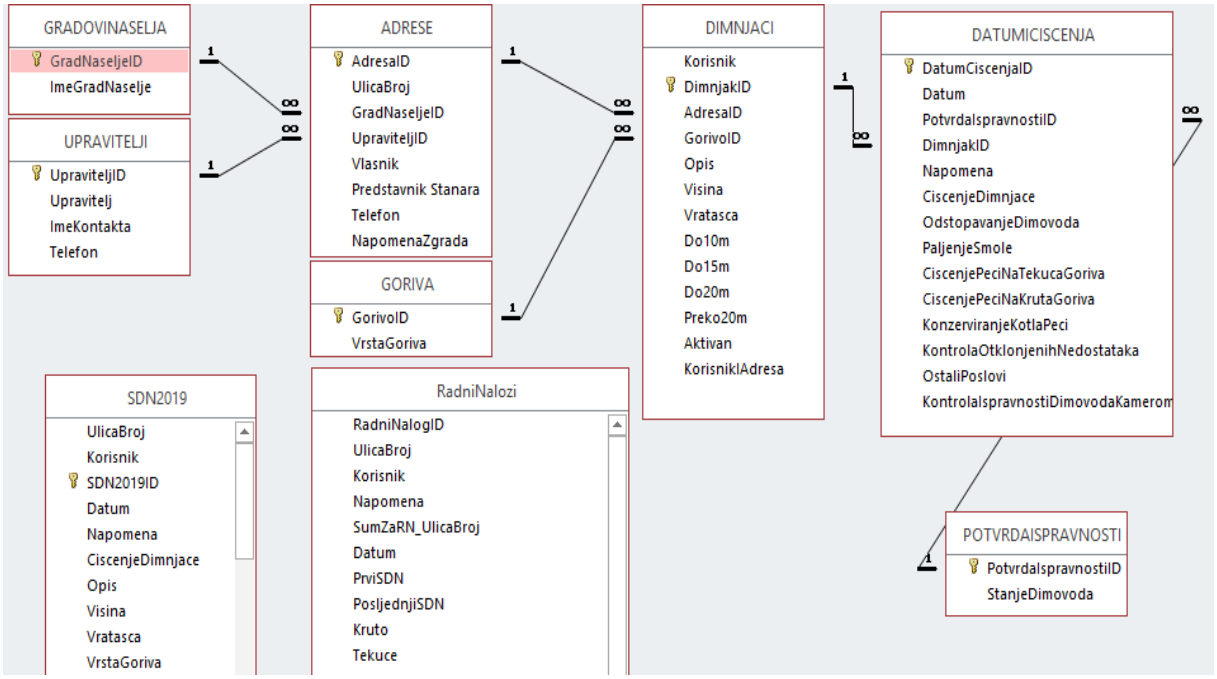
RN2019 (RN2019 ID, Ulica i broj, Korisnik, Datum, Napomena, Kruto, Tekuće, Plinovito, Ime grada naselja, , Odštopavanje dimovoda, Paljenje smole, Čišćenje peći na tekuća goriva, Čišćenje peći na kruta goriva, Konzerviranje kotla peći, Kontrola otklonjenih nedostataka,

Ostali poslovi, Kontrola ispravnosti dimovoda kamerom, Vlasnik, Predstavnik stanara, Telefon, Upravitelj, Do10m, Do15m, Do20m, Preko20m, Evidencijski broj narudžbe, Broj narudžbe ponude)

Prema izrađenom ER modelu izrađujemo relacijski model podataka.

5.3 Izrada relacijskog modela podataka

Za izradu relacijskog modela MS Access ima integrirane funkcije DDL-a, tako da grafičkim stvaranjem tablica vršimo i fizički upis podataka u bazu. Ove funkcije prilično olakšavaju i ubrzavaju izradu same baze podataka, te se eventualne izmjene lako izvršavaju. Pri spremanju izrađene tablice Access automatski prevodi grafički prikaz u SQL kod, odnosno izvršava DDL naredbe.



Slika 27: Relacijski model podataka za potrebe Altus d.o.o.

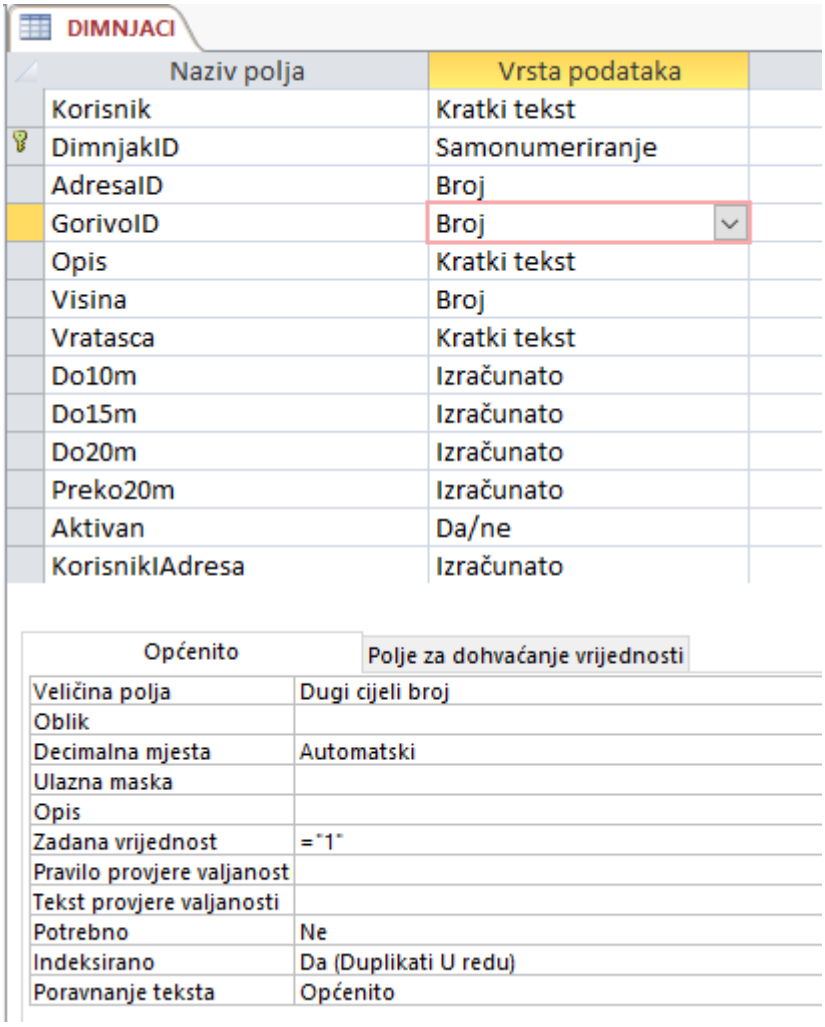
Izvor: Autor

Prilikom upisa atributa u tablicu određujemo tip podataka za svaki atribut. U Accessu postoji nekoliko unaprijed definiranih tipova podataka:

- Number – numerički, može sadržavati nekoliko podtipova numeričkih podataka
- Text – tekst promjenjive duljine do 255 znakova
- Memo – tekst sa preko 255 znakova
- Date/Time – datum/vrijeme
- Yes/No – može sadržavati jednu od dvije logičke vrijednosti (True/False)
- Currency – prikaz novčanih vrijednosti
- Hyperlink – hiperlink veza ili HTTP adresa
- Autonumber – automatski generator rastućih pozitivnih cijelih brojeva
- OLE Object – oznaka OLE objekta (slika, datoteka,…)

Osim određivanja tipa i podtipa podatka, za sprečavanje upisa pogrešnih ili nepravilnih vrijednosti, prilikom izrade tablice definiramo i ograničenja za atribute:

- Default value – vrijednost koja će biti ponuđena pri upisu nove n-torke
- Validation rule – određeni uvjet koji pri nepoštivanju onemogućuje upis unesene vrijednosti
- Validation text – poruka koja se prikazuje pri kršenju uvjeta validacije
- Required – postavljeno na Yes zahtijeva unos ove vrijednosti atributa u n-torku
- Allow zero lenght – postavljeno na No mora biti upisan barem jedan znak
- Indexed – onemogućuje duplikate za vrijednosti atributa (Unique u drugim DBMS)
- Format – definira oblik u kojem će se po defaultu prikazivati podaci polja
- Caption – alternativni naziv polja koji će se ispisati na Access formama
- Input mask – omogućuje defaultni format kod upisa polja na formama



Slika 28: Primjer dizajna tablice

Izvor: Autor

5.4 Dizajniranje obrazaca

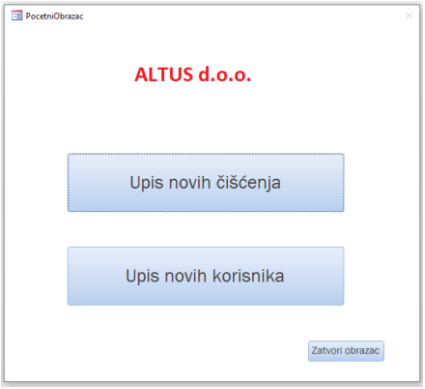
Nakon što smo stvorili strukturu naše baze podataka, ona je potpuno funkcionalna i možemo je koristiti za manipulaciju podacima. Tablični prikaz je, međutim nezgrapan za korištenje korisnicima koji nisu naučeni na rad sa velikom količinom podataka. Stvaranjem obrazaca mi stvaramo korisničko sučelje za krajnjega korisnika, omogućavamo mu brz i efikasan rad, i onemogućavamo ili smanjujemo na minimum greške pri radu uvođenjem ograničenja prema pristupu podacima.

Dizajniranjem i povezivanjem obrazaca mi kreiramo put, odnosno algoritam korištenja podataka iz baze. Pomoću obrazaca prikazujemo podatke iz tabela baze podataka krajnjem korisniku prilagođenu za njegove potrebe, te mu omogućujemo unos novih podataka u tabele.

Pri izradi obrasca prvo ga moramo referirati na neku postojeću tablicu ili upit. Na taj način nam je omogućeno selektiranje atributa tablice koje u obrascu prikazujemo kao tekstni okvir. Ovakvim prikazom obrazac nam omogućuje da u prikazu na jednom listu imamo sve odabrane atribute određene n-torke.

Na obrascima imamo velike mogućnosti dizajniranja prikaza podataka, no kreiranjem i definiranjem *Command buttona* imamo mogućnost pokretanja akcija ili niza akcija prvenstveno prilikom klika. Niz akcija se za jednostavne akcije ostvaruje preko *Control wizarda*, a u dizajniranju niza akcija možemo se referirati na, za ove stvari jako praktičnog *Macro buildera* ili se referirati pak na *Code editora*, gdje onda stvaramo niz akcija u VBA kodu. Vrlo je jednostavna i komunikacija između VBA koda i *Macro buildera*.

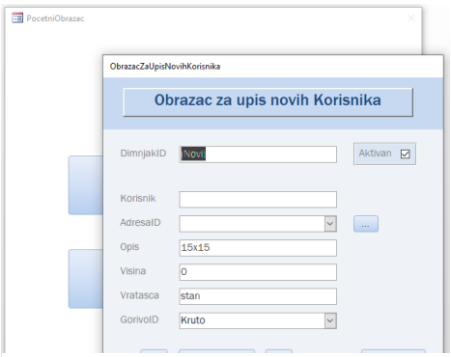
Kao primjer korištenja navigacije po obrascima pomoću *Command buttona* navest ćemo postupak upisa novog korisnika.



Slika 29: Početni obrazac

Izvor: Autor

Nakon obrasca kojim se ulogiramo u bazu podataka Altus d.o.o.-a putem korisničkog imena i lozinke, otvara nam se *Početni obrazac* (Slika 28.)na kojemu odabiremo pritiskom *Command buttona* „Upis novih korisnika“.

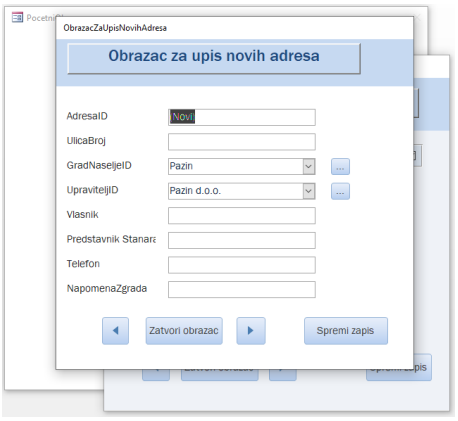


Slika 30: Obrazac za upis novih korisnika

Izvor: Autor

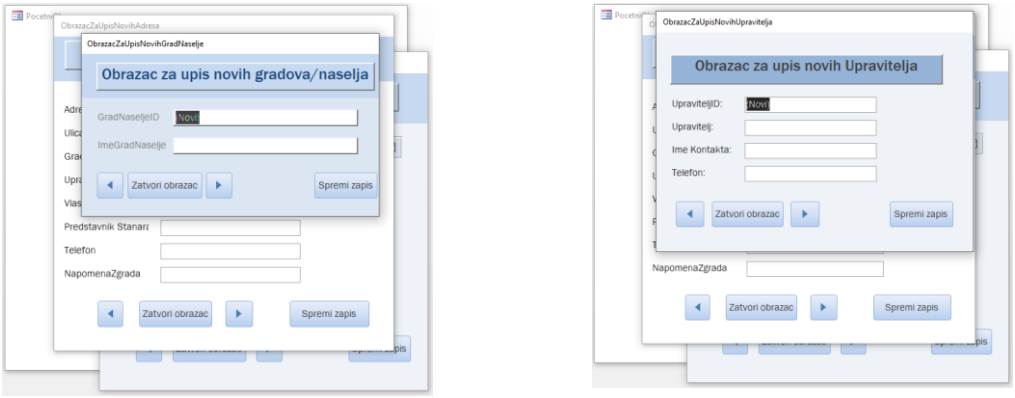
Otvara nam se *Obrazac za upis novih korisnika*, gdje možemo krenuti sa upisom podataka novog korisnika. Za unos vrijednosti Goriva i Adresa predviđen je kombinirani okvir koji preuzima vrijednosti iz drugih relacija. U slučaju da ne postoji upisana željena adresa u tablici Adrese, pritišćemo Command button u nastavku (…) .

U *Obrazac za upis novih adresa* (Slika 30.) upisujemo novu adresu i spremamo zapis. Ukoliko ne postoji tražena vrijednost *Grad* ili *Upravitelj* u kombiniranim okvirima, krećemo dalje i upisujemo ih u njihove tablice pomoću otvorenih obrazaca.



Slika 31: Obrazac za upis novih adresa

Izvor: Autor



Slika 32: Obrasci za upis novih gradova/naselja, te upis novih upravitelja

Izvor: Autor

Kombinirani okvir preuzima vrijednost iz tablice putem upita koji se izvršava prilikom otvaranja obrasca. Nakon upisa novih vrijednosti u tablice, vrijednosti u kombiniranom okviru će ostati nepromijenjene do ponovnog otvaranja obrasca. Da bi korigirali ovakvo nevješto korištenje obrazaca jednostavno rješenje je određivanje komande *Requery* prilikom fokusa na kombinirani okvir, odnosno neposredno prije korištenja istog. Ovim se načinom ponovnog pokretanja upita prikazuju aktualne vrijednosti atributa iz tablice koju kombinirani okvir dohvaća.

5.5 Dizajniranje upita

Upite možemo opisati kao način postavljanja pitanja o podacima u bazi. Upiti se kao i drugi objekti baze spremaju, i kao takve možemo ih pokrenuti u bilo kojem trenutku i dohvatiti

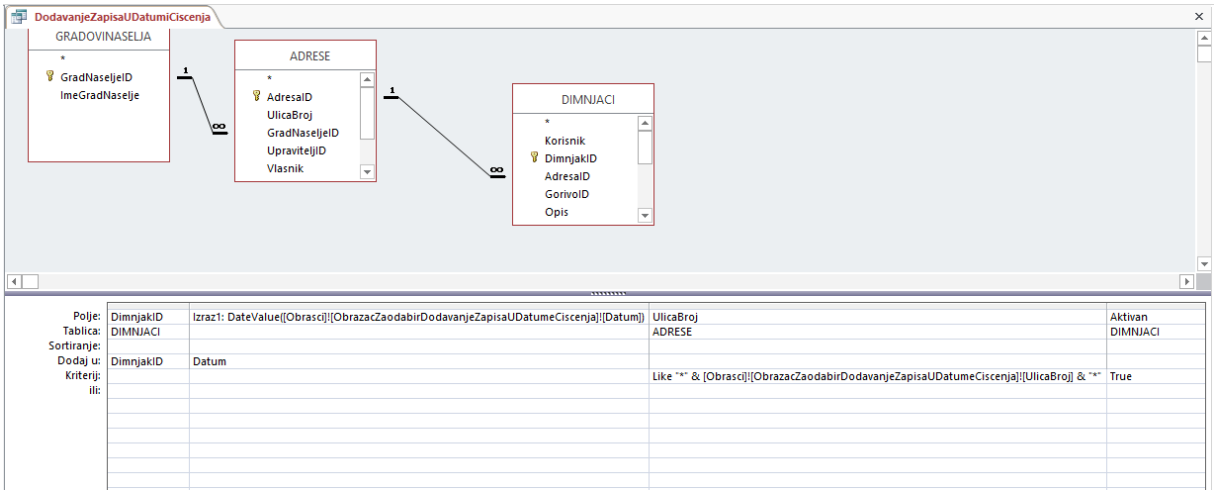
trenutne podatke u bazi definirane kriterijem upita. Spremanje upita ne sprema rezultate upita već dizajn upita, tako da mijenjanjem podataka u bazi dobivamo drugačije rezultate upita. Upitima koristimo da kombiniramo podatke iz povezanih relacija, provodimo izračune i sažetke, te možemo i automatizirati provođenje izmjena u bazi podataka.

Access nam nudi kreiranje upita na tri načina, upotrebom:

- Čarobnjak za upite (*Query Wizard*) – njime možemo ostvariti jednostavne upite
- *Design view* – u pogledu dizajna grafičkim prikazom ostvarujemo upit
- *SQL view* – direktno pisanje SQL naredbi

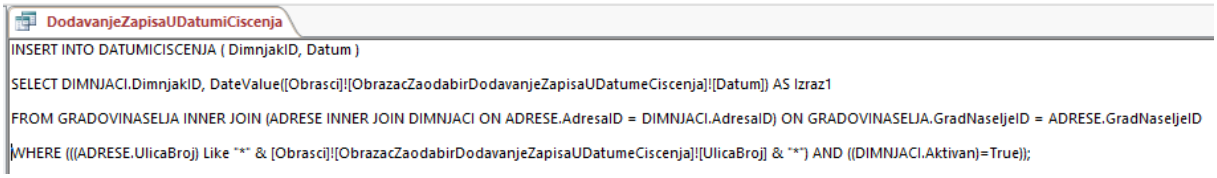
Prva dva načina u velikoj mjeri ubrzavaju proces izrade upita i ne zahtijevaju nikakvo znanje SQL-a, no pomoću njih nije moguće ostvariti zahtjevnije i naprednije upite, koje možemo ostvariti ukoliko upisujemo SQL kod (UNION, LEFT JOIN, OUTER JOIN…). U bilo kojem trenutku kreiranja upita možemo mijenjati prikaz iz pogleda dizajna u SQL prikaz koji se u pozadini generira, i ovo predstavlja dobar uvid u primjere SQL koda i razumijevanje istih, a generirane naredbe se mogu koristiti i u drugim sustavima za upravljanje bazama podataka.

Pokazati ćemo na izvedenom upitu primjer postavljanja upita kroz grafički prikaz i generirani SQL kod.



Slika 33: Grafički prikaz upita u Accessu

Izvor: Autor



Slika 34: Generirani SQL kod

Izvor: Autor

U ovom upitu povezujemo vrijednosti u prvom obrascu za upis novih radova te na osnovu njih dohvaćamo podatke koji zadovoljavaju navedene kriterije. U ovom slučaju definiramo upisanu vrijednost datuma kao *DateValue,* odabiremo samo aktivne korisnike, te korisnike čija adresa zadovoljava *like* kriterij. Sa korištenjem *like* kriterija omogućeno nam je da odaberemo sve korisnike navedene ulice bez obzira na kućni broj, pošto nam je to navedeni uvjet u specifikaciji zahtjeva. Ovaj konkretan upit je vrsta akcijskog upita, pomoću njega dodajemo sve zapise koji zadovoljavaju kriterije *SELECT* upita kao nove redove, odnosno n-torke u drugu tablicu (DATUMICISCENJA). Specifikacija zahtjeva nam nalaže da ostvarimo mogućnost upisa novih radova i na kriteriju adrese, i na kriteriju imena korisnika, i to smo ostvarili sličnim upitima i obrascima.

5.6 Macro Builder i VBA editor

Već smo napomenuli da nam *Macro Builder* otvara velike mogućnosti upravljanja podacima, operacijama i akcijama pomoću stvorene procedure događaja, no *Macro Builder* je alat sa svojim ograničenjima. Grafički dizajn kreiranja naredbi prilikom dizajna procedure događaja omogućuje nam pregledan pristup mogućnostima prilikom programiranja, i jednostavno kreiranje niza naredbi sa unaprijed određenom sintaksom. VBA editor nam omogućuje ostvarivanje i zahtjevnijih upita, koje nije moguće realizirati sa *Macro Builderom*, a neke niti sa SQL-om u Accessu.

Jedan takav zadatak pojavio se pri izradi definicije podataka za ispunjavanje radnih naloga. U radnim nalozima polja *Korisnici* i *Napomena* predstavljaju niz konstruiran iz podataka sa Stručnih dimnjačarskih nalaza koji pripadaju različitim n-torkama. Za rješavanje ove problematike podacima izradili smo kod koji podatke dohvaća iz postojećih tablica, te ih sprema u privremenu tablicu koja se nakon odrađenom upita briše.

```
Private Sub Naredba182_Click()
'Option Compare Database
'On Error Resume Next

Dim db As DAO.Database, rst As DAO.Recordset, sSQL As String
Dim strUlicaBroj As String, strKorisnik As String, strNapomenaKorisnik As String
'Stop
Set db = CurrentDb()
Call RecreateTables(db)

 sSQL = " SELECT UlicaBroj, Korisnik, NapomenaKorisnik, Datum  FROM SDN2019 " _
     & " WHERE UlicaBroj = '" & Me!UlicaBroj & "'  AND Datum LIKE #" & Format(Me![Datum].Value, "yyyy\/mm\/dd") & " # ORDER BY UlicaBroj,Korisnik "

        Debug.Print sSQL
Set rst = db.OpenRecordset(sSQL)
If Not rst.BOF And Not rst.EOF Then
  rst.MoveFirst
  strUlicaBroj = rst!UlicaBroj
  strKorisnik = rst!Korisnik
  strNapomenaKorisnik = rst!NapomenaKorisnik & vbNullString

  rst.MoveNext
  Do Until rst.EOF
    If strUlicaBroj = rst!UlicaBroj Then
      strKorisnik = strKorisnik & ", " & rst!Korisnik
      strNapomenaKorisnik = strNapomenaKorisnik & (", " + rst!NapomenaKorisnik)
    Else
      sSQL = "INSERT INTO tblDrugaProba (UlicaBroj, Korisnik, Napomena)  " _
           & "VALUES('" & strUlicaBroj & "','" & strKorisnik & "', '" & strNapomenaKorisnik & "')"
      db.Execute sSQL
      strUlicaBroj = rst!UlicaBroj
      strKorisnik = rst!Korisnik
      strNapomenaKorisnik = rst!NapomenaKorisnik
    End If
    rst.MoveNext
  Loop

  ' dodavanje zadnjeg zapisa

  sSQL = "INSERT INTO tblDrugaProba (UlicaBroj, Korisnik, Napomena)  " _
       & "VALUES('" & strUlicaBroj & "','" & strKorisnik & "', '" & strNapomenaKorisnik & "')"

  db.Execute sSQL
End If

Set rs = Nothing
Set db = Nothing

DoCmd.OpenQuery ("uptSveZaRN")
DoCmd.OpenForm ("ObrazacZaRN")
DoCmd.DeleteObject acTable, "tblDrugaProba"

End Sub
```


Slika 35: VBA kod za *Korisnici* i *Napomena*

Izvor: Autor

Ovaj kod pokrećemo pritiskom *Command Buttona* na obrascu, i pokrećemo akcije čijih je rezultat novi zapis, odnosno n-torka u tablici RADNINALOZI. Na početku koda dimenzioniramo varijable, te zatim dodjeljujemo vrijednosti varijablama. Naredba *Call RecreateTables* nam poziva drugu subproceduru koja stvara tablicu *DrugaProba* sa atributima *UlicaBroj, Korisnik, Napomena. Recordset* otvaramo na rezultatima sSQL upita, i koristimo ga za dohvaćanje podataka u niz. U Ssql-u mora se formatirati vrijednost datuma da bi se upit izvršio. Tome je razlog oblik datuma koji koristimo u obrascu i tablicama, a to je kratki datum oblika npr. 22.8.2019., te ovakav generira grešku. Formatiranjem u oblik 22/8/2019 SQL kod se uspješno izvršava. Vrijednosti *UlicaBroj* i *Korisnik* niti u jednom slučaju neće biti jednake null vrijednosti, no vrijednosti *NapomenaKorisnik* može biti jednaka null vrijednosti ako napomene za korisnika nema. Ova pojava null vrijednosti u kodu nam također generira grešku. Rješenje za to je dodavanje vrijednosti *vbNullString* sa kojom VBA može izvršiti kod. Zadnjim linijama koda se pozivamo na prethodno definirane objekte u Accessu. Upitom *uptSveZaRN* dohvaćamo podatke iz tablice *DrugaProba*, te ostale podatke obrađene putem upita za potrebe ispunjavanja vrijednosti polja u tablici *Vrsta usluge* u formi radnog naloga. Otvaranjem obrasca

ObrazacZaRN imamo pregled podataka dobivenog prethodnim koracima te iz njega odabirom Command buttona pokrećemo realizaciju ispisa pomoću prethodno izgrađenog oblika izvješća, odnosno reporta.

5.7 Dizajniranje izvješća

Izvješća se mogu definirati kao statički prikaz aktualnih vrijednosti podataka u bazi, prilagođen za ispis na papir.

Dizajniranjem oblika izvješća moguće je dodatno grupiranje i prilagodba podataka, podataka, a to može predstavljati neka bazna relacija ili postojeći upit.

Za potrebe dizajniranja izvješća potrebno je razumjeti koncept, strukturu dizajna izvješća:

Report Header – zaglavlje izvještaja, pojavljuje se na početku izvještaja

Page Header – zaglavlje stranice, ispisuje se na početku svake stranice izvješća

Xn Header – zaglavlje grupe najviše razine – X je polje grupiranja, n je razina

---

X1 Header – zaglavlje grupe najniže razine

Detail – detaljni podaci koji se prikazuju u ponavljajućim redovima unutar najniže grupe

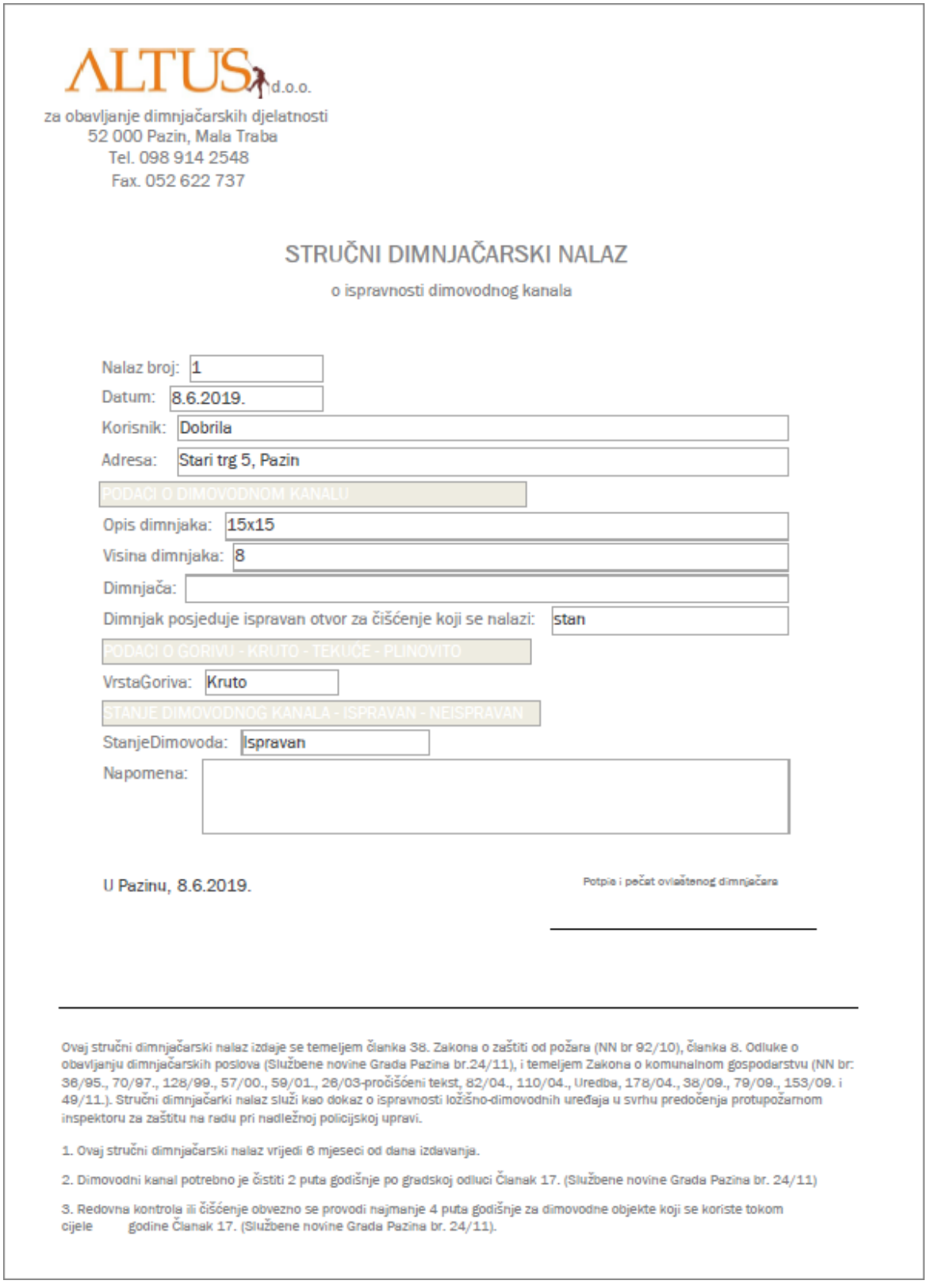
X1 Footer – podnožje grupe najniže razine

Xn Footer

Page Footer - podnožje stranice, ispisuje se na kraju svake stranice izvješća

Report Footer – podnožje izvještaja, ispisuje se na kraju izvještaja

Za potrebe ispisa radnih naloga i stručnih nalaza, u dizajnu izvješća bilo je dovoljno koristiti zaglavlje i podnožje stranice, te detaljne podatke unutar jedne razine grupe.



ALTUS d.o.o.

za obavljanje dimnjačarskih djelatnosti
52 000 Pazin, Mala Traba
Tel. 098 914 2548
Fax. 052 622 737

STRUČNI DIMNJAČARSKI NALAZ

o ispravnosti dimovodnog kanala

Nalaz broj: 1
Datum: 8.6.2019.
Korisnik: Dobrila
Adresa: Stari trg 5, Pazin

PODACI O DIMOVODNOM KANALU

Opis dimnjaka: 15x15
Visina dimnjaka: 8
Dimnjača:
Dimnjak posjeduje ispravan otvor za čišćenje koji se nalazi: stan

PODACI O GORIVU - KRUTO - TEKUĆE - PLINOVITO

VrstaGoriva: Kruto

STANJE DIMOVODNOG KANALA - ISPRAVAN - NEISPRAVAN

StanjeDimovoda: Ispravan
Napomena:

U Pazinu, 8.6.2019.

Potpis i pečat ovlaštenog dimnjačara

Ovaj stručni dimnjačarski nalaz izdaje se temeljem članka 38. Zakona o zaštiti od požara (NN br 92/10), članka 8. Odluke o obavljanju dimnjačarskih poslova (Službene novine Grada Pazina br.24/11), i temeljem Zakona o komunalnom gospodarstvu (NN br: 36/95., 70/97., 128/99., 57/00., 59/01., 26/03-pročišćeni tekst, 82/04., 110/04., Uredba, 178/04., 38/09., 79/09., 153/09. i 49/11.). Stručni dimnjačarki nalaz služi kao dokaz o ispravnosti ložišno-dimovodnih uređaja u svrhu predočenja protupožarnom inspektoru za zaštitu na radu pri nadležnoj policijskoj upravi.

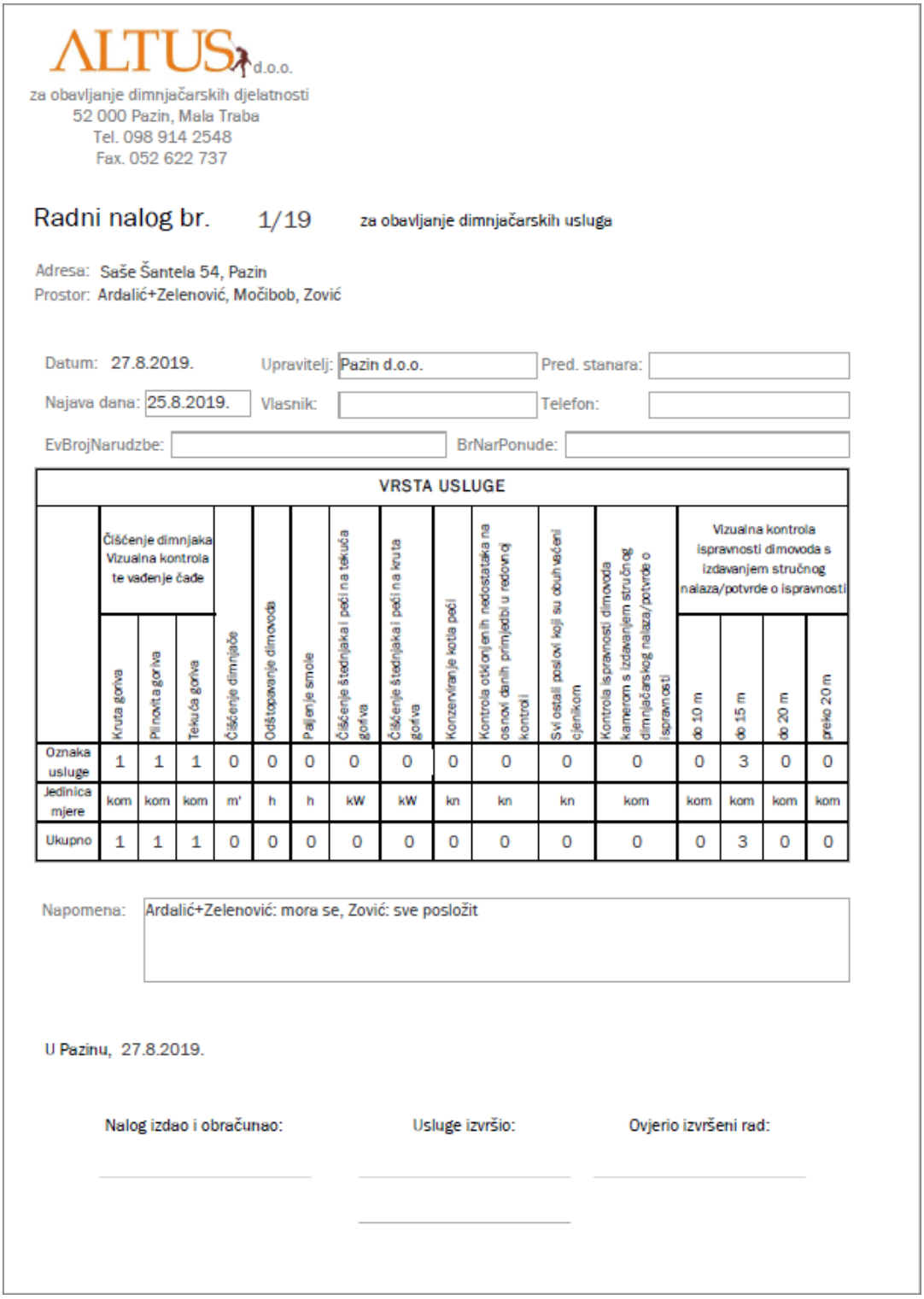
1. Ovaj stručni dimnjačarski nalaz vrijedi 6 mjeseci od dana izdavanja.

2. Dimovodni kanal potrebno je čistiti 2 puta godišnje po gradskoj odluci Članak 17. (Službene novine Grada Pazina br. 24/11)

3. Redovna kontrola ili čišćenje obvezno se provodi najmanje 4 puta godišnje za dimovodne objekte koji se koriste tokom cijele godine Članak 17. (Službene novine Grada Pazina br. 24/11).

Slika 36: Izvješće Stručnog nalaza

Izvor: Autor



ALTUS d.o.o.
za obavljanje dimnjačarskih djelatnosti
52 000 Pazin, Mala Traba
Tel. 098 914 2548
Fax. 052 622 737

Radni nalog br. 1/19 za obavljanje dimnjačarskih usluga

Adresa: Saše Šantela 54, Pazin
Prostor: Ardalić+Zelenović, Močibob, Zović

Datum: 27.8.2019. Upravitelj: Pazin d.o.o. Pred. stanara:
Najava dana: 25.8.2019. Vlasnik: Telefon:
EvBrojNarudzbe: BrNarPonude:

VRSTA USLUGE																
	Čišćenje dimnjaka Vizualna kontrola te vađenje čađe			Čišćenje dimnjaka	Odstopavanje dimovoda	Paljenje smole	Čišćenje štednjaka i peći na tekuća goriva	Čišćenje štednjaka i peći na kruta goriva	Konzerviranje kotla peći	Kontrola otklonjenih nedostataka na osnovi danih primjedbi u radovnoj kontroli	Svi ostali poslovi koji su obuhvaćeni cijenikom	Kontrola ispravnosti dimovoda kamerom s izdavanjem stručnog dimnjačarskog nalaza/potvrde o ispravnosti	Vizualna kontrola ispravnosti dimovoda s izdavanjem stručnog nalaza/potvrde o ispravnosti			
	Kruta goriva	Pil novita goriva	Tekuća goriva										do 10 m	do 15 m	do 20 m	preko 20 m
Oznaka usluge	1	1	1	0	0	0	0	0	0	0	0	0	0	3	0	0
Jedinica mjere	kom	kom	kom	m'	h	h	kW	kW	kn	kn	kn	kom	kom	kom	kom	kom
Ukupno	1	1	1	0	0	0	0	0	0	0	0	0	0	3	0	0

Napomena: Ardalić+Zelenović: mora se, Zović: sve posložit

U Pazinu, 27.8.2019.

Nalog izdao i obračunao: Usluge izvršio: Ovjerio izvršeni rad:

Slika 37: Izvješće Radnog naloga

Izvor: Autor

Uspješno implementiranim rješenjima, pomoću metoda i tehnika obrađenih u ovom radu ostvarili smo zadani cilj – ispunjeni radni nalozi i stručni nalazi bez nepotrebnog prepisivanja, i uspješno odrađene radnje sa podacima koje prethode ispisu.

6. ZAKLJUČAK

Gledajući kronologiju razvoja baza podataka usporedno sa razvojem tehnike, uviđamo da su one neophodne za praktično funkcioniranje sustava. Sve veći zahtjevi korisnika i korištenje sve većeg broja vrste podataka postavljaju ciljeve koje objektne baze podataka mogu uspješno dostići, no raširena upotreba već uhodanih relacijskih baza ne predaje svoje mjesto unatoč slabijim performansama sustava.

Metodike analize podataka i konceptualni dijagrami baze predstavljaju, usprkos uhodanim i definiranim koracima izrade, mjesto gdje sposobnost kreativnog razmišljanja dolazi do izražaja. Iako se pri izradi zahtijeva velika analitičnost u pristupu, te upoznavanje i razumijevanje kompletnog procesa sustava koji opisujemo, realizacija kvalitetnih rješenja ukonstruiranih u dijagram gotovo redovito iznenadi jednostavnošću.

Na kraju izrade praktičnog dijela ovog rada dolazi se do zaključka da MS Access kao RAD alat stvarno zasluženo drži ljestvicu visoko prema ostalim sličnim alatima na tržištu, no nikako ne smijemo zanemariti mogućnosti koje nam otvara u širini pristupa rješavanju zahtjeva. Realizacija fizičkog dizajna baze je izuzetno pregledna i jednostavna, a aplikacijski dio sa svojim alatima nam pruža mnoštvo izbora u oblikovanju i prezentaciji podataka.

Izradom praktičnog djela ovog rada ostvarena je funkcionalna baza podataka koja ispunjava sve uvjete iznesene u specifikaciji zahtjeva. Ostvarivanjem ovog cilja otvorene su nam mogućnosti unaprjeđenja obrade podataka, te se javlja namjera izrade dodatne funkcije koja bi automatizirala i ispise računa u formi potrebnoj za poslovanje, te povezivanje Access baze sa Excell tablicama radi prijenosa odabranih podataka. Mogućnosti koje nam pruža Access se otvaraju, ali pripadati će temi nekog slijedećeg projekta.

POPIS LITERATURE

Knjige i publikacije:

1. Manger R.(2014.): Baze podataka, Element, Zagreb
2. Pavlić M.(1996.): Razvoj informacijskih sustava, CIP, Zagreb
3. Carić T., Buntić M.(2015.): Uvod u relacijske baze podataka, Zagreb
4. Manger R.(2003.): Baze podataka-skripta, PMF, Zagreb
5. Vuk D.,Ciriković E.(2015.): Priručnik za laboratorijske vježbe iz baza podataka, Virovitica
6. Ljubičić I.(2014.): Napredne baze podataka – Microsoft Access 2010, priručnik, ODRAZI, Zagreb
7. Manger R.(2010.): Osnove projektiranja baza podataka, SRCE, Zagreb
8. Prof.dr.sc. Vukmirović S.(2013.):Modeliranje i analiza podataka u poslovanju, Sveučilište u Rijeci, Rijeka

Izvori s interneta:

1. https://www.databasejournal.com/features/msaccess/article.php/2247531/Concatenate-Column-Values-from-Multiple-Rows-into-a-Single-Column-with-Access.htm(22.08.2019.)
2. https://stackoverflow.com/questions/50289586/obtaining-the-correct-date-using-vba/50289973(23.08.2019.)
3. http://uni-mo.sve-mo.ba/~goran/nastava/BAZE_Nastava.pdf (19.02.2019.)
4. http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf (16.2.2019.)
5. https://bib.irb.hr/datoteka/556923.OBP7_skraceno.pdf, str. 25 (11.3.2019.)

POPIS SLIKA

POPIS TABLICA