Poslovna aplikacija za dostavnu službu nad relacijskom bazom podataka - Clarion

Štorić, Luka

Sveučilište u Rijeci – Odjel za informatiku

Preddiplomski jednopredmetni studij Informatike

Luka Štorić

Poslovna aplikacija za dostavnu službu nad relacijskom bazom podataka - Clarion

Završni rad

Mentor: prof. dr. sc. Patrizia Poščić

Rijeka, 11. rujna 2019.

Sadržaj

1.Uvod

Poslovna aplikacija za dostavnu službu nad relacijskom bazom podataka u alatu Clarion. Iako sam naslov na prvu djeluje zbunjujuće, u ovom radu pokazat će se da se radi o zanimljivoj i praktičnoj temi.

Na preddiplomskom jednopredmetnom studiju informatike Odjela za informatiku u Rijeci postoje dva kolegija koja studente uvode u „svijet“ baza podataka. Kolegij Uvod u baze podataka, kao što mu i ime kaže, prikazuje prvu, okvirnu sliku, a kolegij Baze podataka malo dublje ulazi u temu raznih načina na koji se informacije u kontekstu spremaju i koriste. Današnji se svijet, povezan i digitaliziran kakav je, velikim dijelom oslanja na baze podataka, njihovu detaljnost, pristupačnost i naravno sigurnost. Ne postoji niti jedna tvrtka, organizacija ili služba koja se ne koristi nekom vrstom spremišta podataka, a u novije vrijeme to su većinom upravo poslovne aplikacije koje služe kao sučelje prema bazi.

U ovom je radu detaljno opisan proces izrade jedne takve baze. Pokazano je da su za to potrebni određeni koraci, odnosno da se sama aplikacija sastoji od nekoliko glavnih dijelova koji su međusobno ovisni jedan o drugome. Uz pomoć primjera iz stvarnog života, citata iz literature te s*creenshotova* sučelja dan je opis procesa stvaranja relacijske baze podataka kao i aplikacije preko koje će korisnik unositi, mijenjati, ažurirati, uspoređivati te brisati same podatke.

Bitno je razumjeti i način izrade same aplikacije. U današnje doba, kao i sve ostalo, cilj je razviti računalne programe čim prije, čim lakše ali i čim kvalitetnije. Tako je nastala metodologija RAD, a paralelno s njom i RAD alati. RAD (*Rapid application development*) je metodologija koja se umjesto dugog i detaljnog planiranja temelji na brzom izbacivanju prototipa te povratnoj informaciji. Zato su RAD alati osmišljeni na način da *developeru* pružaju pojednostavljeno sučelje u kojem on može doslovno povući elemente svoje aplikacije i slagati ih po volji. Takvi softveri štede vrijeme i resurse neovisno o tome što prvi rezultat možda neće biti dobar. Ono što ne radi ispravno, na jednostavan se način može popraviti i do krajnjeg rezultata se dolazi kroz kratko vrijeme[2].

Za izradu aplikacije u ovom radu korišten je alat Clarion. Radi se o softveru američke tvrtke SoftVelocity koji se služi svojim jezikom (Clarion Programming Language) za kompajliranje koda same aplikacije. Sučelje je, dakle, olakšano na način da je aplikaciju jednostavno oformiti korištenjem gotovih elemenata, a uz minimalno korištenje samog koda. Na mjestu gdje je ipak bilo potrebno napisati nekoliko linija koda, to će u radu biti posebno naglašeno i opisano za bolje shvaćanje.

Sam rad konstruiran je u obliku poglavlja. Svako poglavlje bavi se jednim od tri glavna koraka razvijanja aplikacije nad relacijskom bazom. U prvom poglavlju objasnit će se što je relacijski model, zašto je on osnovni temelj izrade aplikacije, kako ga zapisati i prikazati, koja pravila i prakse vrijede, kako je povezan sa dijagramom Entiteti-Veze, koji su elementi samog dijagrama te njihovo značenje i međusobna povezanost, a sve to na primjeru poslovnog sustava kojim se bavi ovaj rad.

Drugo poglavlje bavi se izradom rječnika. Pokazat će se kako je upravo ta .dct datoteka alatu Clarion najvažnija uputa kako kreirati aplikaciju. Objasnit će se način unosa tablica te njihovih stupaca u koje će korisnik unositi podatke pri korištenju aplikacije. Uz detaljan opis postavki samih stupaca, biti će objašnjene veze među tablicama i koliku ulogu igraju u samoj aplikaciji.

Zadnje poglavlje prikazuje izradu aplikacije u RAD alatu. Tu se vidi ta samostalnost razvojnog softvera, ali i potreba za ponekom „intervencijom“ *developera*. Opisan je način stvaranja i uređivanja svih prozora aplikacije te njihovog povezivanja. Prikazana je implementacija funkcionalnosti koje Clarion nudi te njihova korisnost.

Aplikacija je nazvana „Deli Serv“ kao dvije skraćenice engleskih riječi „Delivery“ i „Service“.

2.Proces izrade poslovne aplikacije nad relacijskom bazom podataka

2.1.Izrada relacijskog modela

Relacijski model predstavlja osnovnu ideju i preduvjet za izradu aplikacije. On je opis poslovnog procesa, prikazuje glavne elemente i njihove odnose.

Sam pojam pojavio se 60-tih godina 20.stoljeća u radovima Edgara Codda. Model se koristi relacijama koje Codd objašnjava na sljedeći način: „Neka su dani skupovi D1, D2,..., D, (ne obavezno različiti), R je relacija (engl. *Relation*) nad ovih n skupova (n > 0) ako je to skup n-torki takav da za svaku n -torku vrijedi da je prvi element n-torke iz D_1, drugi iz D_2,..., n-ti iz D_n. Definiciju se može lakše shvatiti iz zapisa relacije u obliku *Naziv_relacije (Naziv_atributa1, Naziv_atributa2,…)* tj. ako je neposredno zapišemo kao *Naziv_tablice (Naziv_stupca1, Naziv stupca2…)*. Zapis svih tih relacija iz modela naziva se relacijska shema baze podataka."

Ovdje je bitno spomenuti i model Entiteti-Veze te njegov pripadni dijagram. Referenca [1] opisuje model EV ovako: to je „…grafički prikaz međusobno povezanih grupa podataka promatranog sustava. EV je semantički bogata metoda za modeliranje podataka jer raspolaže ljudski bliskim konceptima". Dijagram na kojem će se opisati odabrani poslovni proces definiran je kao „… grafički prikaz modela podataka sustava, metodom EV"[1]. Upravo zato, on će biti primarni materijal za grafički prikaz relacijske sheme. Glavni koncepti DEV-a su: ENTITET, VEZA, ATRIBUT, SLAB ENTITET I SPECIJALNE VEZE, GENERALIZACIJA te AGREGACIJA. Niže u radu, biti će objašnjeni i ovi pojmovi, u kontekstu DEV-a ovog odabranog sustava.

Kroz terminologiju modela Entiteti-Veze detaljno će biti opisan ovaj poslovni proces, tj. kako je zamišljen. On sam je kombinacija stvarnih procesa, primjera pronađenih na Internetu te osobne mašte. Biti će opisan svaki element i uz pomoć literature [1] pojašnjen na koji način oni funkcioniraju.

2.1.1.Relacijska shema

Prvo je potrebno shvatiti što znači relacijska baza podataka. Ovako je to opisano u knjizi [1]: „Relacijska baza podataka (engl. *relational database*) je skup u vremenu promjenljivih relacija opisanih u shemi baze podataka."

Shemu baze opisuje definicija strukture relacijskog modela: „Struktura relacijskog modela definira se shemom relacijskog modela podataka. Elementi strukture relacijske sheme su relacije."[1].

Da bi se bolje razumjelo prikazana je relacijska shema baze podataka, a odmah potom i dijagram Entiteti-Veze ovog sustava.

DOSTAVNI_CENTAR (Sifra_dost_centra, Naziv_dost_centra, Adresa_dost_centra, Podrucje_djelovanja, Broj_dostavljaca, Broj_vozila)

DOSTAVNO_VOZILO (Broj_dost_vozila, Marka, Model, Godina, Datum_regist, Datum_servisa)

DOSTAVLJAC (Sifra_dostavljaca, Ime_dostavljaca, Prezime_dostavljaca, Kontakt, Status_dostavljaca, Ocjena_za_mjesec, Broj_dostava, Broj_sati, Broj_sati_ned_blagdan, Ukupno_placa)

SATNICA (SIfra_satnice, Naziv_satnice, Iznos, Veca_manja)

PRIMA (Sifra_dostavljaca, Sifra_satnice, Iznos_ukupno)

STANJE_POSILJKE (Sifra_stanja, Opis_stanja)

STATUS_POSILJKE (Sifra_statusa, Opis_statusa)

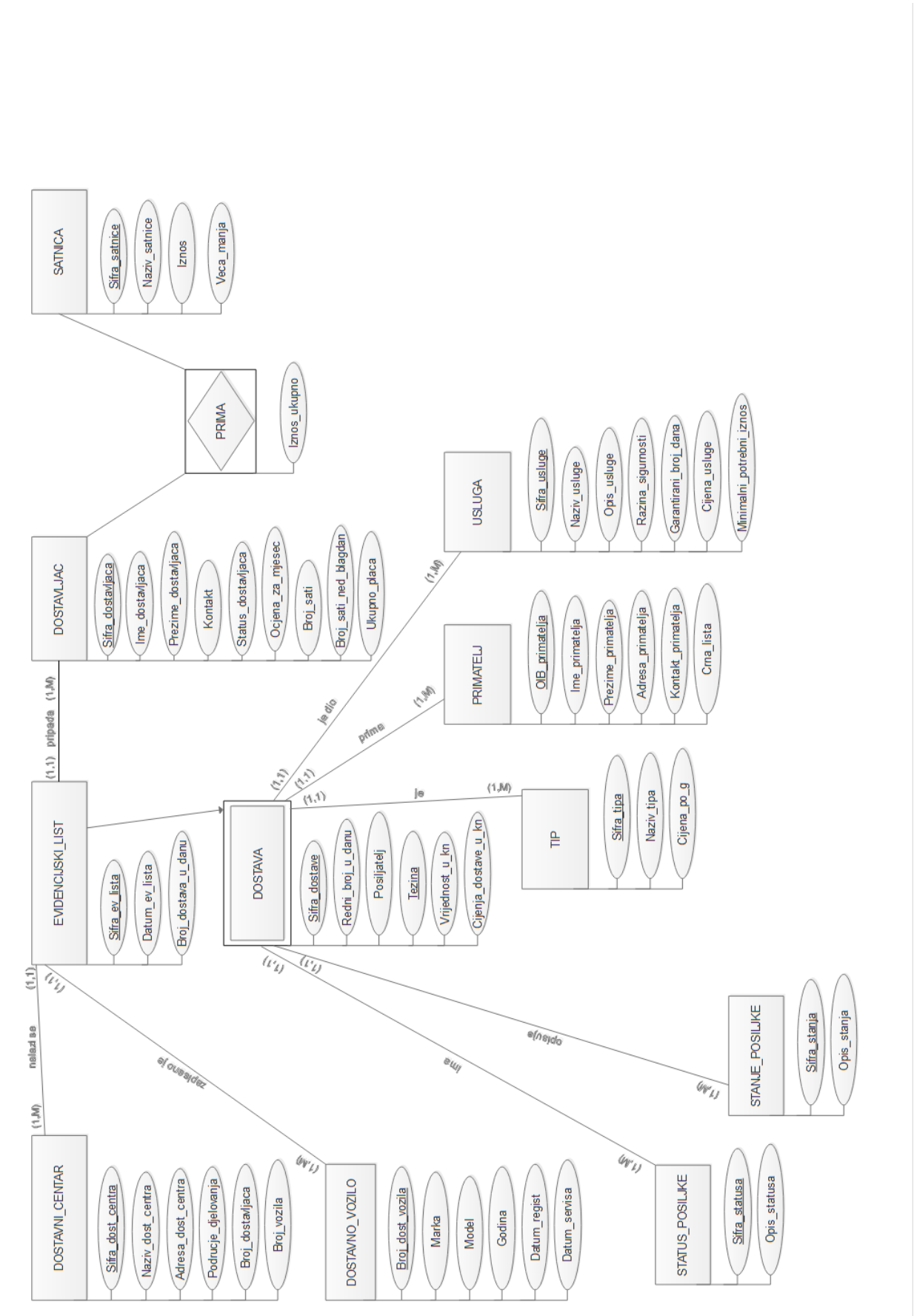
PRIMATELJ (OIB_primatelja, Ime_primatelja, Prezime_primatelja, Adresa_primatelja, Kontakt_primatelja, Crna_lista)

TIP (Sifra_tipa, Naziv_tipa, Cijena_po_g)

USLUGA (Sifra_usluge, Naziv_usluge, Opis_usluge, Cijena_usluge, Razina_sigurnosti, Garantirani_broj_dana, Minimalni_potrebni_iznos)

DOSTAVA (Sifra_ev_lista, Sifra_dostave, Rbr_u_danu, Posiljatelj, OIB_primatelja, Tezina, Vrijednost_u_kn, Sifra_tipa, Sifra_usluge, Sifra_stanja_posiljke, Sifra_statusa_dostave, Cijena_dostave_ukupno)

EVIDENCIJSKI LIST (Sifra_ev_lista, Datum_ev_lista, Broj_dostava_u_danu, Sifra_dost_centra, Sifra_dostavljaca, Broj_dost_vozila)



Slika 1. Dijagram Entiteti-Veze

2.1.2.Objašnjenje DEV-a i samog poslovnog sustava

Na prvi pogled teško je uočiti značenje ovakvog prikaza. Zato je bitno razumjeti spomenute koncepte i njihov odnos prije analiziranja DEV-a ovog sustava.

Prvi koncept je entitet. Prema [1], „Entitet (engl. *entity*) sustava je neki njegov realni ili konceptualni element, to je neka posebnost što u poslovnome sustavu postoji i jasno se razlikuje od drugih entiteta.“. Dakle to su osnovni elementi procesa, a u bazi podataka predstavljaju tablice u koje će se podaci zapisivati. U dijagramu Entiteti-Veze oni se prikazuju četverokutima s nazivima napisanim velikim tiskanim slovima.

Dalje treba shvatiti značenje pojmova u elipsama ispod entiteta. Oni se nazivaju atributi. Opet prema knjizi „Atribut entiteta (engl. *attribute*) je neko svojstvo entiteta.“[1], a u tablicama predstavljaju stupce.

Ako atribut (ili više njih) ispunjava Uvjet jedinstvenosti[1] i Uvjet neredundantnosti[2] onda se on naziva ključ relacije. Ukoliko relacija ima više različitih ključeva oni se zajedno nazivaju kandidati za ključ.

Konačno, odabire se jedan ključ koji se naziva primarni ključ relacije. On ispunjava kriterij da „…je jednoj vrijednosti atributa u relaciji pridružena samo jedna vrijednost ostalih atributa relacije, odnosno da su svi neključni atributi funkcijski ovisni o ključu.“[1]. Primarni ključ označava se podcrtavanjem punom linijom.

Ukoliko tip entiteta sadrži atribut koji nije ključ te relacije, ali je ključ u nekoj drugoj relaciji modela tada se on naziva vanjski ključ i podcrtava se isprekidanom linijom.

Treća vrsta ključa je ključ za sortiranje. On služi, kako mu i ime kaže, za sortiranje podataka u tablici.

Bitno je objasniti i da tip entiteta može biti slab. „Slabi tip entiteta (engl. *weak entity*) je tip entiteta koji je na neki način ovisan o nekom drugome tipu entiteta, a ta se ovisnost prikazuje specijalnim tipom među jakim i slabim tipom entiteta.“[1] . Ključ slabog tipa entiteta sastoji se od ključa podtipa i ključa nadtipa.

Još jedna bitna vrsta tipa entiteta je agregirani tip entiteta ili agregacija. Agregacija se opisuje kao „…apstrakcija u kojoj se tip veze između dvaju ili više tipova tretira kao novi tip entiteta.“[1]. Apstrakcija ovisi o brojnosti veze (objašnjeno dalje u tekstu). Ključ u apstrakciji je složen i sastoji se od ključeva dvaju entiteta koja povezuje.

Sada kada su objašnjeni entiteti i njihovi atributi treba pojasniti njihov međusobni odnos. Entiteti su povezani vezama, a definicija veze je „koncept koji predstavlja neku interakciju među entitetima u sustavu, odnosno predstavlja znanje o njihovoj povezanosti.“[1].

Uz veze bitan je i pojam brojnosti. Prema [1], „Brojnost tipa veze je broj koji kaže koliko entiteta pojedinoga tipa entiteta E_1 sudjeluje (pojavljuje se) u tipu veze V s entitetom iz tipa entiteta E.“[1]. Brojnost se upisuje isključivo na dijagram Entiteti-Veze, a oblika je uređenog para brojeva u kojem prvi broj predstavlja tzv. Donju granicu, a drugi Gornju granicu. Brojnost se upisuje na samu vezu, u blizini entiteta na kojeg se odnosi.

[1] Vrijednost ključa svake n-torke relacije jedinstveno određuje n-torku, to jest, ne postoje dva takva retka u tablici da imaju sve iste vrijednosti svih atributa koji čine ključ.

[2] Ne postoji niti jedan atribut kao dio ključa koji se može izostaviti iz ključa, a da se pritom uvjet jedinstvenosti ne gubi, to jest, ključ je unija minimalnoga broja atributa.

Uz objašnjene glavne pojmove, lakše je shvatiti DEV odabranog sustava.

Krenimo od entiteta. Iako je DOSTAVA, po osjećaju, centralni entitet, započet ćemo od entiteta EVIDENCIJSKI_LIST. On predstavlja zapis dokumenata koji sadrže detaljne informacije o dostavama. Primarni ključ *je Sifra_ev_lista* kao jedinstveni identifikator takvog dokumenta. Ostali atributi su *Datum_ev_lista* te *Broj_dostava_u_danu*. Ovdje se računa ukupan broj dostava koje su zapisane na pojedinom Evidencijskom listu. Ovaj entitet sadrži i atribute koji su u službi vanjskog ključa. One se po pravilu crtanja dijagrama EV ne zapisuju na ovom mjestu. Prvi takav atribut je *Sifra_dost_centra*. To je primarni ključ entiteta DOSTAVNI_CENTAR.

U ovoj se tablici zapisuju podaci o dostavnim centrima koje posjeduje ova služba. Atributi entiteta, uz navedeni primarni ključ, su *Naziv_dost_centra*, koji je najčešće prema najbližem gradu ili mjestu, *Adresa_dost_centra*, *Podrucje_djelovanj*a kao podatak o točnom okrugu u kojem taj centar ima jurisdikciju, *Broj_dostavljaca* te *Broj_vozila* kao podaci o dostavljačima i vozilima koji su dodijeljeni ovom centru. Tablica DOSTAVNI_CENTAR povezana je vezom tipa (1,1)(0,M) s tablicom EVIDENCIJSKI_LIST. Veza s tom brojnošću čita se na način: „Jedan dostavni centar može se nalaziti na najmanje jednom, a najviše mnogo evidencijskih listova. Jedan evidencijski list može sadržavati najmanje jedan, ali i najviše jedan dostavni centar.“ Ovo je specijalan tip veze, kod kojeg nalazimo vanjski ključ te time gore navedeno pravilo o pisanju.

Sljedeći vanjski ključ entiteta EVIDENCIJSKI_LIST je *Broj_dost_vozila*.

U entitetu DOSTAVNO_VOZILO nalazimo atribut i primarni ključ istog imena. Radi se o jedinstvenom broju dodijeljenom svakom vozilu koji se evidentira kroz ovaj stupac tablice. Dalje, imamo atribute *Marka*, kao zapis o proizvođaču vozila, *Model*, *Godina* (proizvodnje), *Datum_regist* kao datum posljednje registracije te *Datum_servisa* kao datum zadnjeg servisa na vozilu. I ovdje nalazimo vezu (1,1)(0,M) te tako i pripadno pravilo o pisanju ključa. Svrha ove tablice je evidencija vozila koje tvrtka posjeduje.

Zadnji vanjski ključ koji se pojavljuje u entitetu EVIDENCIJSKI_LIST je *Sifra_dostavljaca*.

Sada znamo da je veza s entitetom DOSTAVLJAC, u tom slučaju, sigurno tipa (1,1)(0,M). Naravno dostavna služba ne može postojati bez dostavljača, pa je ovo još jedna prirodno prisutna tablica. Atributi ovog entiteta, uz primarni ključ, su, naravno, *Ime_dostavljaca* i *Prezime_dostavljaca*, *Kontakt* (broj telefona), *Status_dostavljaca* (u smislu je li aktivan, trenutno na godišnjem odmoru, bolovanju itd.), *Ocjena_za_mjesec* (brojčana ocjena koja može biti razlog povećanja plaće), *Broj_sati* kao ukupan broj redovnih sati preko tjedna, *Broj_sati_ned_blagdan* kao zapis sati koji se plaćaju po većoj satnici te konačno *Ukupno_placa*, stupac u tablici u koji se upisuje ukupni izračun plaće određenog dostavljača.

Kada smo se dotaknuli dostavljača i plaće, dobro je odmah objasniti i entitet SATNICA te agregaciju PRIMA. Entitet SATNICA predstavlja zapis o različitim satnicama koje tvrtka postavlja kao faktor pri računanju plaće pojedinog djelatnika. Tako su atributi redom *Sifra_satnice* (primarni ključ tablice), *Naziv_satnice* (za lakše razlikovanje), naravno *Iznos* te atribut *Veca_manja* (kao signal radi li se o standardnoj ili uvećanoj satnici). Kako bi veza između tablica SATNICA i DOSTAVLJAC bila (1,M)(1,M) – „jedan dostavljač može primati najmanje jednu, a najviše mnogo satnica te jednu satnicu može dobivati najmanje jedan, a najviše mnogo dostavljača“- dolazimo do potrebe za agregacijom. Nazivamo je PRIMA te je u dijagramu EV označavamo sa dijamantnim oblikom unutar četverokuta.

Brojnost se u ovom slučaju ne zapisuje na veze, međutim ona je, prema pravilima pisanja dijagrama, sa strane agregacije (1,1), a sa strana povezanih entiteta (0,M).

Atributi ove agregacije su *Sifra_dostavljaca i Sifra_satnice* (ne upisuju se na dijagram, a zajedno tvore složeni primarni ključ agregacije) te *Iznos_ukupno* kao stupac tablice u koji se zapisuje upravo rezultat izračuna plaće (izračun će biti definiran na način da će se po *defaultu* množiti vrijednost iz stupca *Broj_sati* za određenog dostavljača i vrijednost stupca *Iznos* za određenu satnicu. Ako satnica ima vrijednost „Veca“ u stupcu *Veca_manja*, tada će se množiti vrijednost iz stupca *Broj_sati_ned_blagdan* i iznos satnice).

Sada možemo preći na entitet DOSTAVA. Na dijagramu EV primjećujemo dupli četverokut. To je oznaka slabog tipa entiteta tj. prema tome znamo da uklanjanjem entiteta EVIDENCIJSKI_LIST, automatski se uklanja i DOSTAVA, jer je zapis o dostavama ovisan o zapisu evidencijskih listova. To nam pokazuje strelica, umjesto obične oznake veze. U ovom slučaju, također se ne upisuju brojnosti na dijagram. Atributi ovog slabog tipa entiteta su *Sifra_dostave*, *Redni_broj_u_danu* (služi lakšem pregledu u samom evidencijskom listu), *Posiljatelj* (stupac u koji će se zapisati podaci o pošiljatelju dostave), *Tezina* (težina pošiljke koja će se koristiti pri izračunu cijene), *Vrijednost_u_kn* (Potrebno pri odabiru usluge, razine čuvanja itd. Biti će detaljnije objašnjeno kod entiteta USLUGA) te *Cijena_dostave_ukupno* (stupac u koji će se zapisati rezultat izračuna troška dostave. Izračun će biti u smislu da se množe težina predmeta i cijena po gramu definirana u tablici TIP te se tom umnošku zbraja cijena usluge iz tablice USLUGA).

I ovdje nalazimo vanjske ključeve tj. atribute koji se po pravilu neće zapisati pod ovaj entitet. Sukladno tome, može se zaključiti da je veza sa sljedećih 5 entiteta tipa (1,1)(1,M). Svaki entitet i veza biti će posebno razjašnjeni.

Prvi atribut, a ujedno i vanjski ključ je *OIB_primatelja*. To je naziv atributa i primarnog ključa tablice PRIMATELJ. Ona zapisuje podatke o primatelju kojem je dostava upućena. Uz standardne atribute (*Ime_primatelja*, *Prezime_primatelja, Adresa_primatelja* te *Kontakt*) nalazimo i atribut *Crna_lista*. Prema svom zapisu, on će korisniku reći je li određeni primatelj stavljen na crnu listu tj. je li prema njemu obustavljena dostava.

Sljedeći vanjski ključ tablice DOSTAVA je *Sifra_tipa*. On je povezuje s tablicom TIP. Radi se o zapisima raznih tipova dostave (pismo, paket…). Zato su atributi, uz primarni ključ, redom *Naziv_tipa*, te *Cijena_po_g*. Ovaj zadnji atribut važan je u izračunu cijene dostave jer se upravo on množi sa težinom dostavljanog predmeta.

Dalje imamo atribut *Sifra_usluge*, primarni ključ tablice USLUGA. Sve dostave nisu jednake. Ovisno o vrijednosti, hitnoći ili nekom drugom faktoru, pošiljatelj može izabrati uslugu koja mu se čini potrebnom. Atributi ovog entiteta su: *Naziv_usluge*, *Opis_usluge*, *Razina_sigurnosti* (brojevna vrijednost koja pokazuje službi prioritet u dostavi), *Garantirani_broj_dana* (zamišljeno je da što je odabrana usluga skuplja, obrnuto proporcionalno se smanjuje broj dana u kojima dostava mora stići do odredišta), *Cijena_usluge* (iznos koji se dodaje u izračunu troška dostave) te *Minimalni_potrebni_iznos* (govori pošiljatelju koliki mora biti minimalni iznos vrijednosti da bi se ova usluga mogla iskoristiti).

Zatim slijedi vanjski ključ *Sifra_stanja*. Radi se o primarnom ključu tablice STANJE_POSILJKE. Ova tablica, uz atribut Sifra_stanja, ima samo još jedan atribut, a to je *Opis_stanja*. Svrha ove tablice je zamišljena da sadrži zapise o mogućim stanjima pošiljke (originalno, oštećeno, izgubljeno…). Kako služba svakog dana radi sa velikim brojem dostava, ove su situacije moguće pa je potrebno imati i zapis o njima.

Posljednji vanjski ključ tablice DOSTAVA je *Sifra_statusa*. On dolazi iz tablice STATUS_DOSTAVE. Svrha ove tablice je zapis o statusima dostave tj. je li ona u pripremi, u tranzitu, dostavljena… Uz atribut *Sifra_statusa* tu je i *Opis_statusa*.

Na taj način objašnjen je Dijagram Entiteti-Veze ovog sustava. Sada slijedi izrada same poslovne aplikacije.

2.2. Izrada rječnika

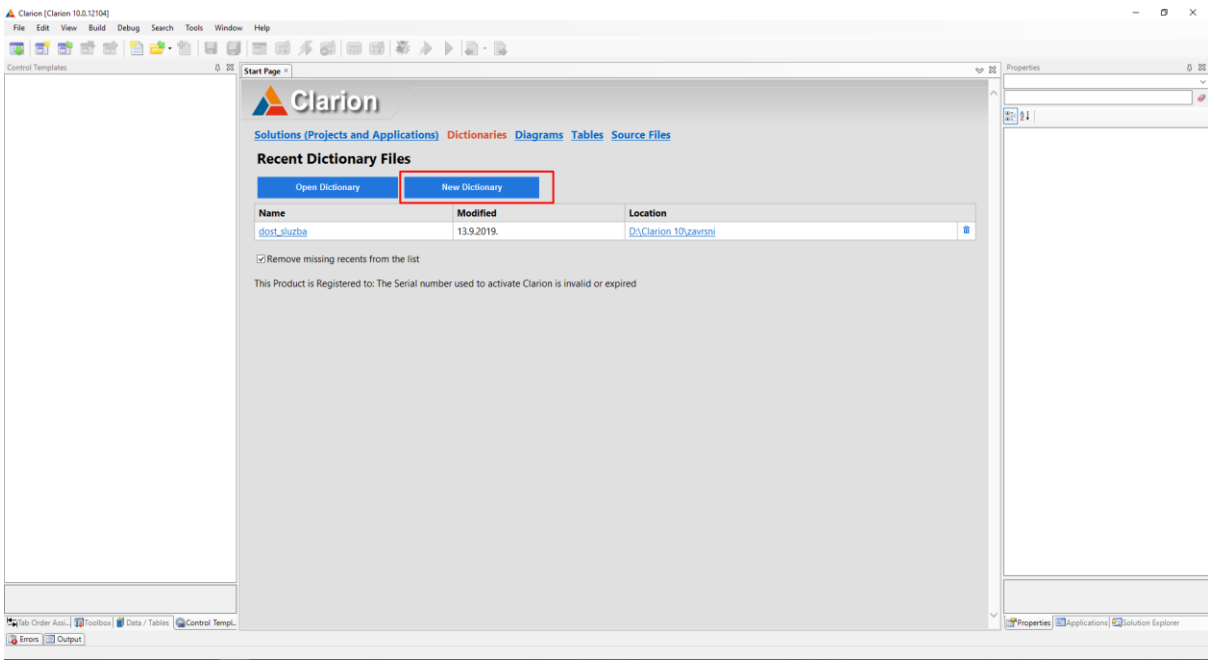
Rječnik je temelj izrade aplikacije u Clarionu. On je uputa softveru kako strukturirati bazu, na koji način oformiti tablice, tj. kako oblikovati podatke koji će se zapisivati u različitim stupcima. Tu su još definirani ključevi tablica te finalno veze među njima, uz njihove brojnosti.

Pri izradi rječnika korisnik prolazi nekoliko koraka: 1) Izrađuje novu .dct datoteku, 2) stvara tablice, 3) formira stupce i njihova svojstva, 4) odabire primarne, vanjske i ključ za sortiranje i finalno, 5)postavlja veze među tablicama.

Svaki od ovih koraka ćemo u nastavku detaljno opisati na našoj aplikaciji i potkrijepiti slikama zaslona za bolje shvaćanje.

2.2.1. Nova .dct datoteka

Pri pokretanju alata Clarion na početnom zaslonu prikazuju se projekti koje možemo kreirati. Ponuđeni su *Solutions*, *Dictionaries*, *Diagrams*, *Tables* te *Source files*. Odabiremo *Dictionaries* te tipku *New dictionary*.



Slika 2. Clarionov početni zaslon

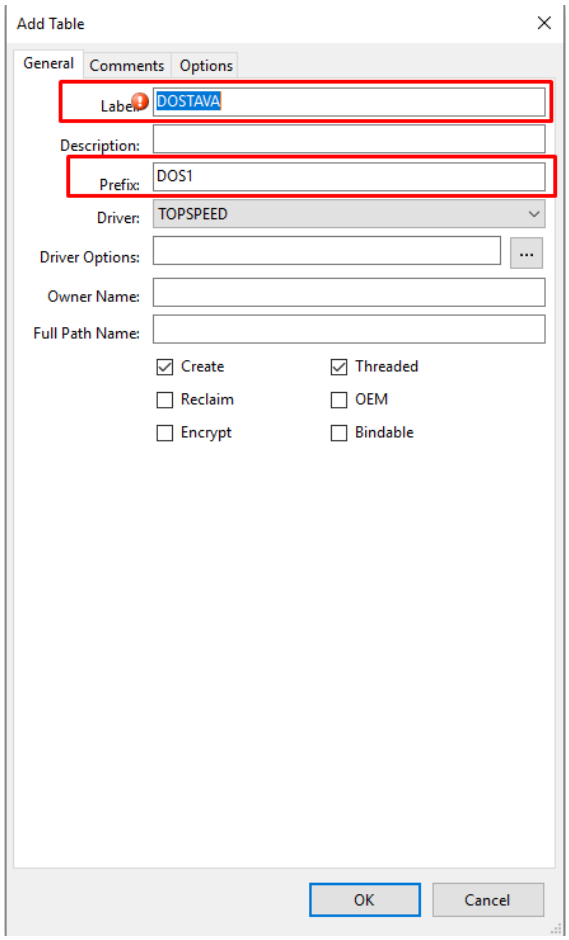
Nakon odabira imena i mjesta na kojem ćemo spremiti rječnik (.dct datoteku) otvara se Clarionovo sučelje za uređivanje rječnika tzv. *Dictionary Editor*.

2.2.2. Dodavanje tablica

U samom *Editoru* s lijeve stane nalazi se *Dictionary Explorer* koji prikazuje trenutno prazne mape: *Pools, Globals* i *Tables*. Odabiremo mapu *Tables* desnim klikom te u izborniku biramo *Add Table.*

U novootvorenom prozoru možemo postaviti osnovna svojstva tablice. Odabrat ćemo samo ime tablice te prefiks. Ime tablice piše se velikim slovima poput naziva tipa entiteta u shemi relacijskog modela.

Prefiks sadrži 3 do 4 slova i služi kao identifikator tablice kojoj određeni stupac pripada. Ostale postavke ostavljamo kako su po *defaultu* postavljene i biramo OK. Na isti način oblikujemo sve potrebne tablice.



Slika 3. ***Add Table***

2.2.3. Dodavanje stupaca tablicama te određivanje ključeva

Sada kada rječnik ima stvorene tablice treba im dodati stupce. Ovaj korak ima veliku važnost jer točnim odabirom svojstava stupca, budućoj aplikaciji daje se uputa kakve podatke treba prihvaćati pojedini redak. Pogrešnim postavljanjem određenog parametra može se narušiti pravilan rad, ali i samo kompajliranje aplikacije, te se zahtjeva posebna pažnja u formiranju stupaca.

Same stupce stvaramo na sljedeći način. Za početak biramo tablicu koju ćemo prvu uređivati. Dobra je praksa krenuti s jednostavnim tablicama, odnosno onima koje nemaju niti složene primarne ključeve niti vanjske ključeve.

Tek kad kreiramo sve stupce u tablici, možemo preći na određivanje primarnog ključa te vanjskog ili ključa za sortiranje, ako postoje za tu tablicu.

2.2.3.1. Tablica DOSTAVNI_CENTAR

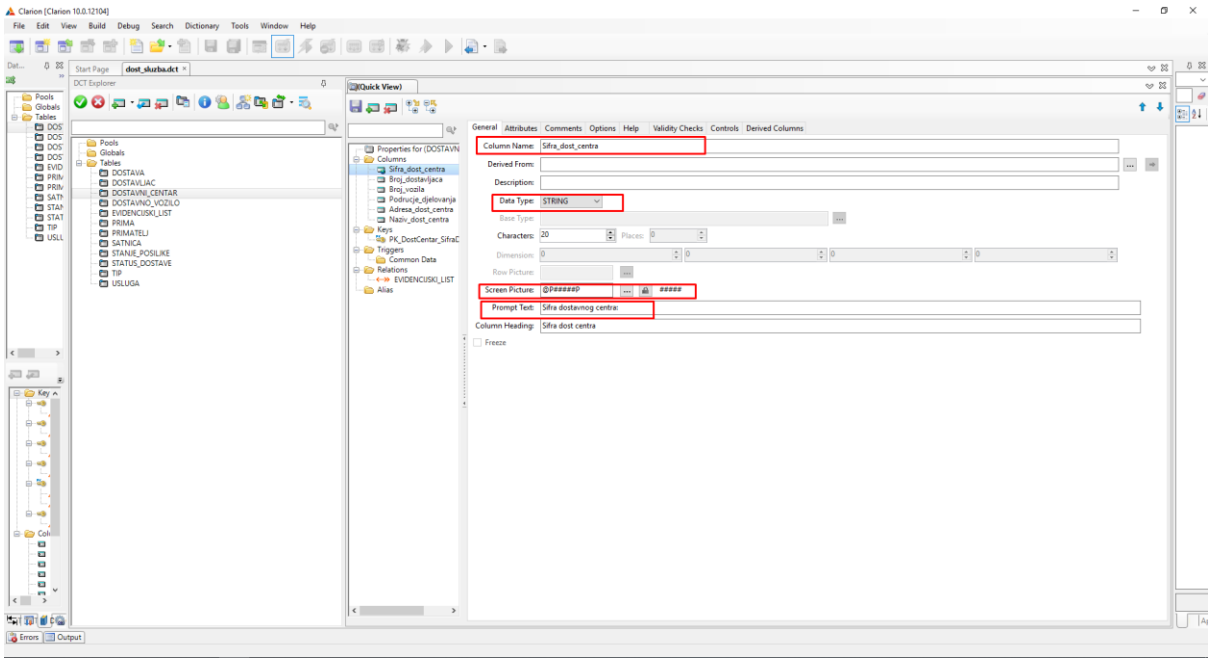
Po odabiru željene tablice, s desne strane otvara se *Quick View* kartica koja prikazuje Clarionove mape za tu tablicu. U početku su one prazne pa desnom tipkom odabiremo mapu *Columns* i biramo tipku *Add Column*.

U prvoj kartici pod imenom *General* upisujemo ime stupca (*Sifra_dost_centra*) pazeći da razmaci u imenu budu prikazani donjom crtom. U polju *Data Type* postavljamo tip podataka da bude STRING, a niže u polju *Screen Picture* postavljamo format u kojem će podatak biti prihvaćen, zapisan i prikazan. U ovom slučaju format je @P#####P što označava *pattern*, odnosno da podatak mora biti 5-ero znamenkasti broj, a u suprotnom se neće prihvatiti. Još izmjenjujemo polje *Prompt Text* (određuje naslov stupca koji će se prikazivati u aplikaciji) na način da brišemo donju crtu i pišemo cijele riječi.

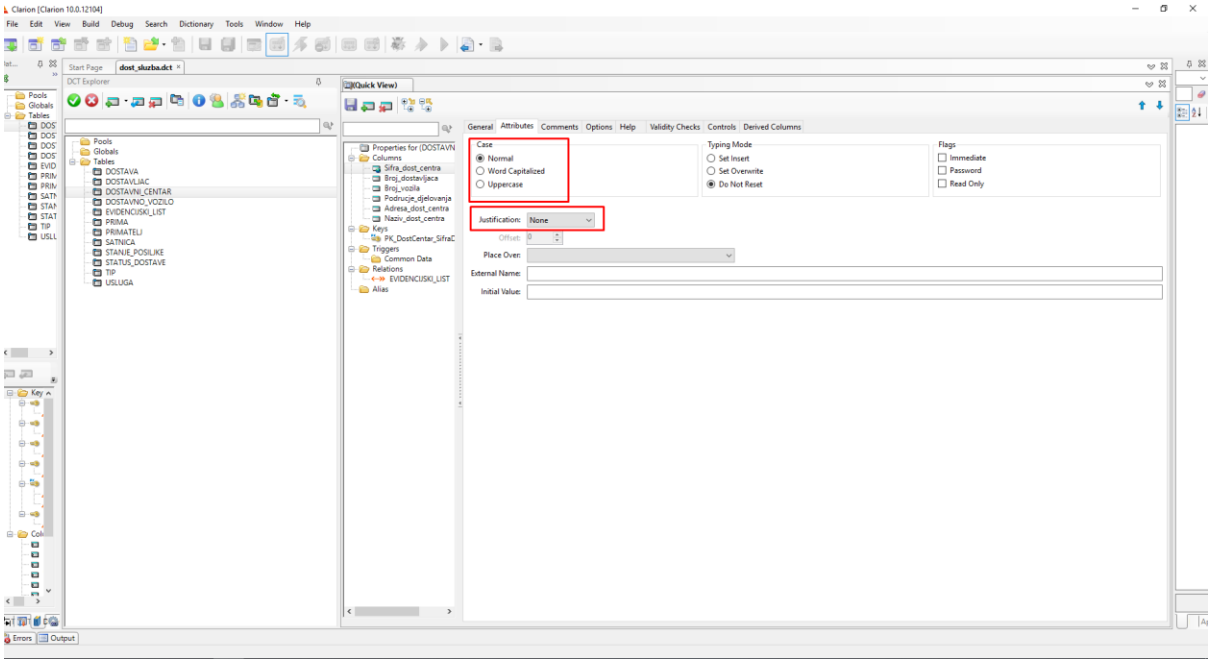
U kartici *Attributes* ostavljamo *Case* pod *Normal* jer je brojevni zapis pa nema smisla postavljati veliko slovo, te niže pod *Justification* odabiremo *None*, ponovno jer se radi o broju.

Kartica *Help* omogućuje nam upis teksta koji će se prikazati korisniku ako zadrži strelicu miša nad ovim poljem. Ova opcija je korisna za pojašnjenje tipa podatka koji se traži.

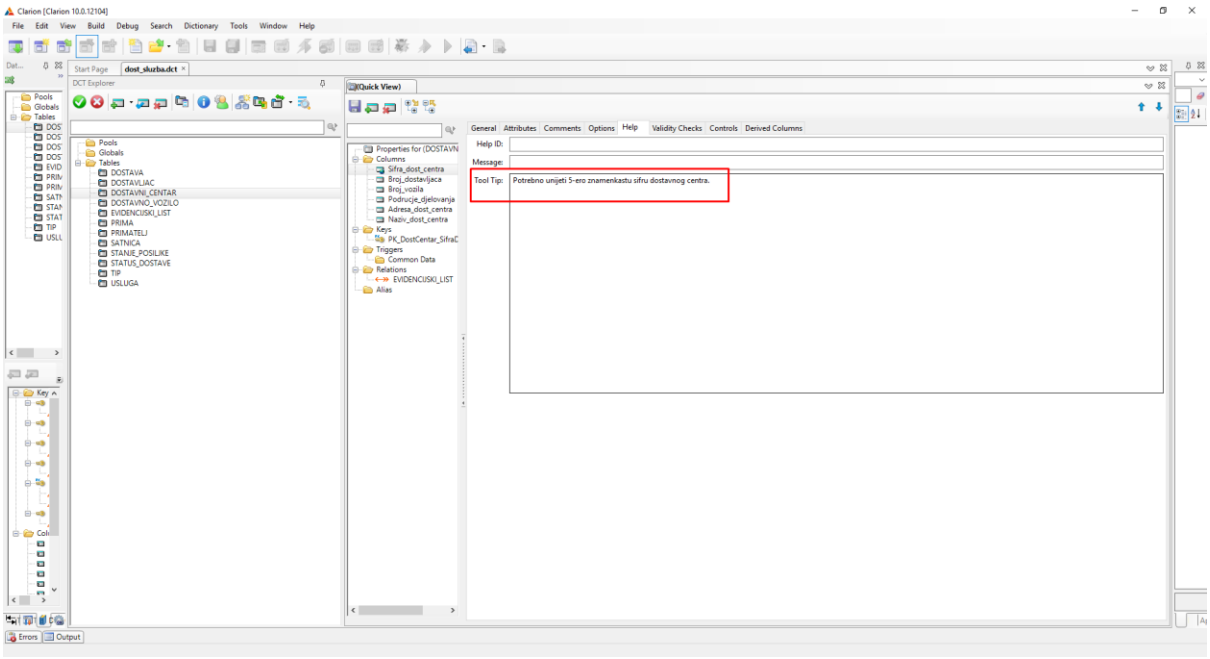
Konačno, na predzadnjoj kartici *Validity Checks* odabiremo opciju *Cannot be Zero or Blank*. Kako znamo da će nam upravo ovaj stupac biti primarni ključ tablice, ova nam je opcija kritična jer govori aplikaciji da zapis u ovoj tablici ne smije imati nulu ili prazan znak u ovom stupcu. Biramo OK i spremamo podatke o stupcu.



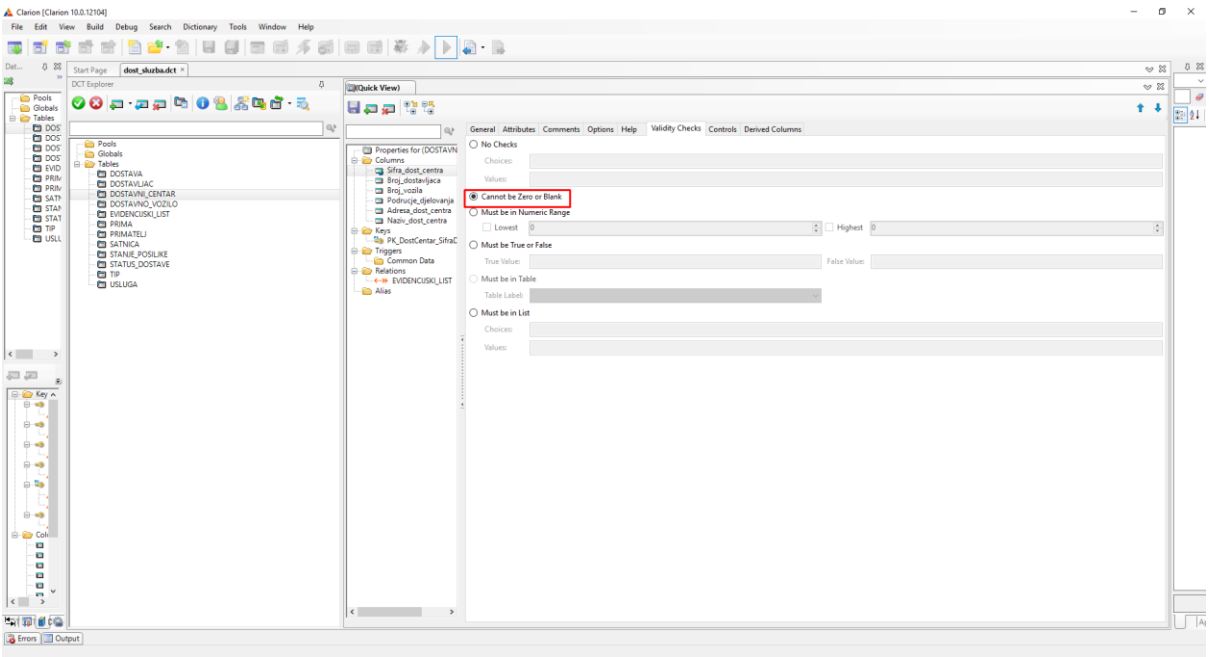
Slika 4. Kartica *General*



Slika 5. Kartica *Attributes*



Slika 6. Kartica *Help*

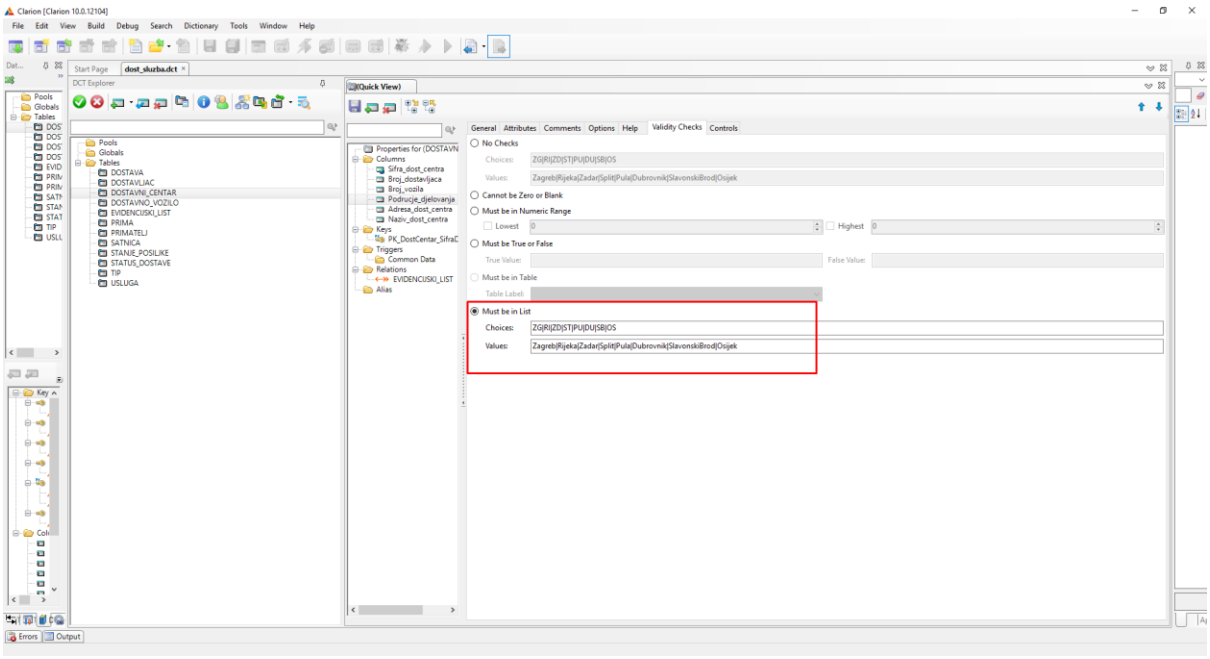


Slika 7. Kartica *Validity Checks*

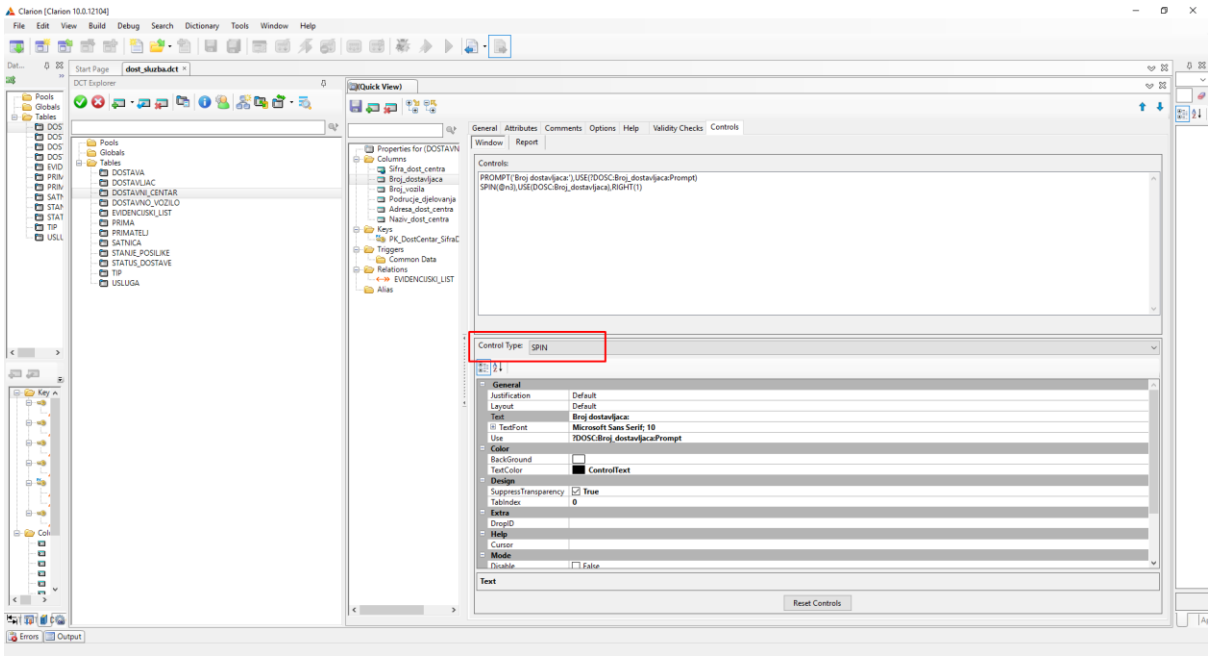
Na isti način kreiramo preostale stupce, vodeći računa o tipu podatka koji će biti spremljen te o veličini. Za stupac *Naziv_dost_centra* tip podataka je CSTRING, a u polju *Screen Picture* format postavljamo na @s49 odnosno *string* sa 49 mjesta za znakove. Istu stvar možemo postići promjenom polja *Characters* na 50.

Kod stupca *Podrucje_djelovanja* uz dosad navedene postavke koristimo još jednu Clarionovu mogućnost. Radi se o opciji *Must be in List* koja se nalazi u kartici *Validity Checks*. Sastoji se od polja *Choices* i *Values*. U prvo polje upisuju se vrijednosti koje će korisniku biti prikazane u obliku *radio gumba*, a u drugo polje vrijednosti koje će biti upisane u tablicu. Dakle funkcionalnost ove opcije je da korisniku ne dozvoli upis proizvoljne vrijednosti, nego samo odabir od ponuđenog. Tako smo ovdje u *Choises* upisali kratice hrvatskih gradova, a u *Values* pune nazive. Vrijednosti se prema sintaktičkim pravilima odvajaju okomitom crtom (|).

U stupcima *Broj_dostavljaca* i *Broj_vozila*, u zadnjoj kartici *Controls,* u polju *Control Type,* promijenili smo vrijednost iz *ENTRY* u *SPIN* tako da se u to polje, uz ručni unos, vrijednost može mijenjati koristeći strelice prikazane na samom polju.



Slika 8. ***Must be in List***



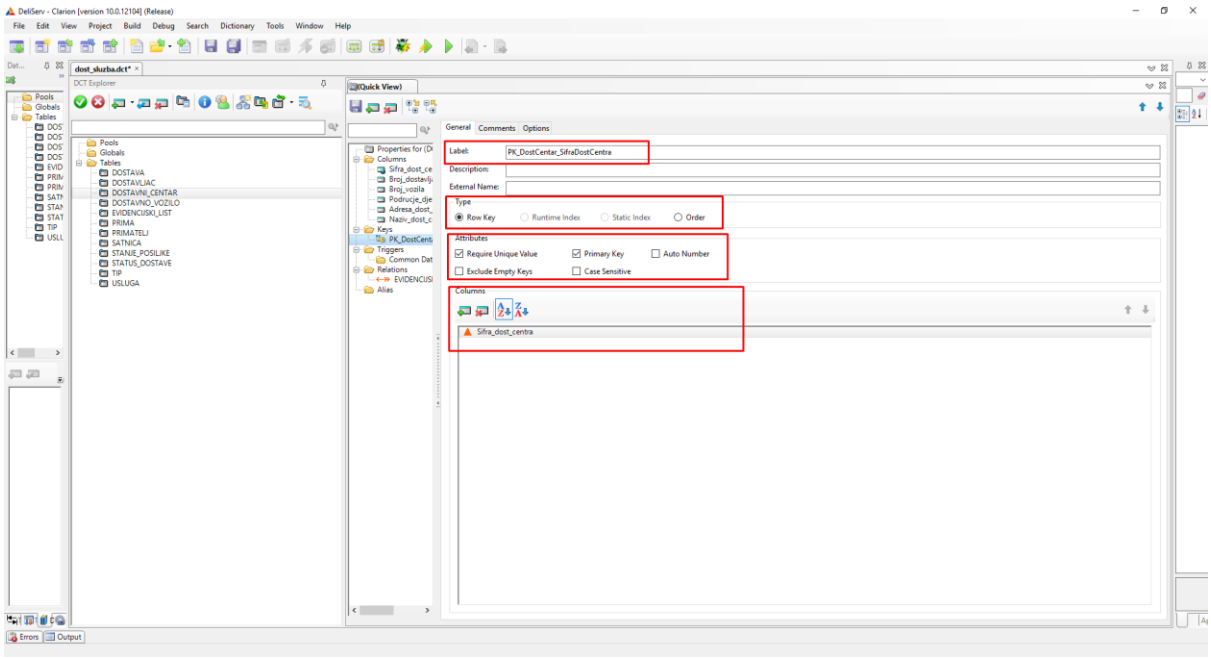
Slika 9. Kartica *Controls*

Sada kada imamo sve stupce u ovoj tablici, možemo postaviti *stupac Sifra_dost_centra* za primarni ključ. To radimo na način da u *Quick View* desnom tipkom biramo mapu *Keys* i odabiremo *Add Key*.

U novom prozoru, u polju *Label* ključu dajemo ime. Ponovno po dobroj praksi imena ključeva daju se u formatu *PK_Ime_tablice_Ime_stupca*. Umjesto PK (*Primary Key*), za vanjski ključ piše se VK ili FK (*Foreign Key*) te za ključ za sortiranje kratica SK (*Sort Key*).

Dakle, u ovom slučaju, pišemo *PK_DostCentar_SifraDostCentra*. Niže, u polju *Type* biramo *radio* gumb *Row Key* jer se radi o ključu, zatim u polju *Attributes* označavamo *checkbox Require Unique Value* te *checkbox Primary key* . Ova dva *checkboxa* signaliziraju da se radi o primarnom ključu te se koriste samo za njegovu izradu.

Finalno, u polju *Columns*, pritiskom na ikonu „+“, dodajemo stupac za koji izrađujemo ključ tj. *Sifra_dost_centra*. Biramo OK i spremamo kreirano.

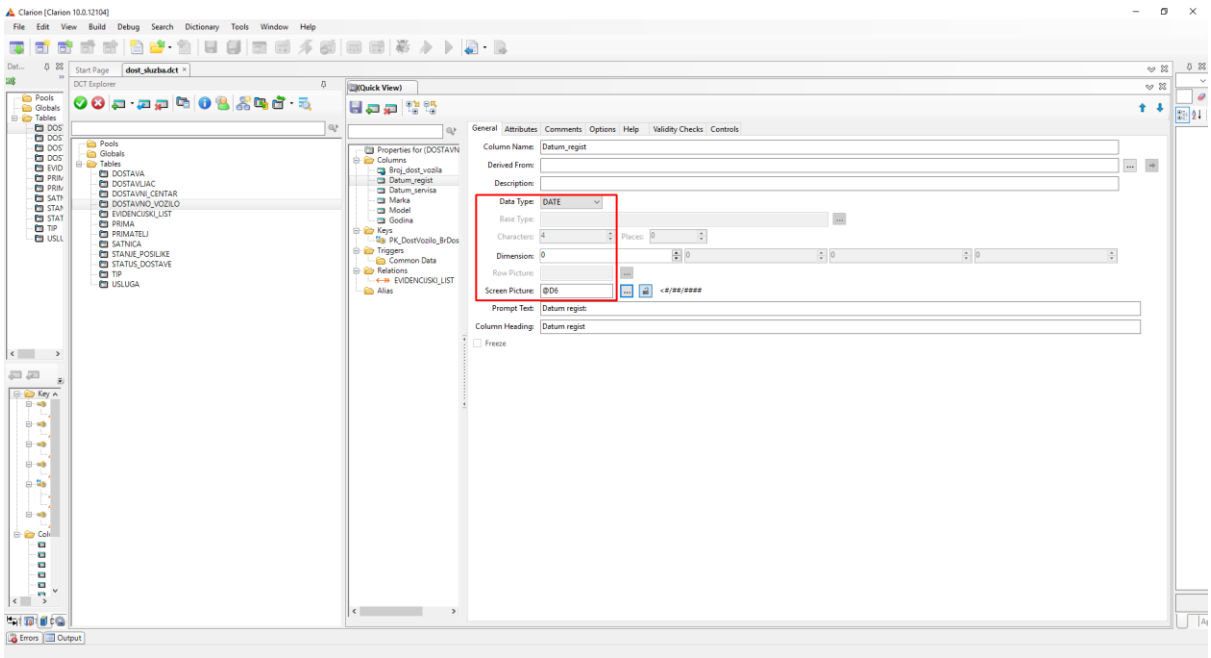


Slika 10. Dodavanje primarnog ključa

2.2.3.2. Tablica DOSTAVNO_VOZILO

Na isti način kao prethodnu tablicu, kreirali smo stupce za tablicu DOSTAVNO_VOZILO. Ovdje je važno naglasiti da smo za stupce *Datum_regist* i *Datum_servisa* koristili tip podataka DATE, a format je @D6, gdje broj 6 predstavlja oblik datuma u kojem su dan, mjesec i godina odvojeni kosom crtom.

Ponovno stvaramo novi ključ, namještamo postavke da bude primarni, biramo stupac Broj_dost_vozila, spremamo kreirano.



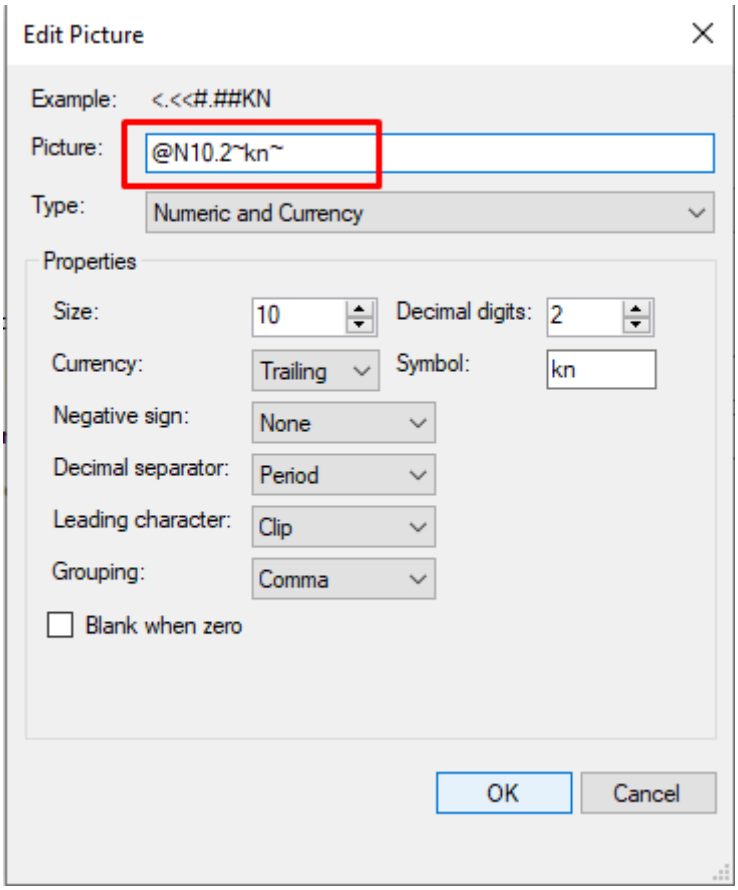
Slika 11. Tip podataka DATE

2.2.3.3. Tablica DOSTAVLJAC

Ista procedura se nastavlja dalje. Stupac Sifra_dostavljaca bit će ključ tablice DOSTAVLJAC, pa koristimo opciju *CBZOB*. Ostali stupci formiraju se prema tipovima podataka koji su potrebni. Za stupac Status_dostavljaca koristimo ponovno opciju *Must be in List* jer će tu korisnik putem *radio* gumba moći odabrati je li dostavljač aktivan, na godišnjem odmoru, bolovanju itd.

Tu je bitno pokazati da smo za stupac *Ukupno_placa* koristili format @N10.2~kn~. To znači da je tip podataka SHORT, odnosno numerički, cijeli dio može imati 10 znamenki, a decimalni dvije te nakon broja ide fiksni dio „kn“ koji predstavlja oznaku valute.

Kreiramo primarni ključ i nastavljam dalje.

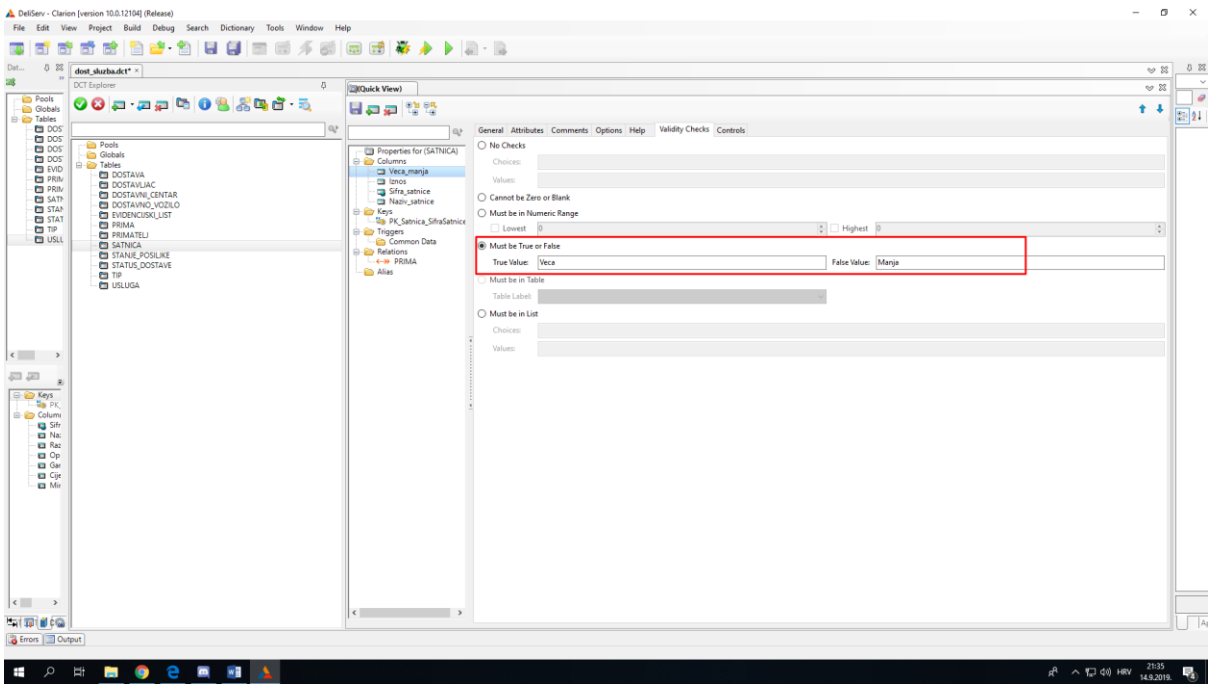


Slika 12. Format sa oznakom valute

2.2.3.4. Tablica SATNICA

Po pravilu kreiramo stupac *Sifra_satnice*, budući ključ, te odabiremo postavku *CBZOB*. Nastavljam sa stupcima *Naziv_satnice* i *Iznos* prema njihovim tipovima. Konačno stvaramo stupac *Veca_manja*. On će služiti kao signalna vrijednost aplikaciji pri izračunu plaće (ovaj dio ćemo objasniti u nastavku rada). Kako ovaj stupac može sadržavati samo dvije vrijednosti biramo tip STRING te format @s10, a u kartici *Validity Checks* biramo opciju *Must be True or False* gdje za istinitu vrijednost postavljamo „Veca“, a za laž „Manja“. To znači da će u aplikaciji na određenom mjestu postojati *checkbox* čijim će označavanjem korisnik u tablicu spremiti istinitu vrijednost, a ostavljanjem praznim upisat će lažnu vrijednost. Ponovno, u nastavku će biti jasnije na što se misli.

Kreiramo primarni ključ i nastavljam s radom.



Slika 13. ***Must be True or False***

Tablice PRIMATELJ, TIP, USLUGA, STANJE_POSILJKE i STATUS_DOSTAVE također su jednostavne tj. imaju jednostavan primarni ključ koji smo, nakon izrade tablica prema traženim tipovima podataka, odredili na identičan način.

Sada dodajemo stupce tablicama koje imaju složene ključeve.

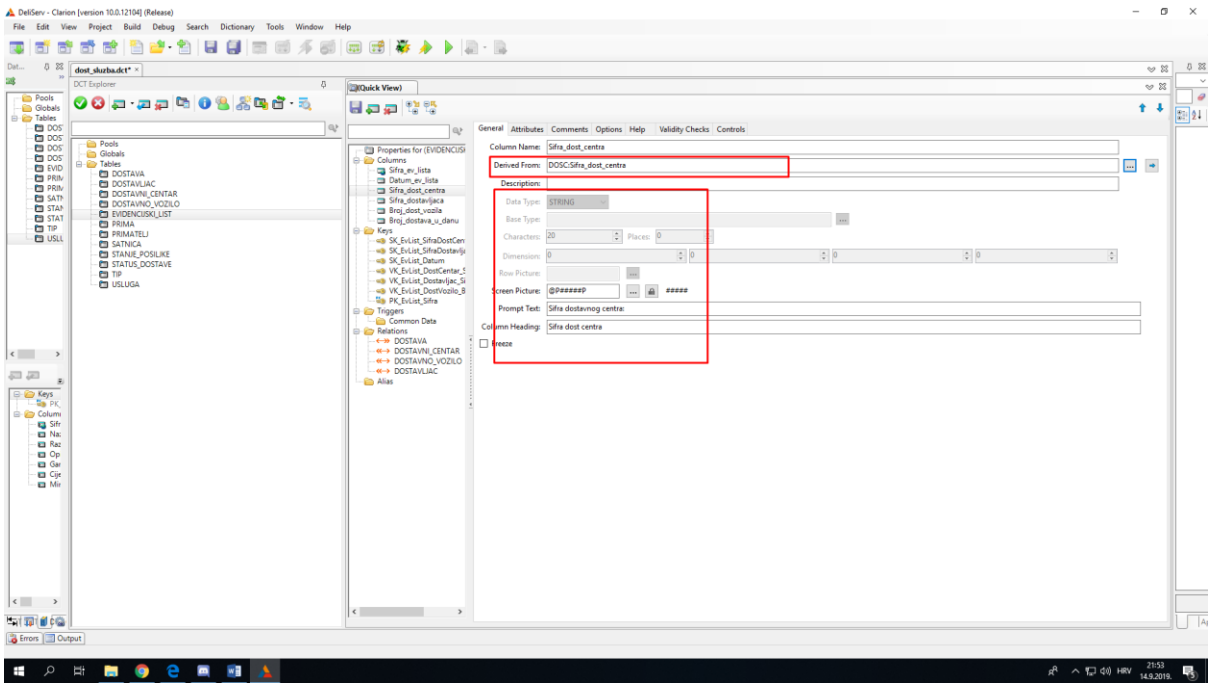
2.2.3.5. Tablica EVIDENCIJSKI_LIST

Ova tablica predstavlja nekakav centar aplikacije. Korisnik koji popunjava njezine stupce, automatski će morati koristiti podatke, odnosno unositi nove, u ostale tablice.

Unosimo stupac *Sifra_ev_lista*. On će biti kasnije primarni ključ. Nakon toga dodajemo stupac Datum_ev_lista koji je tipa DATE. Sada slijedi Sifra_dost_centra. Taj stupac ima isti naziv kao i onaj u tablici DOSTAVNI_CENTAR jer će se podaci „vući“ upravo od tamo. To postavljamo na način da u polju *Derived From*, a preko tipke „ …“ s desne strane, odaberemo tablicu DOSTAVNI_CENTAR i pod njom stupac *Sifra_dost_centra*. Nakon potvrde, tip podataka i format se automatski preuzimaju, kao i postavke u kartici *Validity Checks*.

Sljedeći stupac je Sifra_dostavljaca. Ponavljamo postupak tj. u polje *Derived From* biramo stupac istog imena u tablici DOSTAVLJAC i nakon provjere da su se ostale postavke preuzele spremamo stupac.

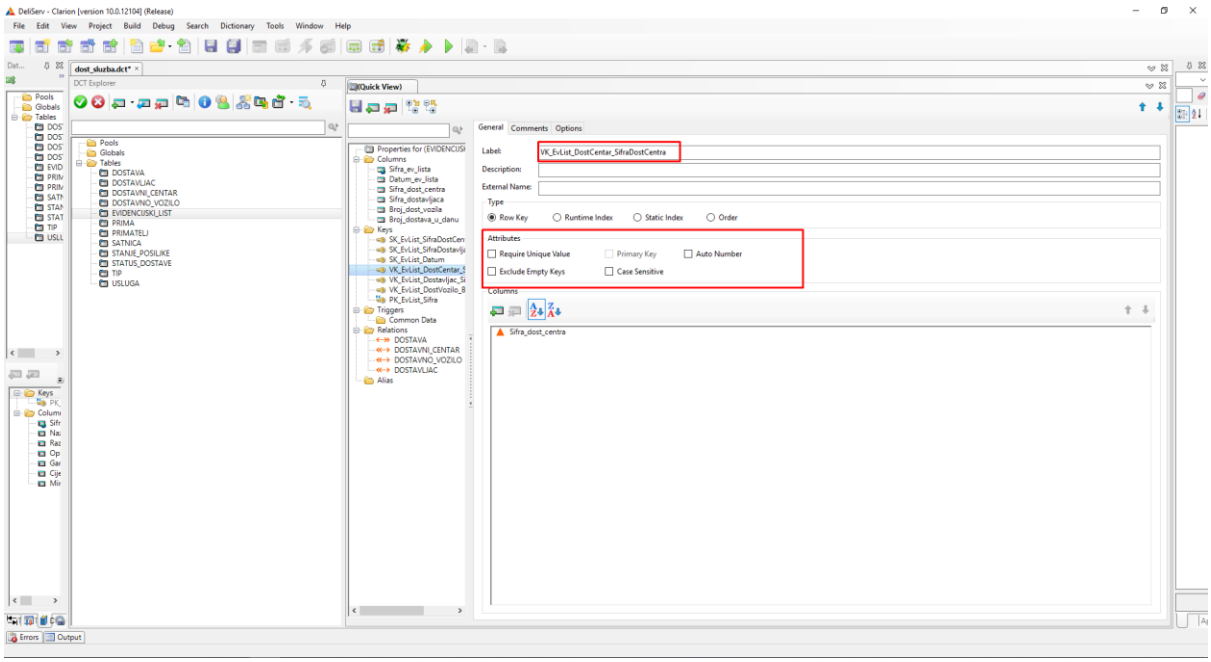
Nastavljam tako i za stupac Broj_dost_vozila te još dodajemo stupac *Broj_dostava_u_danu* koji nije preuzet iz druge tablice, već će služiti za spremanje izračuna broja dostavi u danu. Opširnije slijedi kasnije u radu.



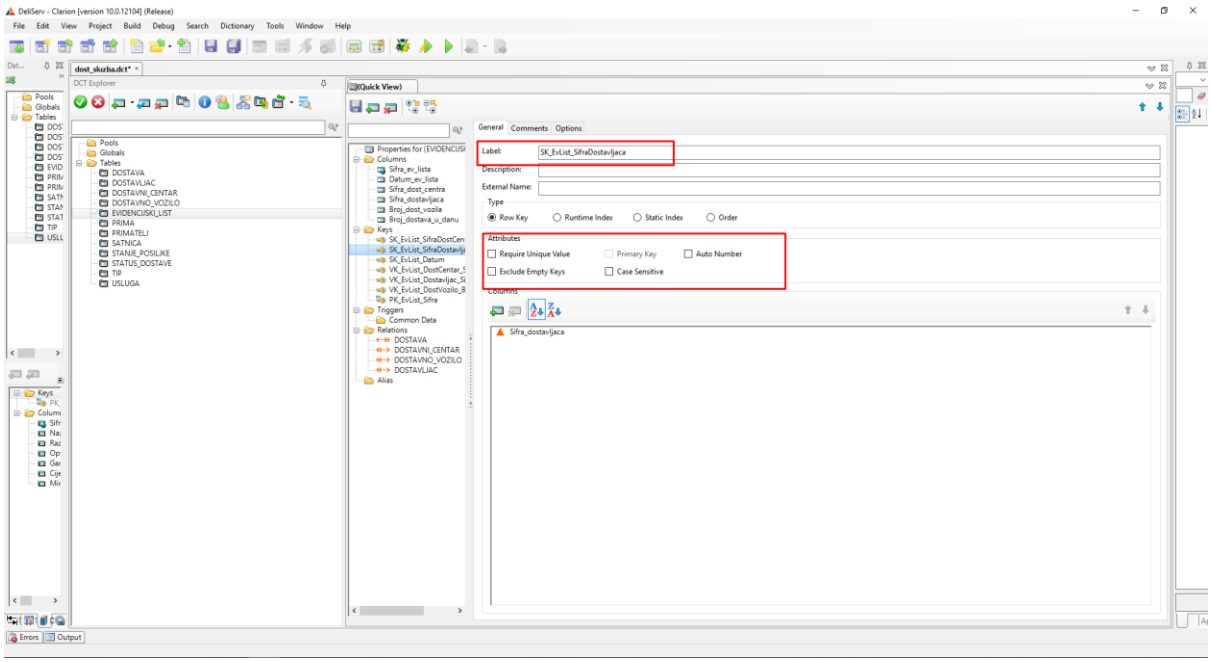
Slika 14. ***Derived From***

Još nam ostaje dodati ključeve. Prvo na prije opisani način dodajemo primarni ključ na stupcu *Sifra_ev_lista*. Potom moramo dodati vanjske ključeve. Procedura za to je ista kao i za primarni ključ, s razlikom što ćemo u prozoru *Add Key* ostaviti prazne *checkboxe Require Unique Value*, a s time i *Primary Key.* Dodajemo potrebni stupac i spremamo ključ.

Finalno još dodajemo ključeve za sortiranje. Oni će nam služiti za sortiranje zapisa u tablici unutar aplikacije. Dodajemo ih na isti način kao i vanjske, tj. ostavljamo prazne *checkboxe* iz grupe *Attributes* i biramo stupac po kojem ćemo sortirati. Izrađujemo onoliko različitih ključeva koliko ćemo imati različitih sortiranja.



Slika 15. Vanjski ključ



Slika 16. Ključ za sortiranje

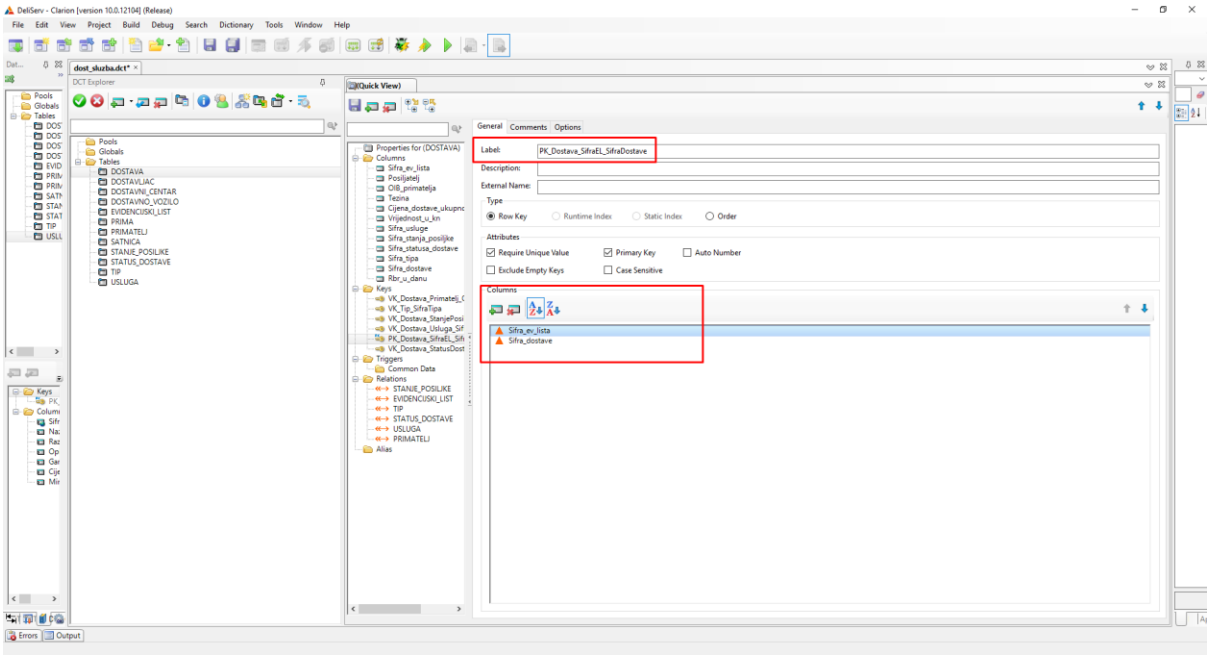
2.2.3.6. Tablica DOSTAVA

U tablici DOSTAVA dodajemo stupce na sličan način, s time da ovdje imamo stupac *Sifra_ev_lista* kojeg preuzimamo iz prethodne tablice te stupac *Sifra_dostave* kojeg formiramo kao budući primarni ključ (*CBZOB*). Dodajemo ostale potrebne stupce, odnosno one preuzete iz drugih tablica te one koje su autohtone ovoj tablici. Bitno je naglasiti da u

stupcu Posiljatelj, pod karticom *Controls,* polje *Control Type* mijenjamo u TEXT. To osigurava da se kasnije, u aplikaciji, korisniku pod ažuriranjem tablice ne prikazuje standardno polje za unos, nego *textbox* polje, tj. polje za upis teksta.

Kad smo završili s dodavanjem stupaca, postavljamo ključeve. Kod primarnog ključa ove tablice bitno je uočiti da pri izradi dodajemo dva stupca, a ne jedan kao u prethodnim tablicama. To je zato što je DOSTAVA slabi tip entiteta, odnosno ovisna je o tablici EVIDENCIJSKI_LIST, pa se tako njezin primarni ključ sastoji od svog primarnog ključa i primarnog ključa nadtipa. Dakle u polje *Columns* dodajemo stupac Sifra_dostave i *Sifra_ev_lista*.

Potom dodajemo potrebne vanjske ključeve po prije opisanoj proceduri i spremamo izrađeno.



Slika 17. Primarni ključ slabog tipa entiteta

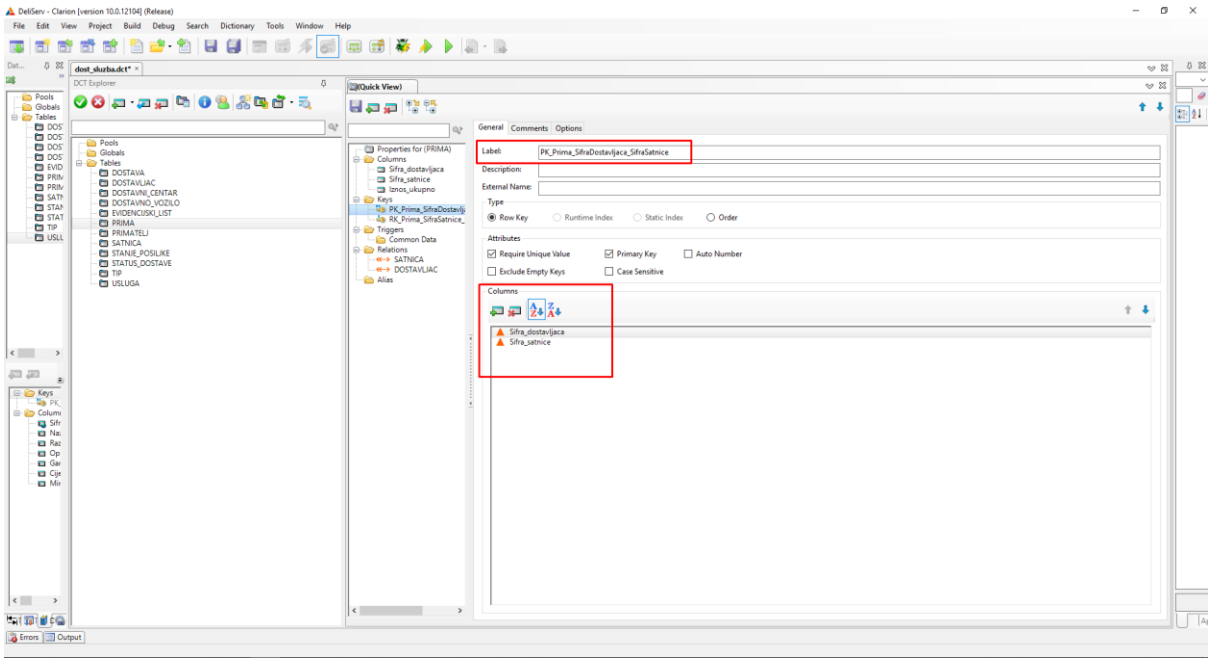
2.2.3.7. Tablica PRIMA

Finalno, ostaje nam još kreirati stupce u tablici PRIMA. Dodajemo stupce *Sifra_dostavljaca* te *Sifra_satnice*. Oboje su „derivirani“ iz istoimenih stupaca pripadajućih tablica. Još dodajemo i stupac Iznos_ukupno u koji će se spremati rezultat prije spomenutog izračuna (vezanog uz vrijednost *checkboxa*).

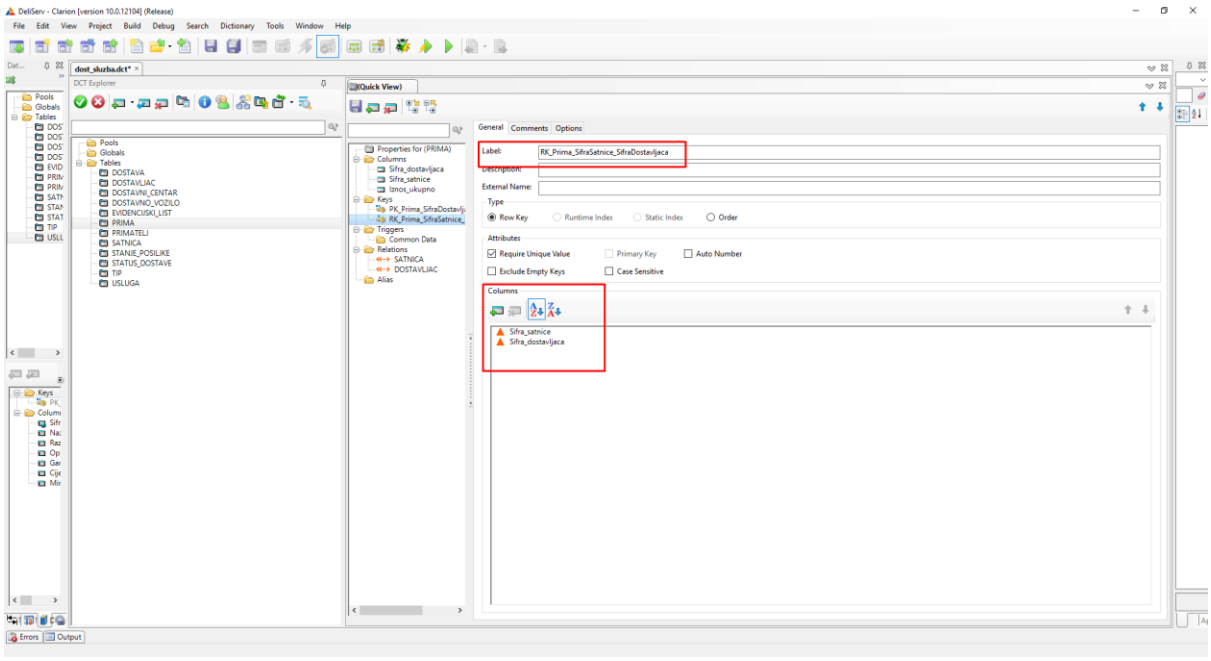
Što se tiče ključeva, PRIMA je agregirani tip atributa tj. agregacija pa se njezin primarni ključ sastoji od primarnih ključeva tablica koje povezuje. To znači da u polje *Columns* dodajemo upravo stupce *Sifra_dostavljaca* i *Sifra_satnice*.

Ovdje je potrebno dodati još jednu vrstu ključa. Radi se o rezervnom ključu koji se dodaje jedino u agregaciji. On se sastoji od istih stupaca kao i primarni ključ, ali u obrnutom redoslijedu. Na takav način se i imenuje.

Spremamo kreirano.



Slika 18. Primarni ključ agregacije



Slika 19. Rezervni ključ agregacije

2.2.4. Stvaranje veza

Na kraju potrebno je još stvoriti veze između tablice i odrediti njihove brojnosti. Ponovno, po dobroj praksi, krećemo od tablica koje su najviše povezane s ostalima jer na taj način „pokupimo“ ostale tablice i prije smo gotovi. Dakle, u našem slučaju, krenut ćemo od tablice EVIDENCIJSKI LIST.

2.2.4.1. Veze s tablicom EVIDENCIJSKI_LIST

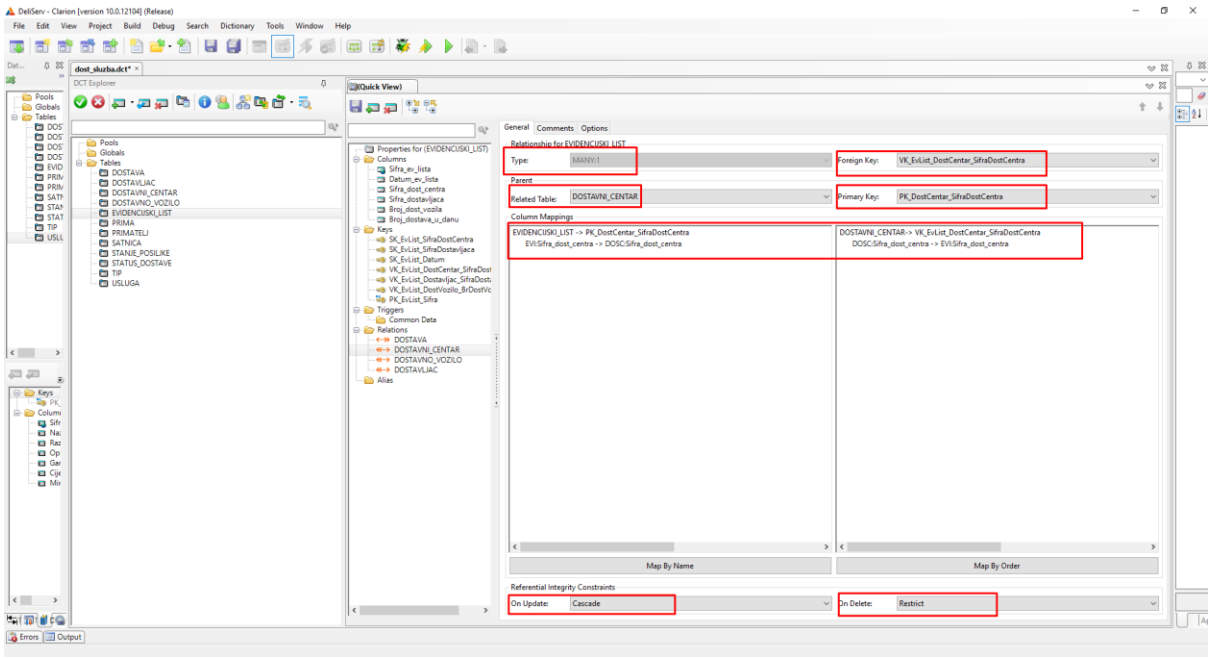
Veze se kao i prethodni zadaci kreiraju kroz *Quick View*. Desnim klikom miša biramo mapu *Relations* i odabiremo *Add Relationship*.

Uzimamo za početak vezu sa tablicom DOSTAVNI_CENTAR. U prvom polju pod nazivom *Type* biramo MANY:1. To znači da je u vezi između tablica EVIDENCIJSKI_LIST i DOSTAVNI_CENTAR sa strane tablice iz koje izrađujemo vezu (EVIDENCIJSKI_LIST) brojnost veze po gornjoj granici 1. U polje *Foreign Key* postavljamo vanjski ključ za tablicu DOSTAVNI CENTAR. Dalje u polju *Related Table* odabiremo samo tablicu DOSTAVNI CENTAR te iza toga njezin primarni ključ u polju *Primary Key*.

Polje *Column Mappings* prikazuje spajanje samih stupaca iz dviju povezanih tablica. Primarni ključ jedne tablice povezuje se s istoimenim vanjskim ključem druge tablice. Samo povezivanje možemo obaviti pritiskom na tipku *Map by Name* (stupci će se povezati prema imenu) ili *Map by Order* (stupci se povezuju redoslijedno kako su prikazani).

Na kraju još ostaje podesiti tzv. *Referential Integrity Constraints*. Radi se o postavkama koje će *Application Generatoru* reći kako da reagira na promjenu ili brisanje vrijednosti primarnog ključa. Postavljamo polje *On Update* na *Cascade*, a polje *On Delete* na *Restrict*. To znači da će generirani kod pri promjeni određene vrijednosti primarnog ključa obaviti promjenu i na svim vanjskim ključevima s istom vrijednosti, dok će kod pokušaja brisanja vrijednosti primarnog ključa generirani kod tražiti istu vrijednost i ako je pronađe u vanjskim ključevima, neće dozvoliti brisanje.

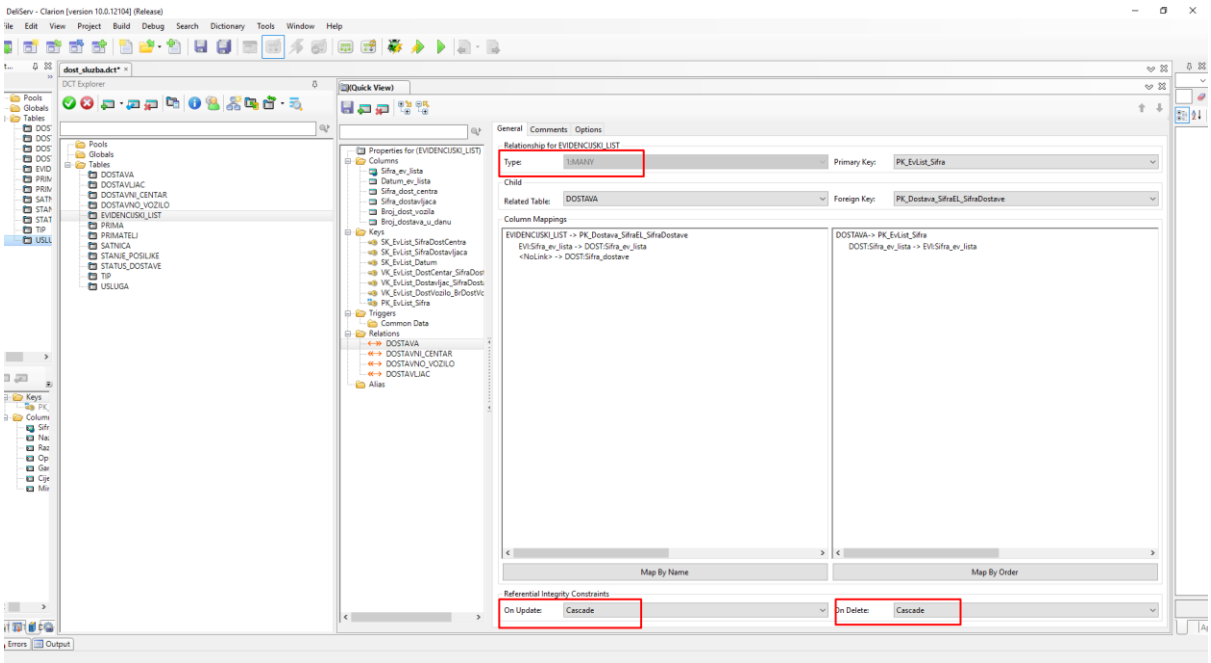
Općenito, kod veza između jakih tipova entiteta, polja se postavljaju na *On Update: Cascade* i *On Delete: Restrict.* S druge strane, kod veza jaki tip – slabi tip te jaki tip – agregacija, oba polja postavljaju se na *Cascade*.



Slika 20. ***Add Relationship***

Na isti način kreiramo vezu sa tablicama DOSTAVNO_VOZILO i DOSTAVLJAČ.

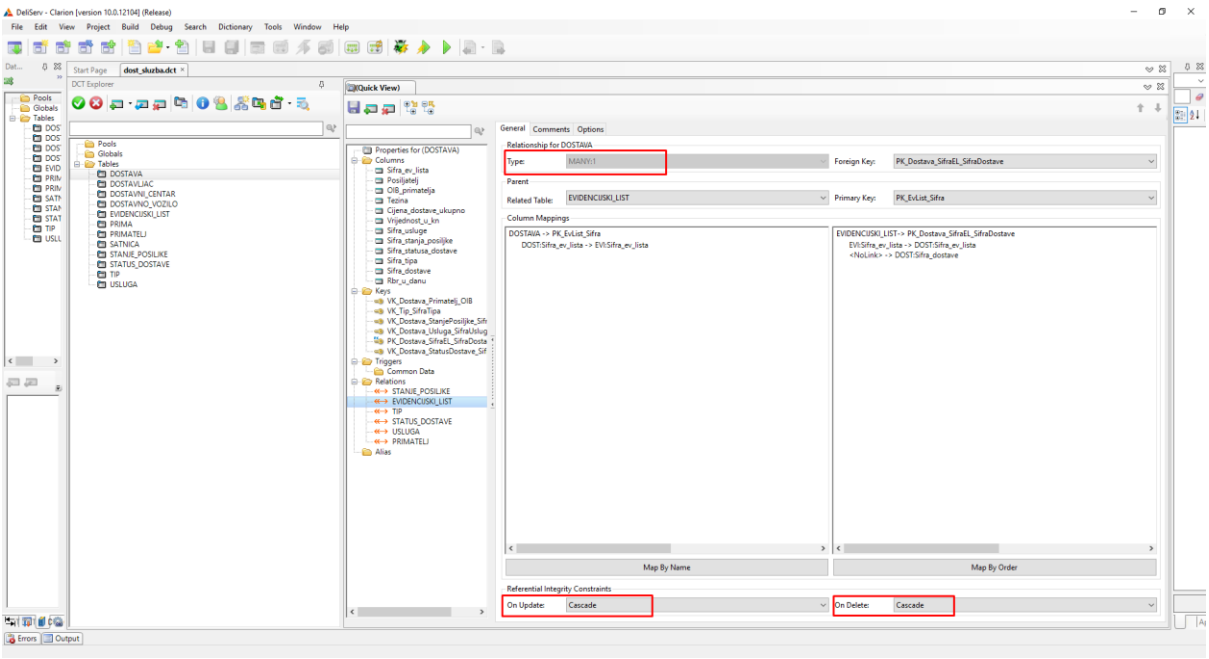
Razlika u kreiranju veze sa tablicom DOSTAVA je u tome što je EVIDENCIJSKI_LIST jaki tip, a DOSTAVA slabi pa je gornja granica brojnosti sa strane ove tablice M. To znači da se *Type* postavlja na 1:MANY. Također, *Referential Integrity Constraints* se postavlja, kako je gore rečeno, na *On Update:Cascade* i *On Delete:Cascade*.



Slika 21. Veza Jaki tip - Slabi tip

2.2.4.2. Veze s tablicom DOSTAVA

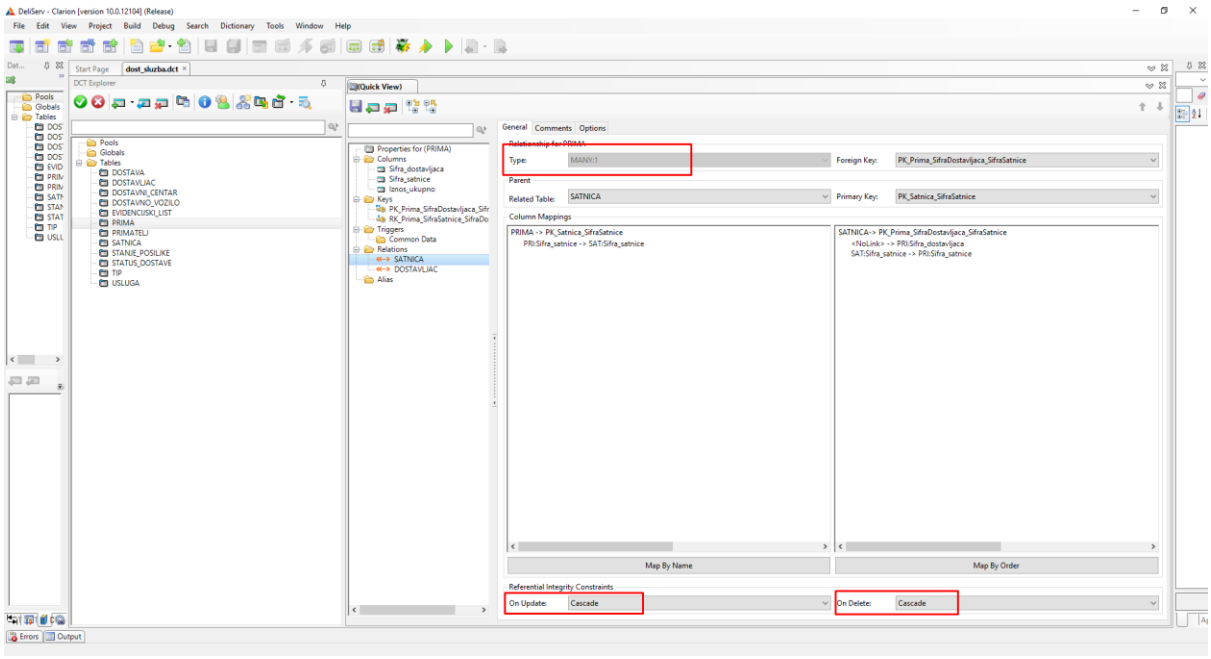
Veze su na isti način formirane među tablicom DOSTAVA i ostalim tablicama koje su povezane s njom (EVIDENCIJSKI_LIST, TIP, USLUGA, STANJE_POSILJKE, STATUS_DOSTAVE, PRIMATELJ). U vezi s tablicom EVIDENCIJSKI_LIST može se vidjeti kako se *Type* automatski postavio na MANY:1 jer je gornja granica brojnosti s ove strane 1. Također, *Referential Integrity Constraints* se automatski postavio na „*Cascade-Cascade“*.



Slika 22. Veza DOSTAVA-EVIDENCIJSKI_LIST sa strane tablice DOSTAVA

2.2.4.3. Veze s tablicom PRIMA

Kako je PRIMA agregacijski tip entiteta (agregacija) te je povezan sa tablicama DOSTAVLJAC i SATNICA postavljamo vezu prema njima od tu. Kao i za vezu između jakog i slabog tipa entiteta moramo paziti da *Referential Integrity Constraints* postavim na *Cascade-Cascade.* Brojnost u agregaciji je uvijek sa strane agregacije (1,1), a sa strana povezanih tablica (1,M). Prema tome *Type* postavljamo na MANY:1, i ostale postavke uobičajeno.



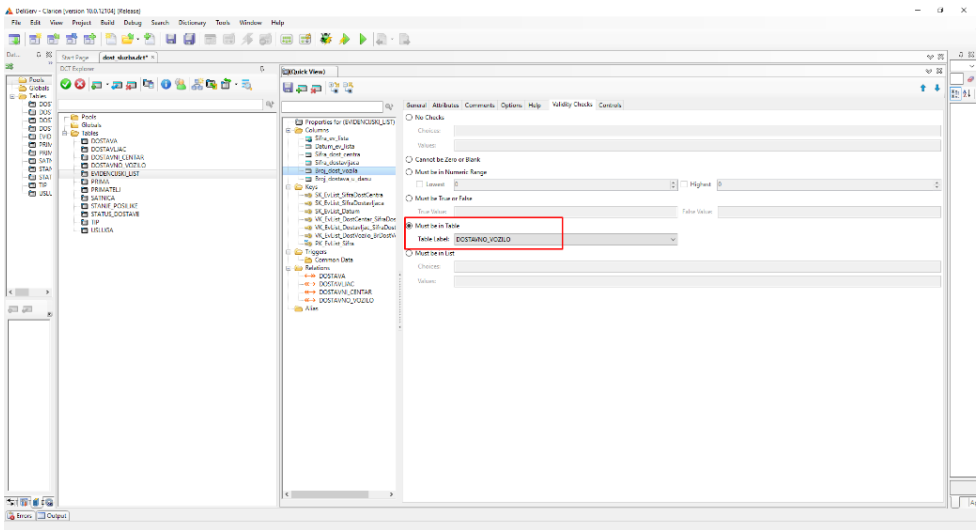
Slika 23. Veza agregacije PRIMA s ostalim tablicama

2.2.5. Must be in Table

Rječnik je sada pri kraju. Ostaje nam još jedan zadatak, a to je u svim tablicama koje sadrže vanjski ključ, odnosno „derivirane“ stupce, u kartici *Validity Checks* za te stupce, odabrati postavku *Must be in Table.* Ova postavka govori aplikaciji da prihvati unos jedino ako takva vrijednost postoji u istoimenom stupcu tablice upisane u ovo polje. Ovaj se korak obavlja tek nakon što su postavljene veze između tablica.

Dakle, u tablici EVIDENCIJSKI_LIST, biramo stupac *Sifra_dost_vozila* te u kartici *Validity Checks* biramo postavku *Must be in Table* i u polju biramo tablicu DOSTAVNO_VOZILO.

Spremamo odabrano i nastavljamo za sve vanjske ključeve. Konačno, dovršena je .dct datoteka i možemo započeti sa izradom same aplikacije.



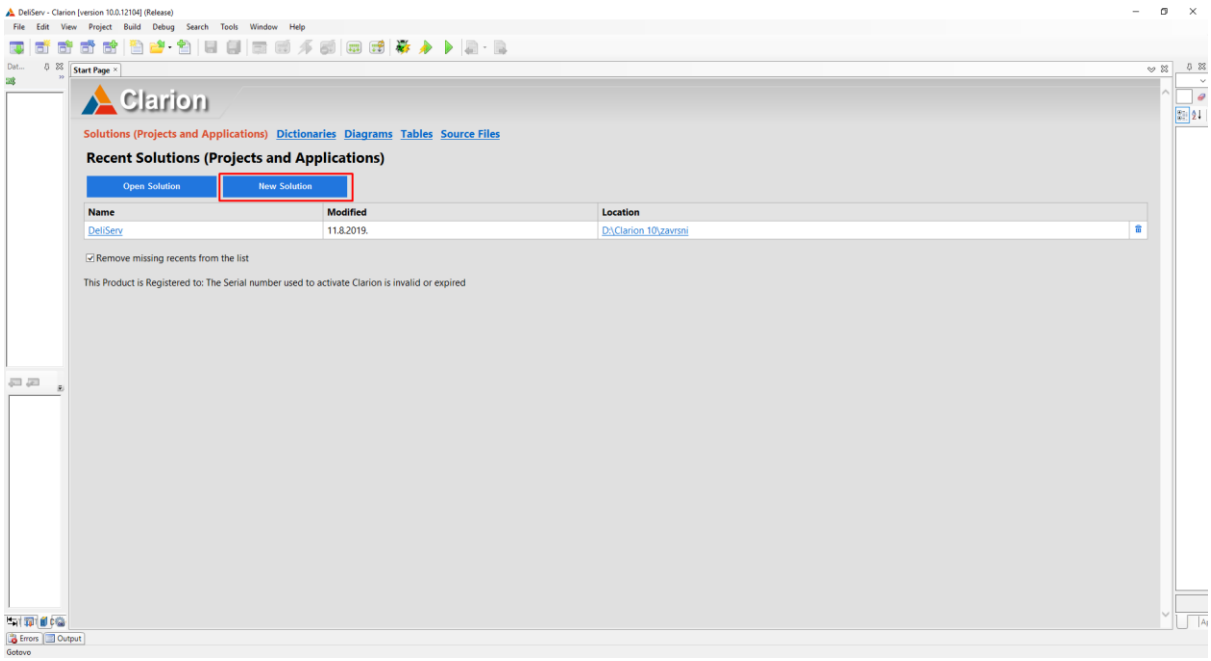
Slika 24. Must be in List

2.3. Izrada aplikacije

Treći i zadnji dio izrade ovog projekta je upravo izrada same aplikacije. Alat Clarion pogodan je za to upravo zato što ima prilagođeno grafičko sučelje koje korisniku omogućuje kreiranje, dodavanje, izmjenu i brisanje elemenata na jednostavan i instinktivan način. Upotreba čistog koda smanjena je na minimum, a pomažu i mnogi predlošci koje softver nudi. Ipak, postoje *bugovi* i greške u programu koje možemo prouzročiti kvar na samom projektu (aplikaciji) pa je dobra praksa često spremati aplikaciju te stvarati sigurnosne kopije (*back-up*) kako bi se osiguralo od gubitka rada.

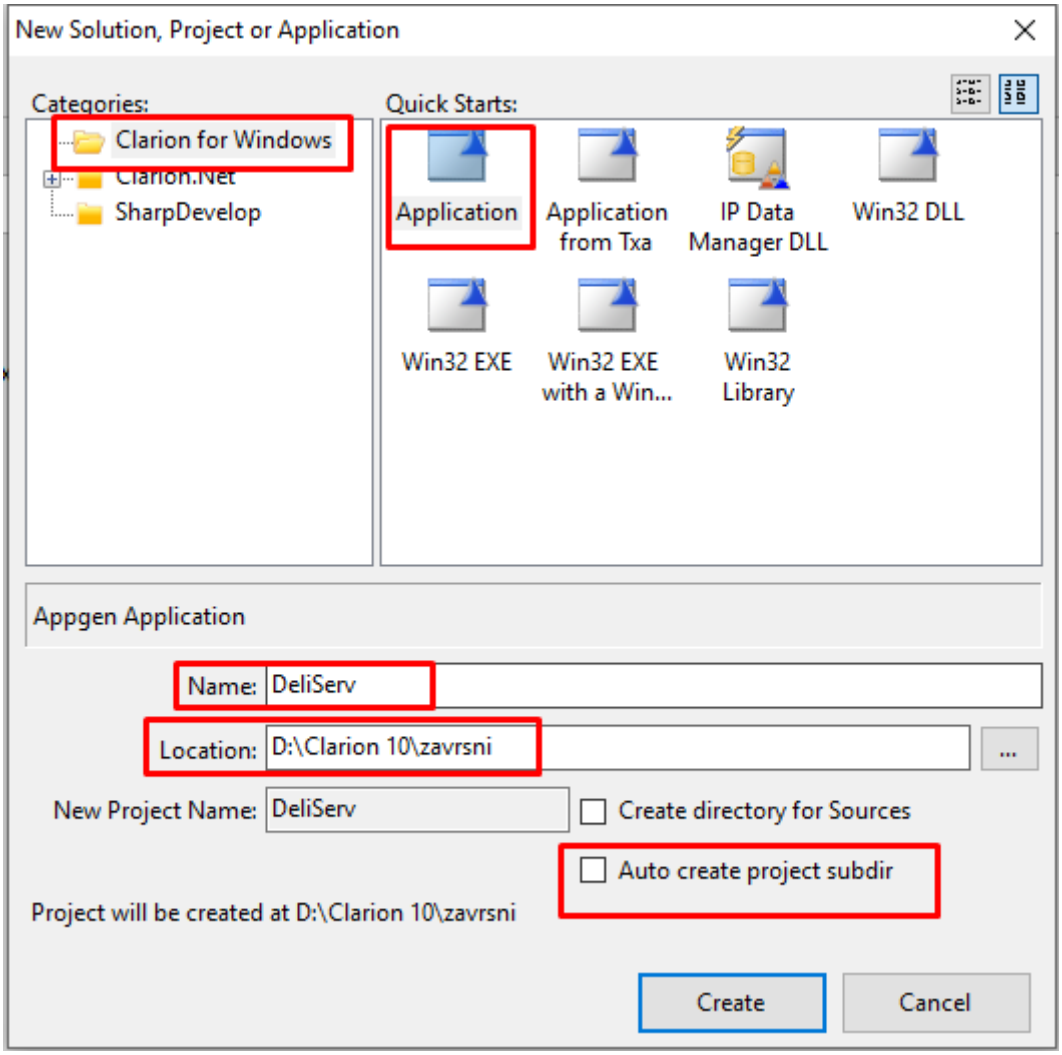
2.3.1. Kreiranje nove .app datoteke

Na početnom zaslonu alata Clarion, pod *Solutions*, biramo *New Solution.*



Slika 25. ***New Solution***

Otvara se novi prozor u kojem, pod mapom *Clarion for Windows*, a u polju *Quick Starts*, biramo *Application*. Potom dajemo ime aplikaciji te biramo mjesto na kojem će biti spremljena. Niže još možemo odznačiti *checkbox Auto create project subdir* kako nam Clarion ne bi nepotrebno stvarao podmapu projekta. Biramo gumb *Create*.



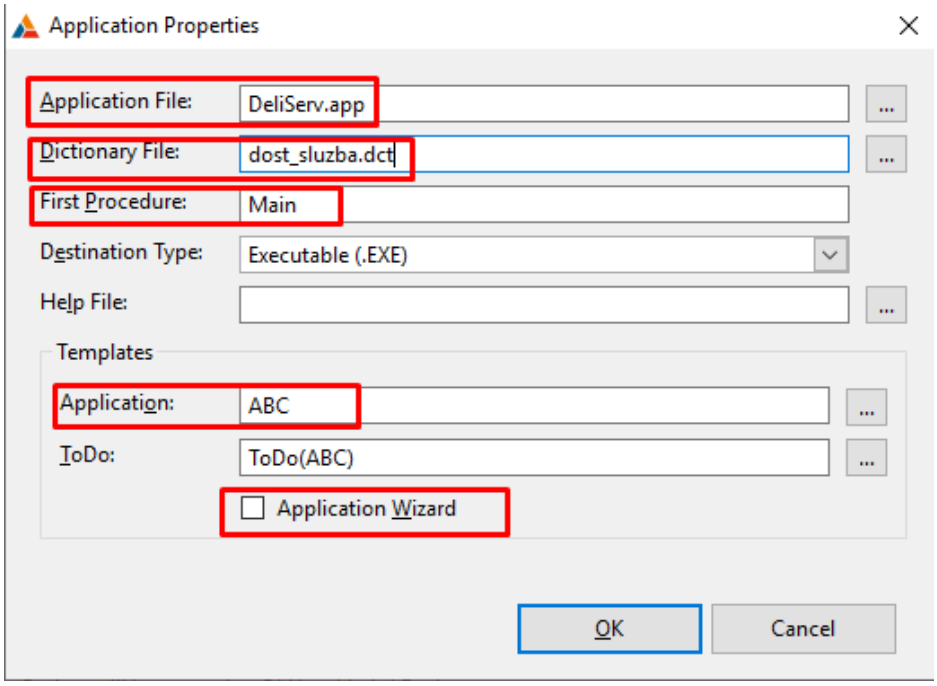
Slika 26. *New Solution* – postavke

Kad smo odabrali temeljne postavke, otvara se sljedeći prozor u kojem se postavljaju parametri naše aplikacije. Polje *Application File* ne moramo dirati, ono je automatski nazvano prema nazivu koji smo dali u prethodnom prozoru. Sljedeće je polje *Dictionary File*. Ovdje biramo .dct datoteku rječnika kojeg smo izradili.

Polje *First Procedure* traži da imenujemo početnu proceduru tj. početni zaslon koji će se prikazati korisniku. Dat ćemo mu ime *GlavniIzbornik*.

U nastavku još postavljamo polja *Application* i *ToDo*, u okviru *Templates*, na *ABC*. To znači da ćemo za izradu prozora (procedura) aplikacije koristiti ABC predloške umjesto Clarionovih.

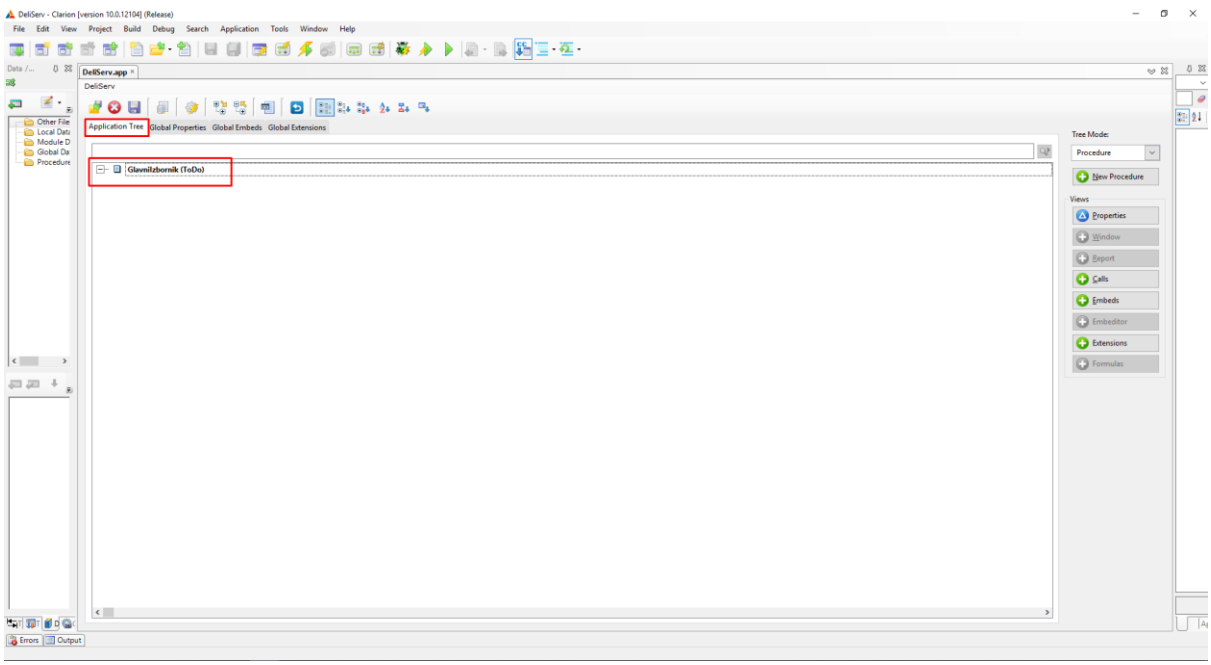
Konačno, mičemo oznaku sa *checkboxa Application Wizard* jer želimo svaku proceduru izraditi i oblikovati samostalno i biramo OK.



Slika 27. Postavke aplikacije

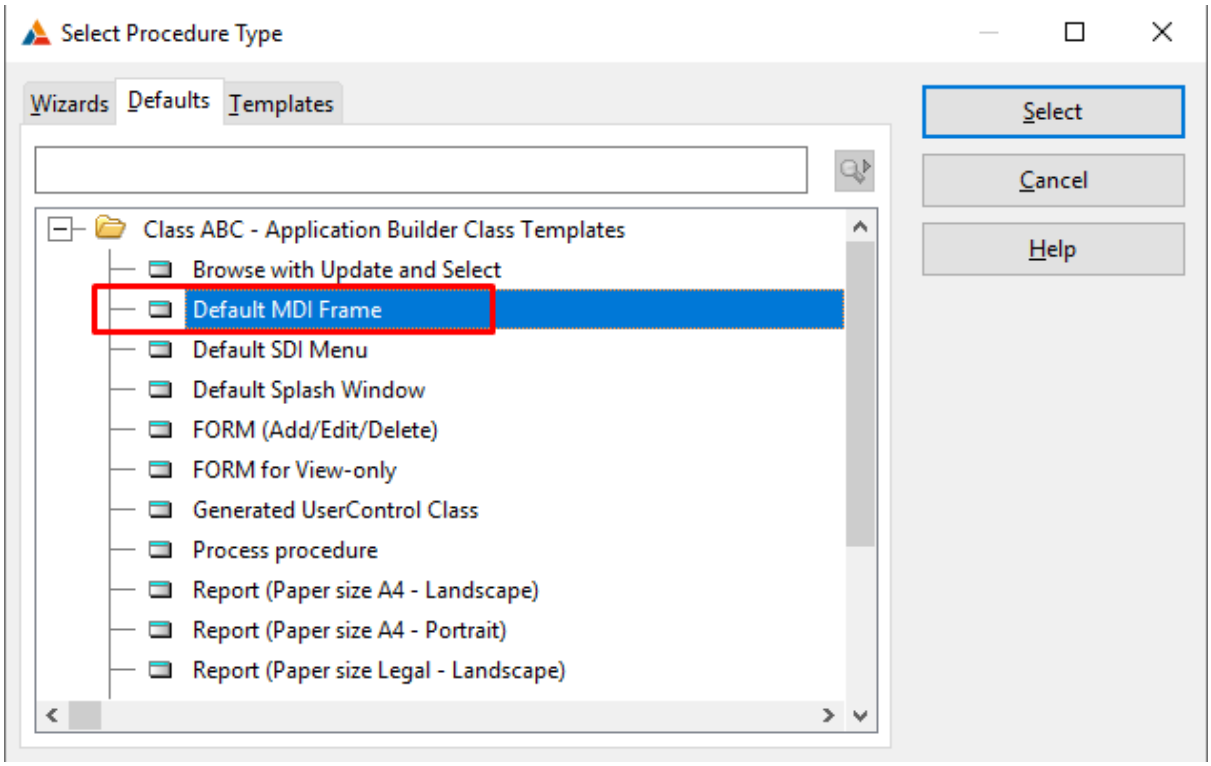
2.3.2. Kreiranje procedure *GlavniIzbornik*

Konačno, otvara se Clarionovo sučelje za izradu aplikacije. Kartica *Application Tree* pregledno prikazuje sve procedure aplikacije i njihove veze. Kako još niti jedna procedura nije stvorena prikazuje se samo *GlavniIzbornik* uz napomenu (*ToDo*) koja nam signalizira da procedura još nije definirana.



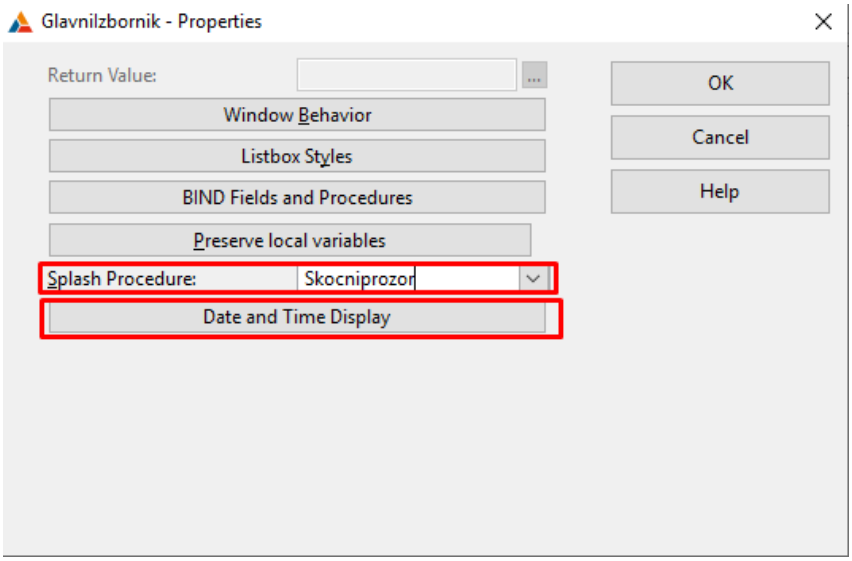
Slika 28. *Application Tree*

Dvoklikom na nju otvara se novi prozor u kojem, pod karticom *Defaults*, biramo tip procedure *Default MDI Frame*. Ovaj tip koristimo samo jednom i to ovdje, za izradu Glavnog izbornika.



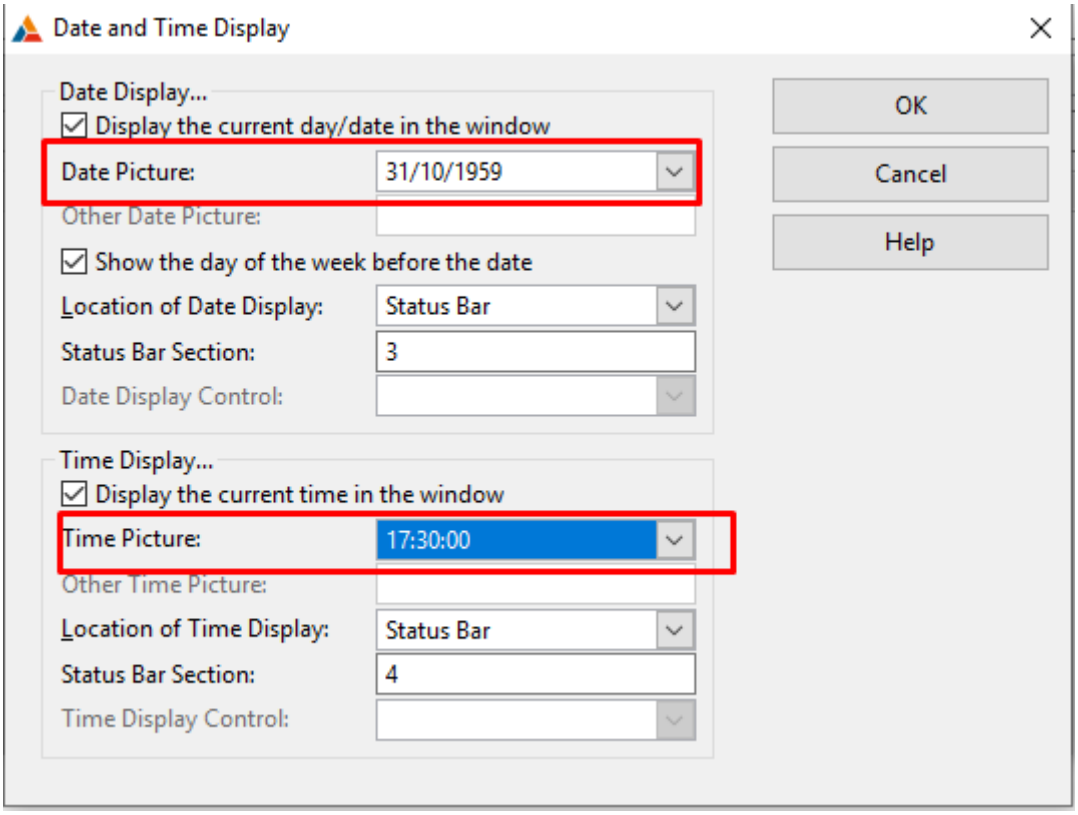
Slika 29. Tip procedure *Default MDI Frame*

Odabirom tipa, prikazuje se *Procedure Properties* prozor u kojem možemo mijenjati *Actions* i sam *Window*. Za početak otvaramo *Actions...* i u novom prozoru u polju *Splash Procedure*, upisujemo *SkocniProzor*. To će biti procedura za skočni prozor tj. prozor koji se kratkotrajno prikazuje pri otvaranju aplikacije.



Slika 30. ***GlavniIzbornik - Actions***

Ispod toga biramo gumb *Date and Time Display*. Ponovno se otvara novi prozor u kojem označavamo *checkbox Display the current day/date in the window* (ispod toga u polju *Date Picture* biramo format datuma koji će se prikazati) te *checkbox Display the current time in the window* (u *Time Picture* biramo format vremena za prikaz). Biramo OK i spremamo postavke za *Actions*. Prelazimo na *Window*.



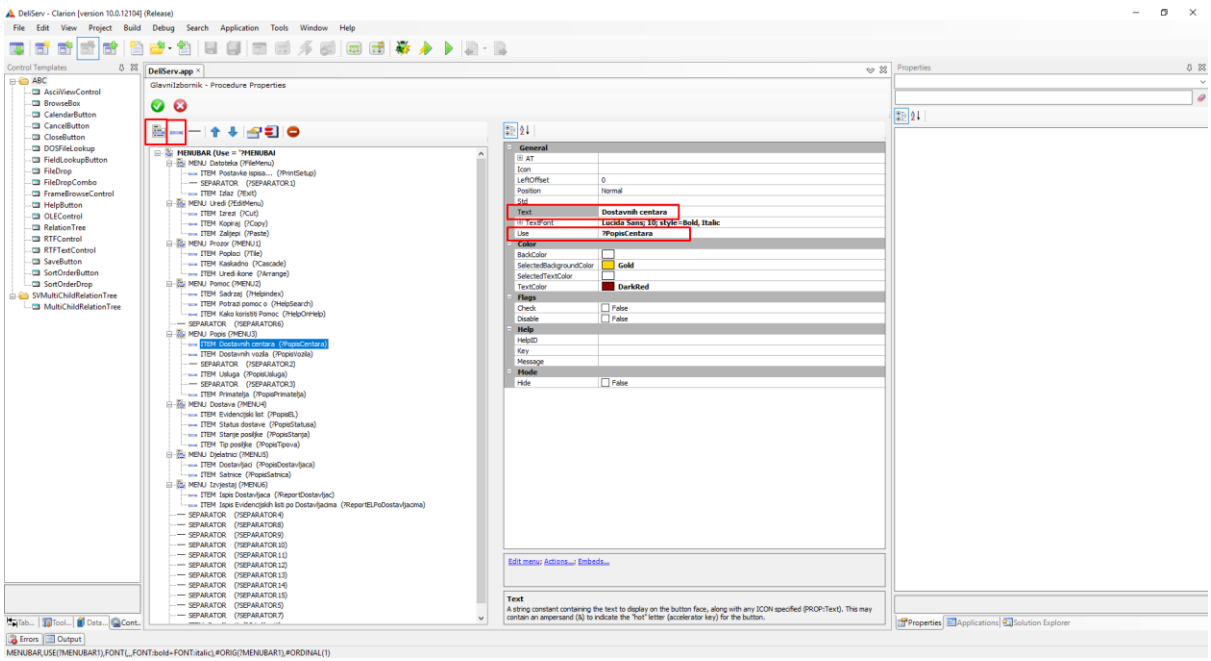
Slika 31. ***Date and Time Display***

Window (prozor) je glavni dio aplikacije. To je ono što će korisnik vidjeti i zato se posebna pažnja posvećuje funkcionalnosti i dizajnu prozora.

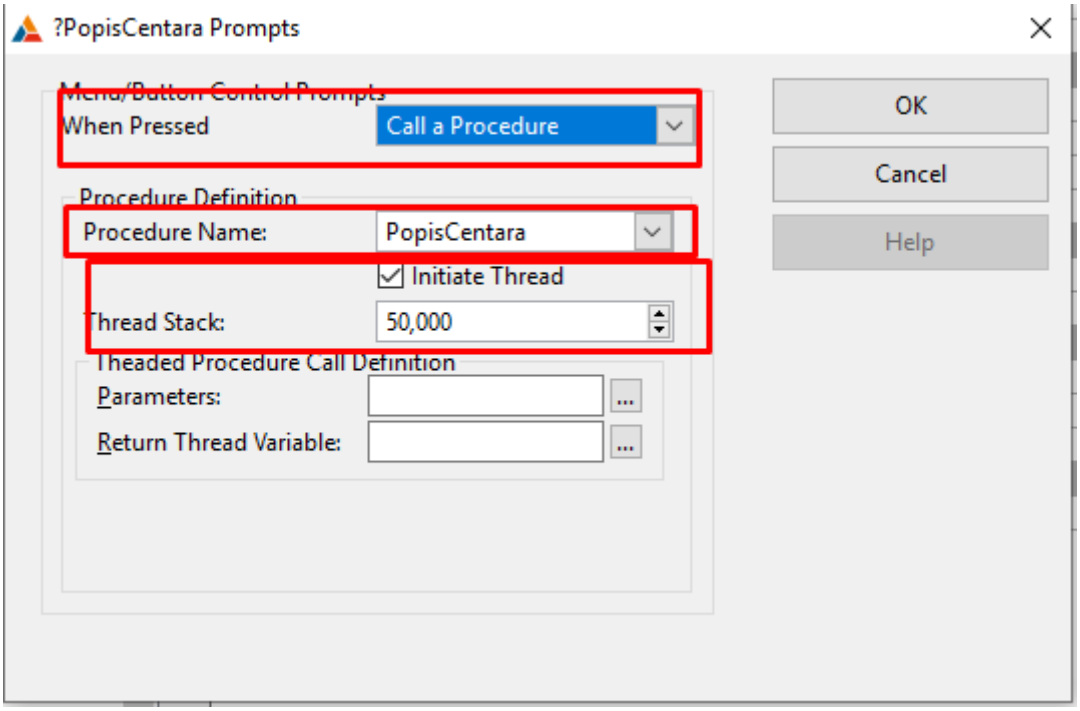
Prozor Glavnog izbornika u početku je siv i prazan. S desne strane prikazana je kartica *Properties*. S nje će se odvijati sve promjene i uređivanja. Pa za početak mijenjamo naslov prozora (*Title*) u DeliServ, dodajemo ikonu aplikacije (*Icon*) te postavljamo *defaultni* font *Lucida Sans* veličine 10. Boja fonta neka bude *DarkRed.* S obzirom da smo ove postavke odabrali na Glavnom izborniku, držat ćemo ih se kontinuirano kroz ostale prozore. Dodajemo i pozadinsku sliku (*BackgroundImage*).

Dalje želimo urediti alatnu traku svoje aplikacije kako bi se na njoj nalazili gumbi za pristup ostalim prozorima tj. procedurama. Alatnu traku biramo desnom tipkom i idemo na opciju *Edit Menu*.

Otvara se novi prikaz, sa svim padajućim izbornicima alatne trake i njihovim elementima. U gornjem lijevom kutu biramo tipku *New Menu* i onda do toga *New Item*. Biramo novi izbornik i dajemo mu ime Popis te uređujemo boje i onda novi element kojeg nazivamo Dostavnih centara. Da bi on bio gumb u polju *Use* s desne strane upisujemo *?PopisCentara*. Međutim, to nije dovoljno da bi se otvarala nova procedura. Potrebno je desnom tipkom odabrati element Dostavnih centara i potom *Actions…* . Otvara se ponovno novi prozor u kojem u polju *When Pressed* biramo *Call a Procedure*, zatim u polju *Procedure Name* upisujemo *PopisCentara*, označavam *checkbox Initiate Thread*, te u polje *Thread Stack* dodajemo broj 50,000. To znači da će se pritiskom na taj element padajućeg izbornika pozvati procedura *PopisCentara* tj. otvorit će se prozor sa zapisima iz tablice DOSTAVNI_CENTAR i za to će se pokrenuti novi *thread* sa 50 000 procesnih niti. Biramo OK i spremamo.



Slika 32. Dodavanje izbornika alatnoj traci i njihovih elemenata



Slika 33. Postavke elementa padajućeg izbornika alatne trake

Na isti način stvaramo dodatne elemente u izbornik Popis. To su Dostavnih vozila (procedura *PopisVozila*), Usluga (*PopisUsluga*), Primatelja (*PopisPrimatelja*).

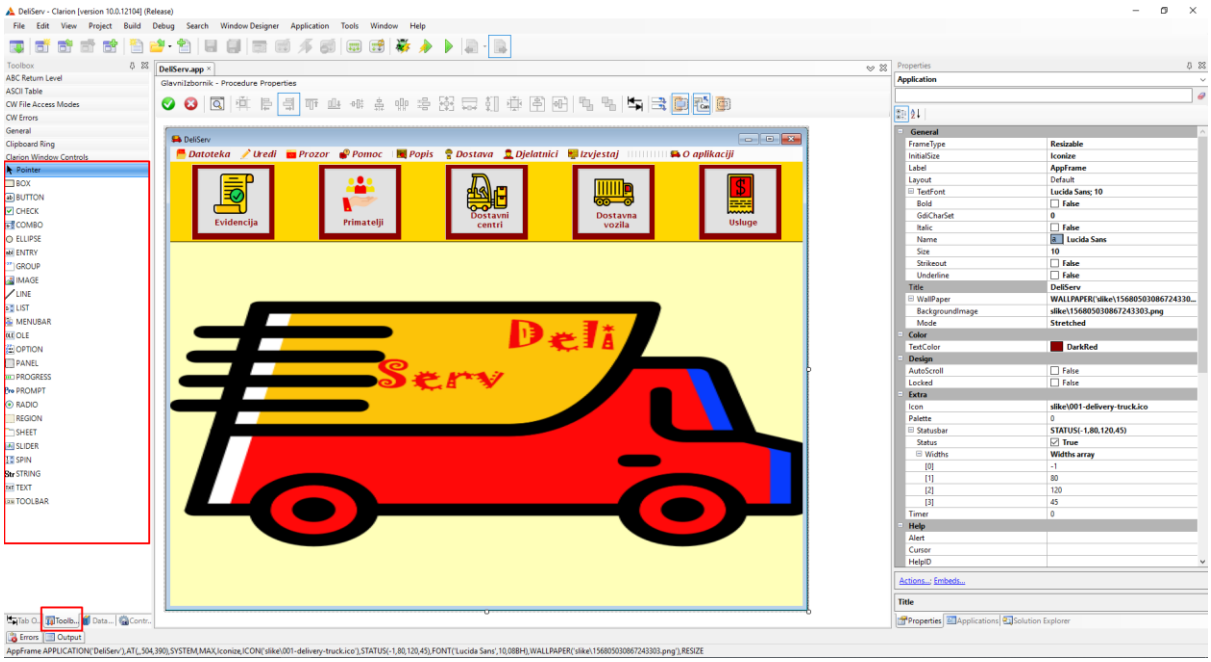
Kako dodajemo nove nazive procedura u polje *Procedure Name*, tako se oni prikazuju u *Application Tree* kartici na početnom sučelju, svi označeni oznakom *Todo.*

Dodajemo još meni Dostava sa elementima Evidencijski list (*PopisEL*), Status dostave (*PopisStatusa*), Stanje posiljke (*PopisStanja*) te Tip posiljke (*PopisTipova*). Još ostaje meni Djelatnici sa elementima Dostavljaci (*PopisDostavljaca*) te Satnice (PopisSatnica).

Tablice koje su agregacije ili slabi tip entiteta nikad nemaju svoju proceduru nego se umeću u povezane.

Spremamo sve i vraćamo se na *Window* Glavnog izbornika. U središnji dio dodajemo, iz kartice *Toolbox*, Clarionov element *Box.* Radi se o kvadratu koji će služiti kao dodatna traka s gumbima. Same gumbe (*Button*) također povlačim iz *Toolboxa* i dodajemo u tu novu traku. Svakog uređujemo na način da mu dodajemo ikonu, mijenjamo veličinu i boju fonta te boju pozadine (sve kroz karticu *Properties*). Da bi oni doista funkcionirali kao gumbi, odabiremo svakog zasebno desnom tipkom i otvaramo prozor *Actions*…. Na isti način kao u elementima menija alatne trake, biramo točnu proceduru koja će se pozvati (sada su već ponuđene u padajućem izborniku), postavljamo *threads*, i spremamo.

Glavni Izbornik je dovršen.



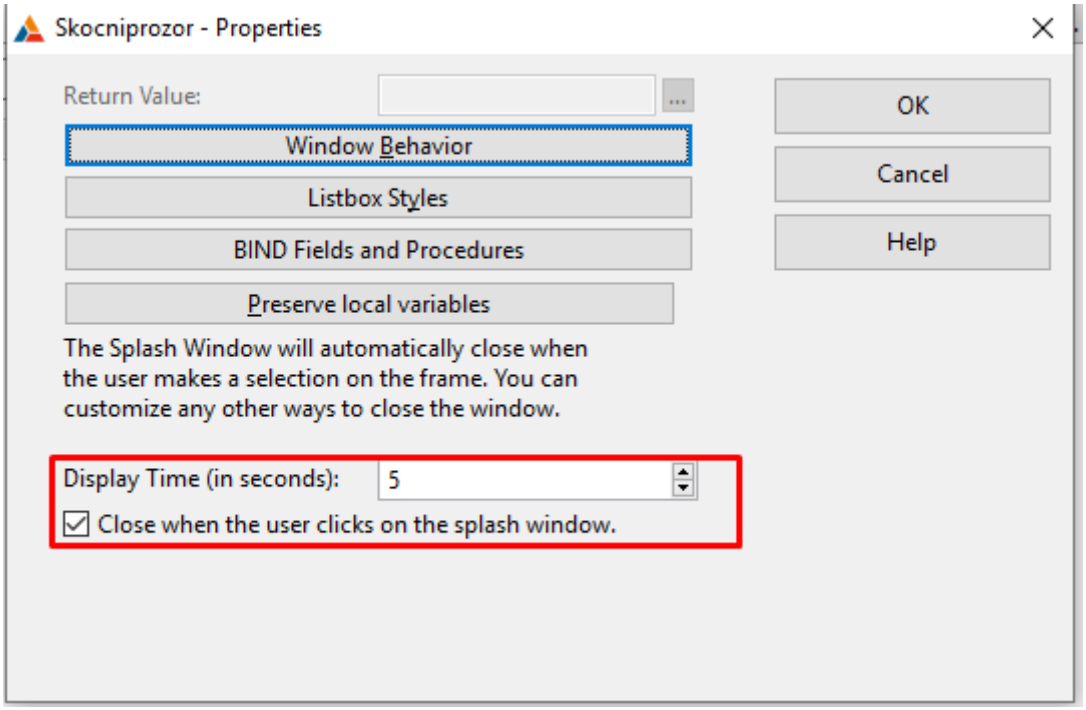
Slika 34. Glavni izbornik

2.3.3. Kreiranje procedure *SkocniProzor*

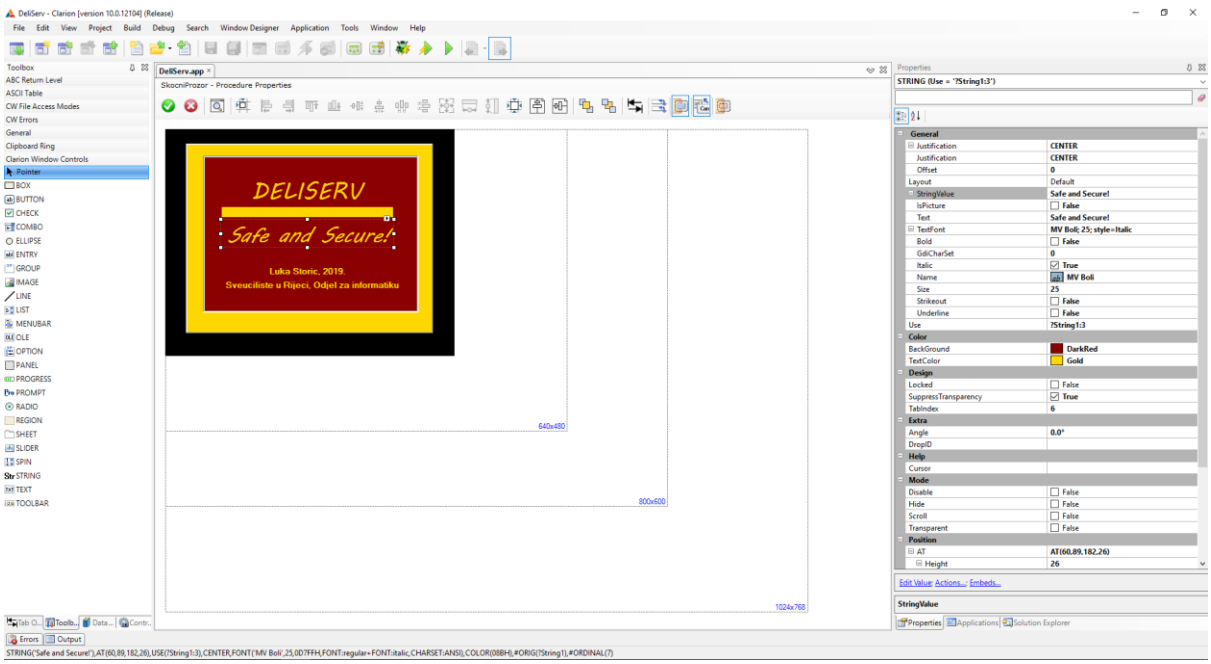
Proceduru za skočni prozor smo najavili već u izradi Glavnog izbornika, tako da on u *Application Tree* postoji pod oznakom *ToDo*. Dvoklikom na njega otvara se prozor za odabir tipa procedure gdje biramo *Default Splash Window*.

Otvaramo njegov *Actions…* i postavljamo *timer* u polju *Display Time (in seconds)* na 5 što znači da će po pokretanju aplikacije skočni prozor biti prikazan 5 sekundi. Kako bi korisnik mogao ukloniti skočni prozor prije isteka tog vremena, označavam *checkbox Close when the user clicks on the splash window*. Spremamo s OK i prelazimo na *Window*.

Ovdje ga uređujemo na način kako smo uređivali i glavni izbornik. Dodajemo oblike iz *Toolboxa*, mijenjamo pozadinske boje, fontove i veličine u *Properties* te na kraju spremamo.



Slika 35. Skočni prozor - *Actions...*



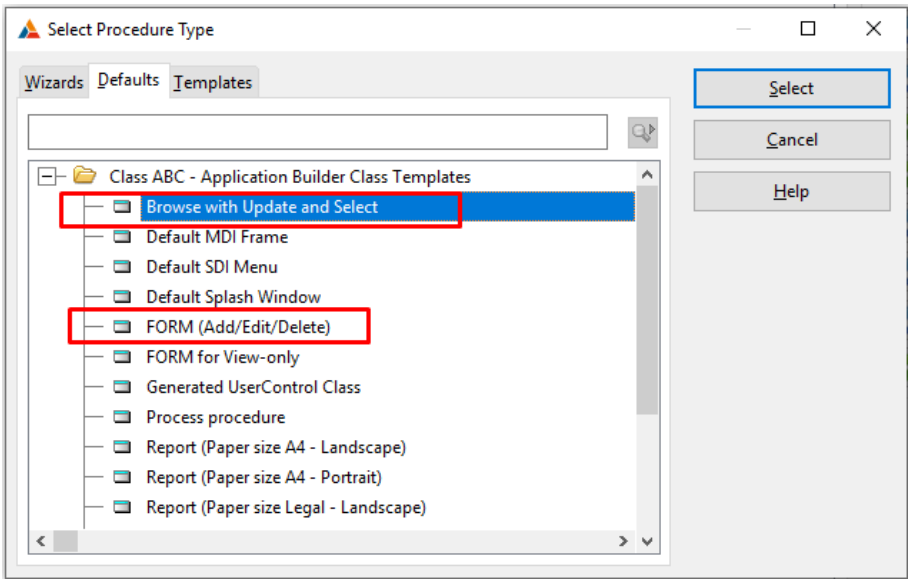
Slika 36. Skočni prozor - *Window*

2.3.4. Kreiranje procedura *PopisCentara* i *AzuriranjeCentara*

Sada kad imamo Glavni izbornik i skočni prozor, možemo početi sa izradom glavnih procedura. Za svaku tablicu (osim PRIMA i DOSTAVA) potrebne su nam po dvije

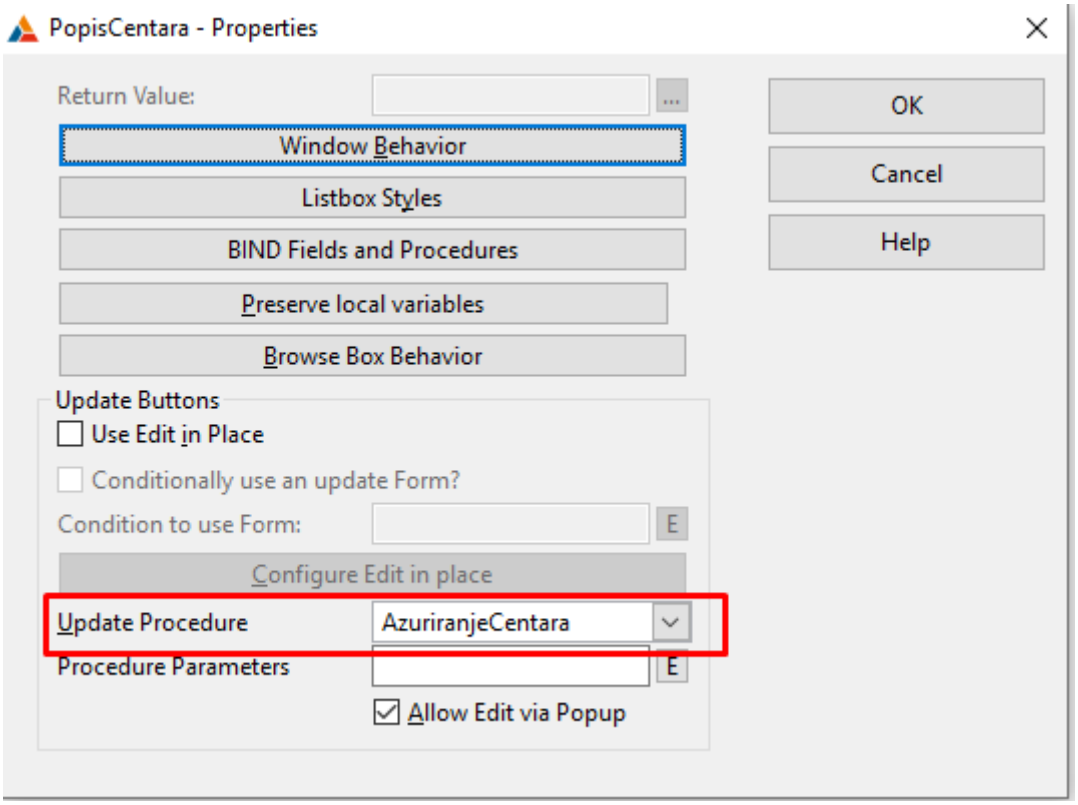
procedure: jedna za pregled podataka i jedna za ažuriranje podataka. Prema tome ćemo ih i nazivati.

Krećemo od tablice DOSTAVNI_CENTAR tj. procedure *PopisCentara* (*ToDo).* Dvoklikom na nju otvara se prozor za odabir tipa procedure i tu biramo *Browse with Update and Select* .



Slika 37. *Browse with Update and Select* i *FORM(Add/Edit/Delete)*

Otvara se, standardno, *Procedure Properties* i tu prvo otvaramo *Actions…* gdje samo moramo u polje *Update Procedure* upisati *AzuriranjeCentara*. To znači da će se za ažuriranje podataka ove tablice otvarati nova procedura pod tim imenom. Vrijedno je primijetiti da se izrada svake procedure, neovisno o tipu, vrši istim ili gotovo istim postupkom. Nakon *Actions…* otvaramo *Window* (!).

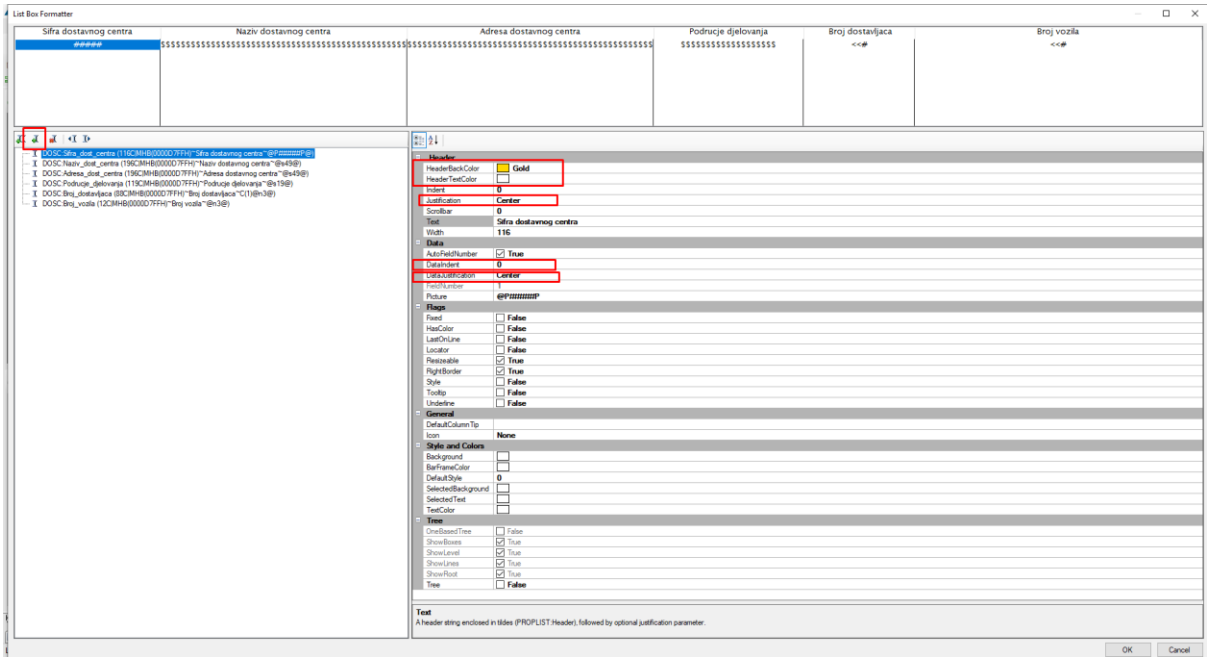


Slika 38. *PopisCentara - Actions...*

Kod kreiranja *Browse* procedure, u *Windowu*, krećemo od toga da u kartici *Data/Tables*, pod mapom *File-Browsing List Box,* dvokliknem *ToDo* i odaberemo tablicu DOSTAVNI_CENTAR. Zatim ponovno dvokliknemo na ime tablice te odaberemo ponuđeni primarni ključ. Ovaj postupak je, dakle, kritičan i bez njega prikaz podataka iz tablice neće raditi. Nakon toga, na standardan način, uredimo prozor. U kartici *Properties* podesimo font, veličinu, boju fonta, boju pozadine itd. Poseban detalj je promjena naslova prozora te naslova kontrolnih gumba u hrvatski jezik te dodavanje ikone. Kad smo gotovi sa dizajnom, desnom tipkom odabiremo bijeli prostor za prikaz podataka tzv. *List Box*. Odabiremo *List Box Format*…. Ovdje biramo stupce iz tablice koji će se prikazivati u *List Boxu* i to na način da s lijeve strane biramo gumb *Add Field* i redom dodajemo stupce. Nakon odabira prvog stupca, koji je obično i primarni ključ, možemo postaviti, s desne strane, oblikovanje stupca. Centriramo naslov stupca te sadržaj. Također, možemo promijeniti boju pozadine zaglavlja stupca.

Tu dolazimo do neobičnog problema kojeg smo spomenuli u uvodu. Iz nekog razloga, kada smo probali promijeniti i boju fonta u stupcu, podaci se više nisu zapisivali u točne stupce niti na ispravan način. S obzirom da je teško shvatiti poveznicu između boje fonta i rječnika, izostavili smo ovaj dio dizajna.

Nakon što smo postavili oblikovanje prvog stupca, ostali dodani će po *defaultu* biti oblikovani na isti način. Biramo OK i spremamo sve.

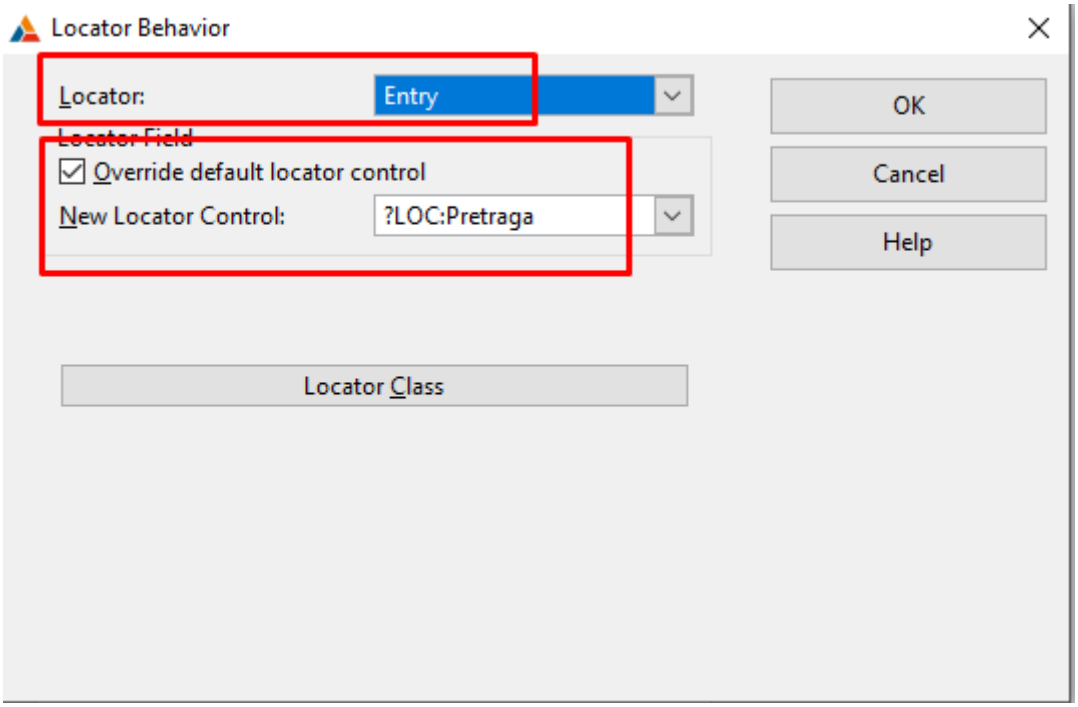


Slika 39. List Box Format...

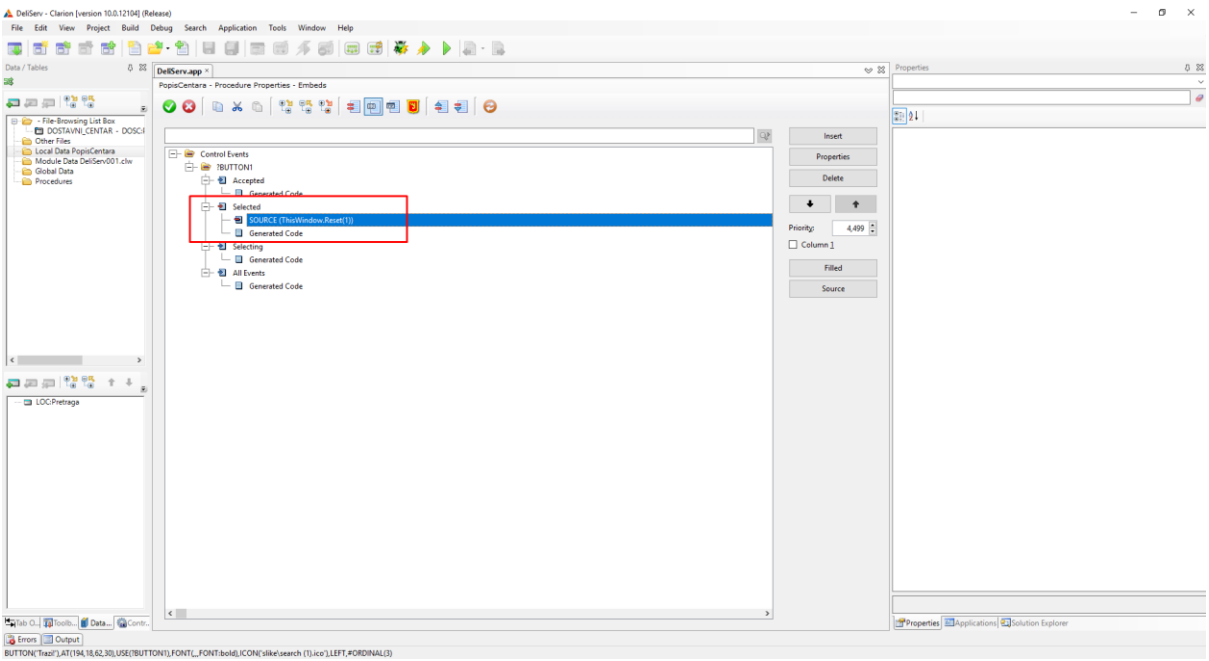
Kod velikog broja zapisa u tablici uvijek je dobro imati omogućeno pretraživanje. To radimo na način da u kartici *Data/Tables*, u mapi *Local Data PopisCentara*, desnim klikom dodamo stupac (*Add Column*). Dodavanje vršimo na isti način kao i u rječniku, a stupac nazivamo LOC:Pretraga. Skraćenica LOC signalizira da se radi o lokalnom stupcu tj. stupcu koji nije dio originalne tablice. On će služiti samo za pretragu. Tip podataka i format ne diramo. Kada je stupac kreiran, iz okvira s lijeve strane povlačimo stupac u prozor, iznad *List Boxa.* Tekst prije polja mijenjamo u *Pretraga:* te boju samog polja mijenjamo u bijelo radi lakše vidljivosti.

Sljedeći korak je odabrati *List Box* desnim klikom, te otvoriti prozor *Actions….* U kartici *Default Behavior* nalazi se gumb *Locator Behavior* kojeg biramo. Otvara se novi prozor u kojem, u prvom polju, mijenjamo vrijednost na *Entry*, ispod toga označavamo *checkbox Override default locator behavior* te u polju *New Locator Control* biramo ?*LOC:Pretraga*. Bitno je odabrati polje, a ne *Prompt* tj. tekst ispred polja. Spremamo sve i vraćamo se u *Window.*

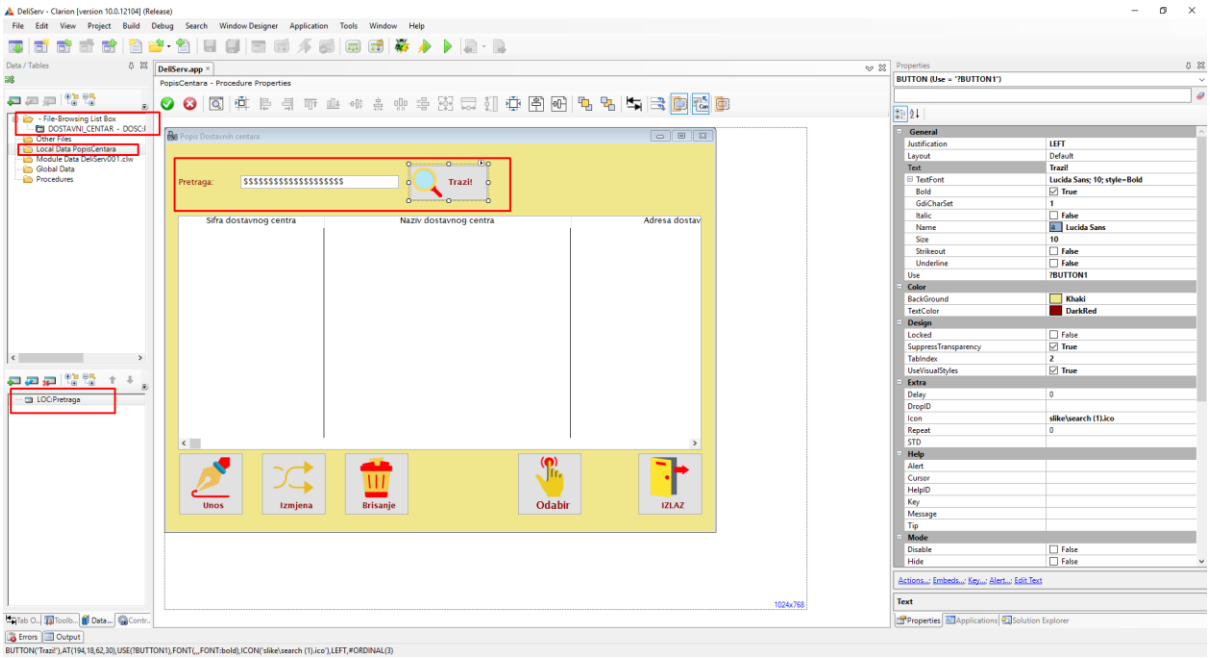
Ovdje ćemo još dodati gumb za pretragu. Iz *Toolboxa* povlačimo *button* pored polja za pretragu, mijenjamo mu ime, font i boju. Zatim, biramo gumb desnim klikom i otvaramo *Embeds….* Otvara se prikaz sličan *Application Treeu* , koji nam nudi zapise koda o ponašanju gumba kad je *Accepted, Selected, Selecting* ili *All Events.* Biramo dvoklikom *Selected* i tu se prvi put susrećemo s upotrebom samog koda u Clarionu. Upisujemo ime klase *ThisWindow.Reset(1)*. Time osiguravamo da ,sa svakim novim unosom, gumb resetira prozor tj. radi ponovnu pretragu. Spremamo sve i izlazimo u *Application Tree.* Procedura je izrađena.



Slika 40. Postavljanje pretraživanja - *Locator Behavior*

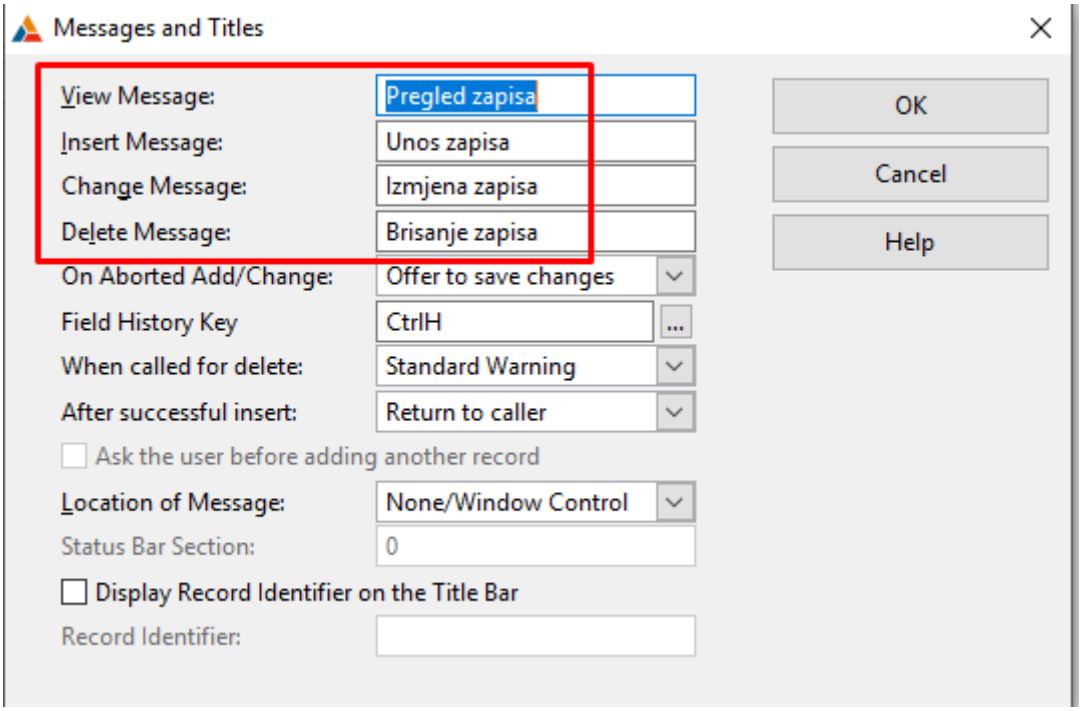


Slika 41. Pretraga - *Embeds...*



Slika 42. ***PopisCentara - Window***

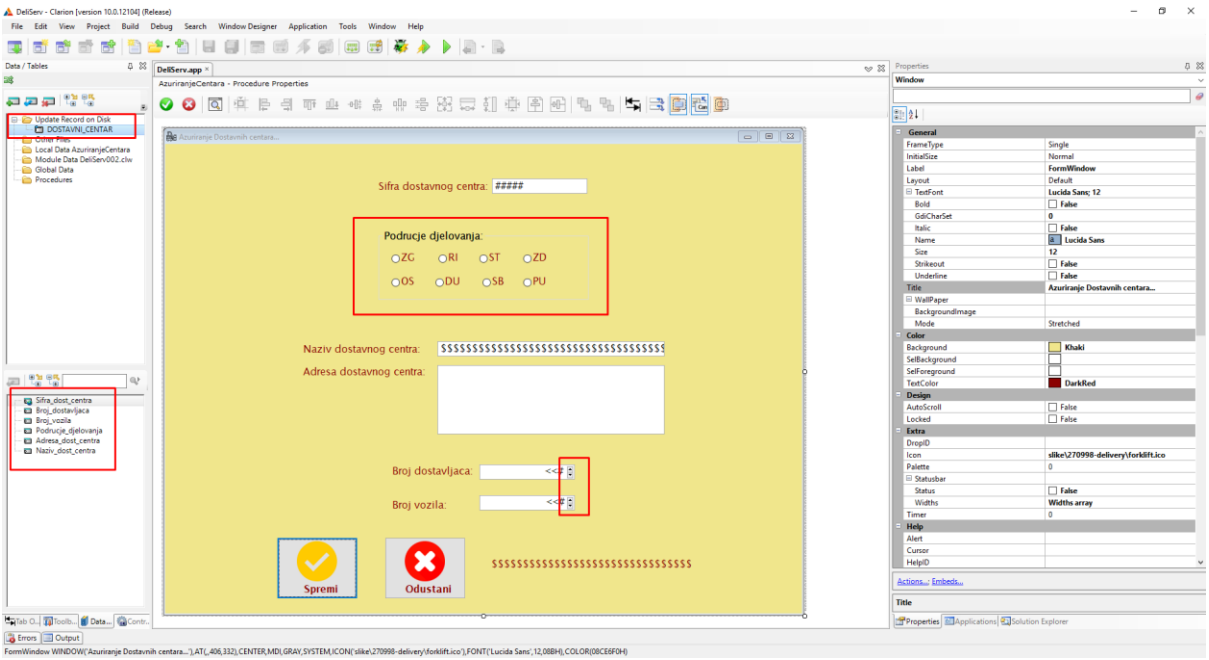
Kada je završena *Browse* procedura, moramo kreirati i *Update* proceduru tj. *AzuriranjeCentara*. Na početnom zaslonu dvokliknemo proceduru te u novom prozoru biramo *FORM (Add/Edit/Delete)* . Ponovno prvo idem u *Actions…,* biramo gumb *Messages and Titles* i u tom prozoru prevodimo na hrvatski poruke kod pregleda, unosa, izmjene te brisanja podataka. Spremamo i idemo u *Window*.



Slika 43. ***Messages and Titles***

U *Windowu* prvo, kao i kod *Browse* procedure, u *Data/Tables* dodajemo odgovarajuću tablicu, ali ne dodajemo i primarni ključ. Nakon toga, sve stupce iz tablice vučemo u prozor i slažemo ih proizvoljno. Stupac *Podrucje_djelovanja*, zbog postavki u rječniku, ne prikazuje se kao polje za upis već kao okvir sa prikazanim *radio* gumbima. Polja za unos stupaca *Broj_dostavljaca* i *Broj_vozila* imaju prikazane strelice, također, zbog postavki u kartici *Controls* u rječniku.

Uređujemo prozor standardno, preko kartice *Properties*, i spremamo proceduru.



Slika 44. *AzuriranjeCentara - Window*

U trenutku kad tablica ima obje procedure (*Browse* i *Update)* definirane, ona je spremna za aplikaciju i prelazi se na sljedeću tablicu.

Na identičan način kreirali smo procedure: *PopisPrimatelja* i *AzuriranjePrimatelja*, *PopisSatnica* i *AzuriranjeSatnica*, *PopisStanja* i *AzuriranjeStanja*, *PopisStatusa* i *AzuriranjeStatusa*, *PopisTipova* i *AzuriranjeTipova*, *PopisUsluga* i *AzuriranjeUsluga*, *PopisVozila* i *AzuriranjeVozila* te *AzuriranjeDostavljaca*.

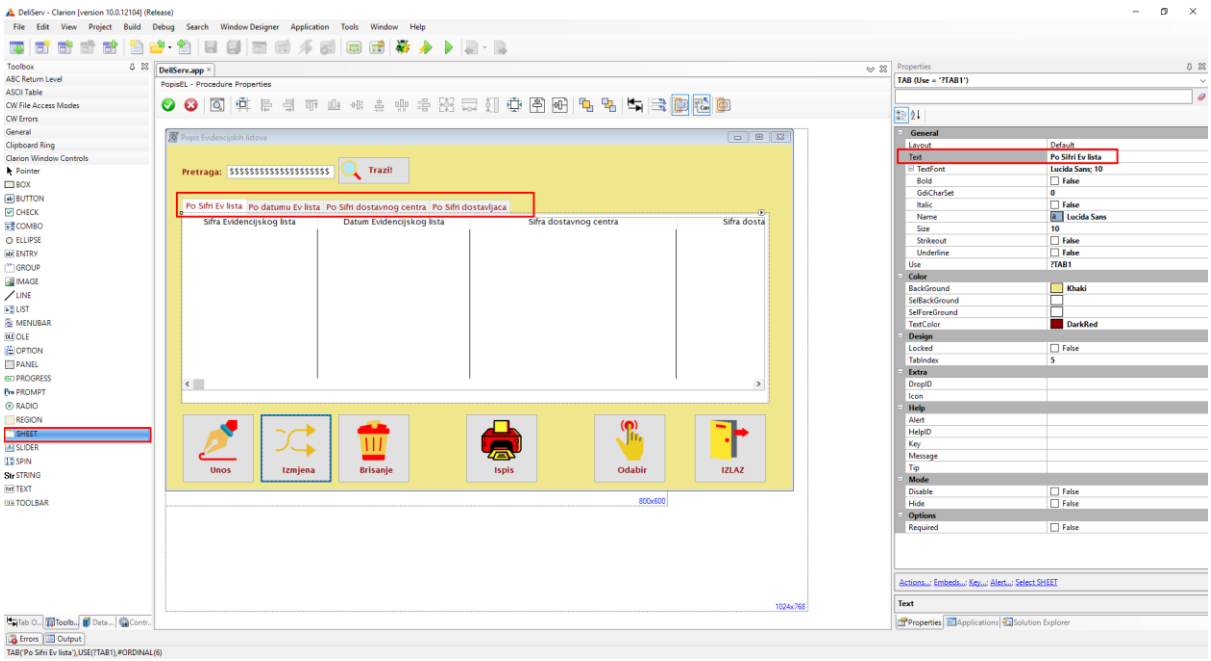
Procedure *PopisEL*, *AzuriranjeEL*, *AzuriranjeDostava*, *PopisDostavljaca* te *AzuriranjePrima* imaju svoje posebnosti te ćemo njih objasniti u nastavku.

2.3.5. Kreiranje Procedura *PopisEL* i *AzuriranjeEL*

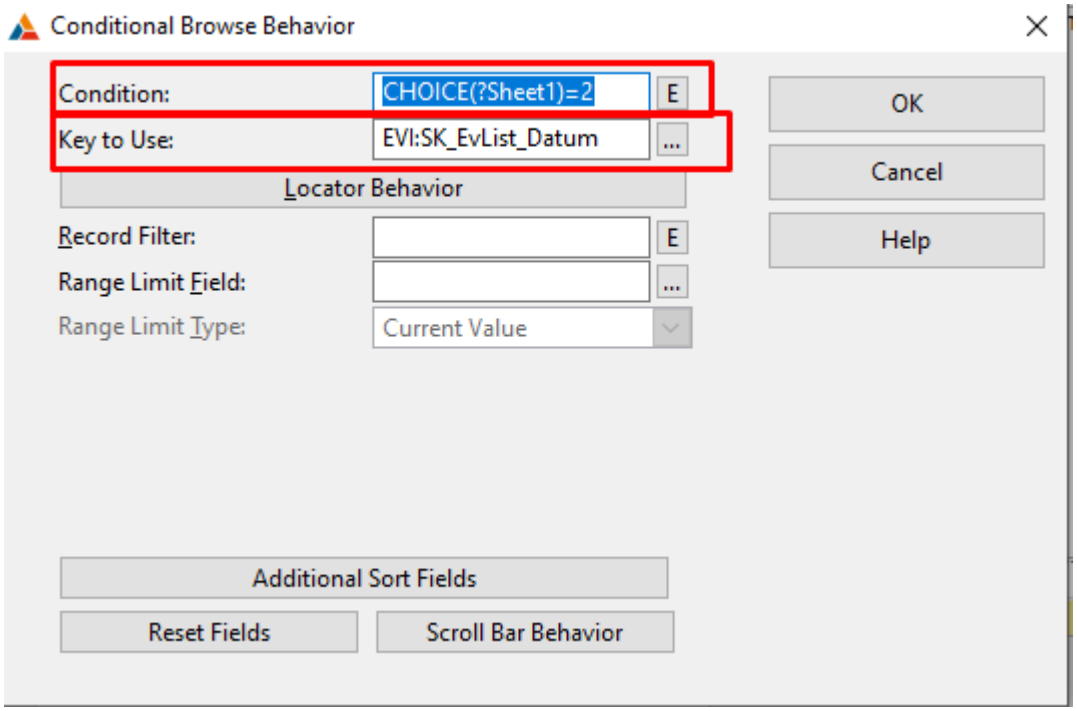
Kreiranje procedure *PopisEL* započinje kao i sve ostale. Bira se tip procedure, u *Actions…* definiramo *Update Procedure* i ulazimo u *Window.*

Ovdje po pravilu biramo tablicu i njezin primarni ključ, uređujemo prozor standardno preko *Properties,* te otvaramo *List Box Format…* . Dodajemo stupce iz tablice, oblikujemo ih i spremamo odabrano. Također, i ovdje dodajemo pretraživanje s pripadajućim gumbom.

U ovoj tablici definirali smo tri ključa za sortiranje, pa ćemo ih uključiti na ovom *List boxu*. To radimo na način da iz *Toolboxa* uzmemo element *Sheet* i pozicioniramo ga preko cijelog *List boxa*. Odabirom male strelice u desnom gornjem kutu, *Sheetu* dodajemo 4 kartice (*Add Tab*). U *Propertiesu* biramo jednu po jednu karticu i imenujemo (*Text*) ih prema ključevima tj. prvu karticu prema primarnom ključu (Po Sifri Ev lista), a ostale prema ključevima za sortiranje (Po datumu Ev lista, Po Sifri dostavnog centra i Po Sifri dostavljaca). Zatim, *sheet* prebacimo u pozadinu (*Send to Back*), a sam *List Box* odaberemo desnim klikom i otvorimo *Actions…* . U kartici *Conditional Behavior* biramo gumb *Insert* i otvaramo novi prozor. Ovdje, u polje *Condition*, upisujemo izraz „*CHOICE(?sheet1)=2*“ .To znači da se uvjet za sortiranje postavlja na drugoj kartici (jer je prva rezervirana po *defaultu* za primarni ključ). U polju *Key to Use* biramo ključ za sortiranje *SK_EVList_Datum* kako bi se uistinu sortiralo prema datumu Evidencijskog lista. Biramo OK i spremamo. Dodajemo tako i za preostala dva ključa, jedino u izrazu mijenjamo broj iza jednakosti na 3 pa na 4. Sa OK spremamo sve i ova procedura je dovršena.



Slika 45. *PopisEL* - Sortiranje po karticama



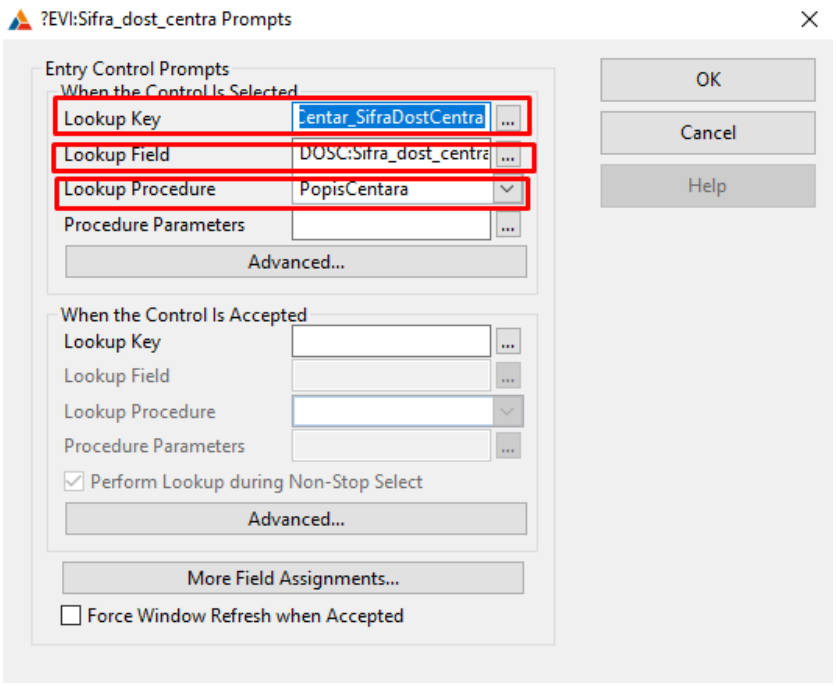
Slika 46. *Conditional Behavior* - Postavljanje sortiranja prema ključu

AzuriranjeEL razlikuje se od prethodnih *Update* procedura po tome što sadrži *List Box.* Međutim, bez obzira na to, početak je isti. Tip procedure je *FORM (Add/Edit/Delete)*, u *Actions…* prevodimo sistemske poruke, otvaramo *Window*, kroz *Properties* uređujemo prozor.

U kartici *Data/Tables* dodajemo tablicu EVIDENCIJSKI_LIST te iz nje vučemo stupce u prozor i ovdje ih rasporedimo po želji.

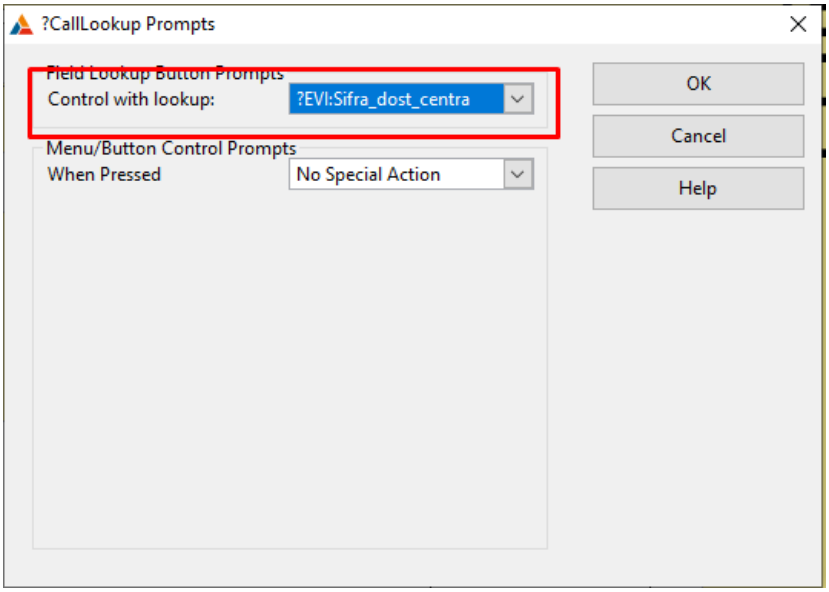
Kako je većina stupaca ove tablice nečiji vanjski ključ, potrebno je postaviti tzv. *Lookup* proceduru. To znači da kad korisnik odabere određeno polje, umjesto upisa podataka, otvoriti će mu se pripadajući prozor tj. *Browse* procedura iz koje će odabrati podatke za to polje.

Za početak biramo polje *Sifra_dost_centra* desnim klikom te otvaramo *Actions….* U novom prozoru, prvo ponuđeno polje zove se *Lookup Key*. Tu biramo ključ po kojem ćemo doći do tražene tablice tj. izvršiti pretragu u njoj. Zatim u polju *Lookup Field* biramo polje iz kojeg ćemo uzeti vrijednost (polje mora biti dio ključa). Konačno, u polju *Lookup Procedure* odabiremo proceduru *PopisCentara* koja će se otvoriti po odabiru polja. Biramo OK i spremamo.



Slika 47. ***AzuriranjeEL – Lookup***

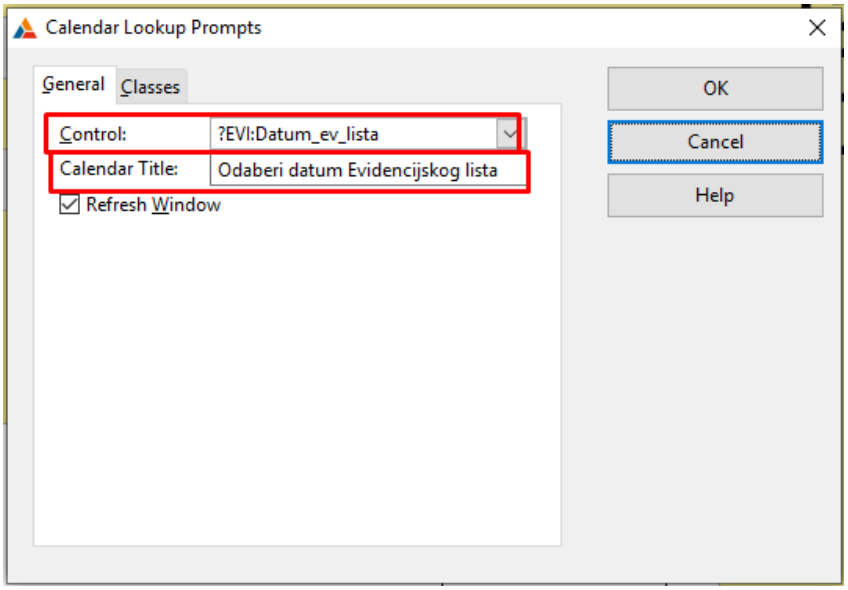
Da olakšamo pretraživanje, uz polje dodat ćemo *Field Lookup Button* iz kartice *Control Templates*. Clarion se, nažalost, zna srušiti pri korištenju ovih predložaka pa prije svega spremamo dosadašnji rad. Nakon što smo dodali gumb, mijenjamo mu dizajn i biramo ga desnom tipkom te otvaramo *Actions…* . Ovdje, u prvom polju *Control with Lookup*, postavljamo vrijednost *?EVI:Sifra_dost_centra*. Dakle, pritiskom na gumb, izvršava se *lookup* kao i pritiskom na polje.



Slika 48. ***Field Lookup Button - Actions...***

Ove dvije radnje ponavljamo i za stupce Sifra_dostavljaca te Broj_dost_vozila. Kod polja za unos stupca *Datum_ev_lista*, iz kartice *Control Templates,* uz ponovno spremanje rada, dodajemo *Calendar Button*. Uređujemo mu dizajn te ga biramo desnom tipkom i otvaramo *Actions…* . Biramo gumb *Calendar Lookup Prompts* i u novom prozoru, u polju *Control*,

upisujemo izraz *?EVI:Datum_ev_lista*, te niže, u polje *Calendar Title*, dodajemo naslov kalendaru tj. uputu korisniku koji datum treba odabrati. Spremamo i vraćamo se u *Window*.



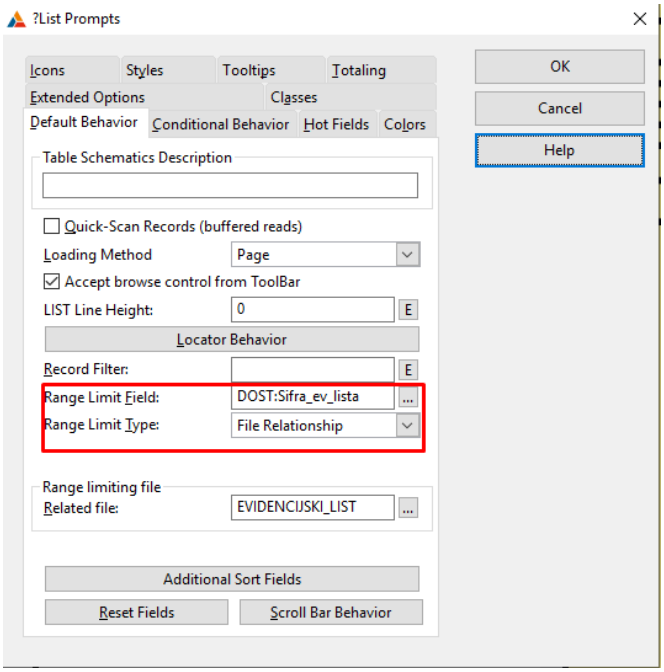
Slika 49. *Calendar Lookup Prompt*

Sada dodajemo novi *List Box,* onaj u kojem će se prikazivati sve dostave tog Evidencijskog lista. Nakon što smo sve spremili, u prozor, iz kartice *Control Templates*, povlačimo *BrowseBox* i smještamo ga ispod prije stvorenih polja. U kartici *Data/Tables* stvara se još jedna mapa *File-Browsing List Box* sa oznakom *ToDo.* Dakle, dodajemo kao i do sad, tablicu DOSTAVA te njezin primarni ključ.

Sam *List Box* biramo desnom tipkom, otvaramo *List Box Format…* i dodajemo te uređujemo stupce koje želimo da se prikazuju. S obzirom da želimo da se prikazuju i stupci iz drugih, povezanih, tablica, prilikom dodavanja stupca, preko gumba *Add,* možemo dodati tablice te potom birati njihove stupce. Također, radi veće preglednosti, grupiram određene stupce (*Add Group)* pod isto zaglavlje. Spremamo sve i vraćamo se u *Window*.

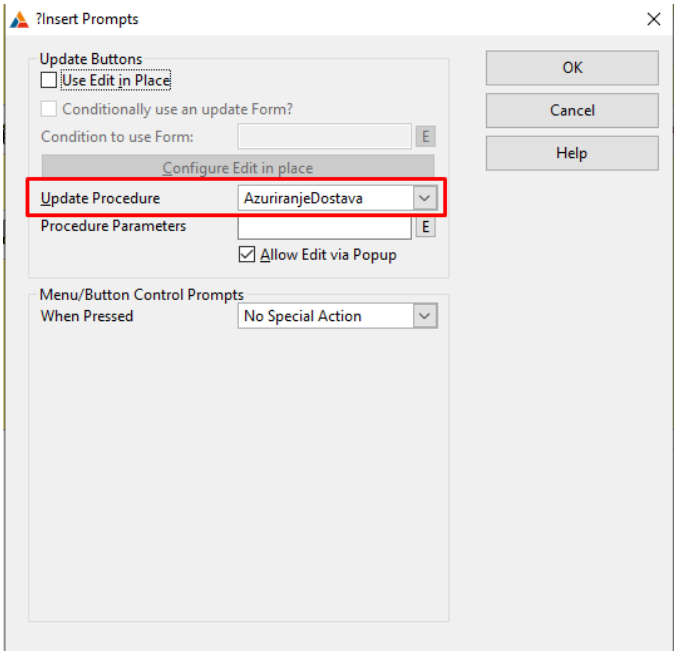
Sljedeći korak koji želimo postaviti je taj da se u ovom *List Boxu* ne prikazuju sve dostave, već samo one koje pripadaju ovoj Evidencijskoj listi. To radimo preko polja *Range Limit Field* koje se nalazi unutar *Default Behavior* kartice prozora *Actions…* samog *List Boxa.*

Dakle, u polje *Range Limit Field* dodajemo *DOST:Sifra_ev_lista* tj. dio primarnog ključa koji dijeli sa nadtipom entiteta. U polju *Range Limit Type* biramo *File Relationship* da bi postigao upravo željeno, a to je da će se sada prikazivati samo dostave kojima je primarni ključ Sifra_dostave i Sifra_ev_lista kojeg korisnik ažurira u ovoj proceduri.



Slika 50. PopisDostava - *Range Limit Field*

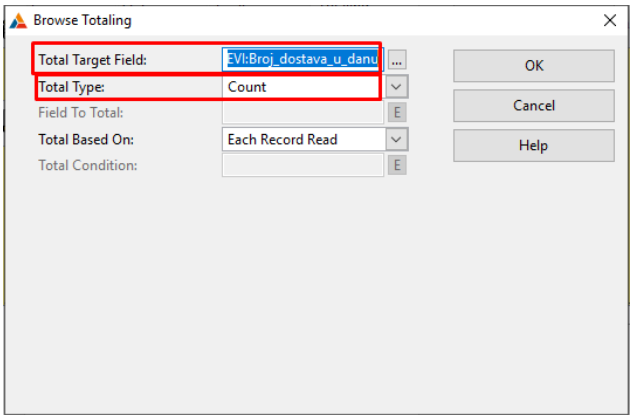
Da bi korisnik mogao ažurirati PopisDostava potrebno je još iz kartice *Control Templates* povući *BrowseUpdateButtons* tj. gumbe za unos, izmjenu i brisanje. Gumbima mijenjamo dizajn te prvog odabiremo desnom tipkom i biramo *Actions*…. U polje *Update Procedure* upisujemo *AzuriranjeDostava* i spremamo. Automatski se taj unos prenosi i na ostale gumbe.



Slika 51. PopisDostava - *Browse Update Buttons*

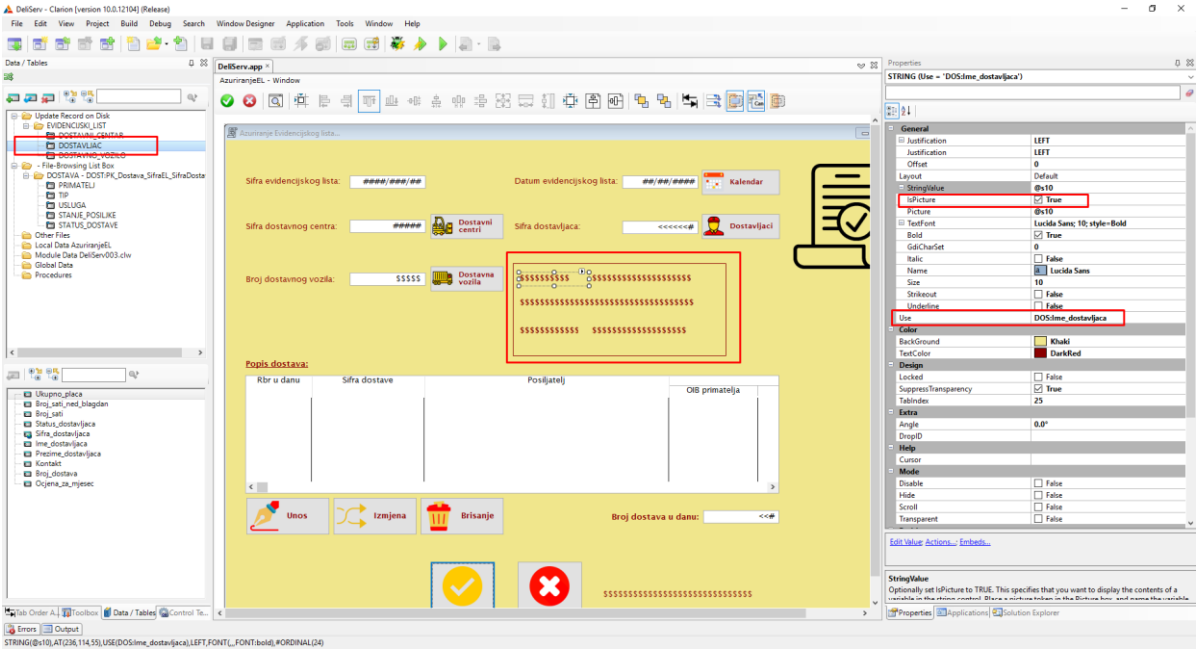
Ispod *List Boxa* dodajemo još stupac iz tablice EVIDENCIJSKI_LIST pod nazivom *Broj_dostava_u_danu*. Za to moramo postaviti opciju *Totaling* koja će brojati dostave iz *List Boxa* i broj upisivati u ovo polje.

Dakle, odabiremo *List Box* desnim klikom, otvaramo *Actions…*, biramo karticu *Totaling,* biramo gumb *Insert* te, u novom prozoru, u polju *Total Target Field* odabiremo *EVI:Broj_dostava_u_danu*. Ispod toga, u polju *Total Type,* postavljamo vrijednost *Count*. Spremamo sve i izlazimo.



Slika 52. PopisDostava - *Totaling*

U prozoru ćemo još dodati jedan okvir u kojeg će se upisivati ukratko podaci koje je korisnik odabrao za ovaj Evidencijski list. Dakle, kad odabere šifru dostavljača, u tom okviru prikazat će se ime i prezime, kad odabere šifru vozila, pokazat će se marka vozila itd. To radimo na način da, kao i kod *List Boxa,* u kartici *Data/Tables* dodam povezane tablice. Zatim u taj novi okvir (*Toolbox* → *Box),* također iz *Toolboxa*, vučemo *String* i u *Propertiesu* odabiremo opciju *IsPicture*. Ona postavlja novi *string* u dinamički oblik tj. da se mijenja ovisno o nekom parametru. Da bi to funkcioniralo, u polju *Use* odabiremo pripadajući stupac određene tablice.



Slika 53. ***AzuriranjeEL – Window***

2.3.6. Kreiranje procedure *AzuriranjeDostava*

AzuriranjeDostava izrađuje se kao i ostala ažuriranja. Bira se tip procedure, prevode se sistemske poruke, u *Windowu* se bira tablica DOSTAVA te se stupci vuku u sami prozor proizvoljnim redoslijedom. Biti će nam potrebne i povezane tablice, pa i njih dodajemo te koristimo njihove stupce.

Za *Sifru_tipa*, *OIB_primatelja*, *Stanje_posiljke* i *Status_dostave* dodajemo *Lookup*, kako smo prije opisali, preko samog polja i preko gumba sa strane.

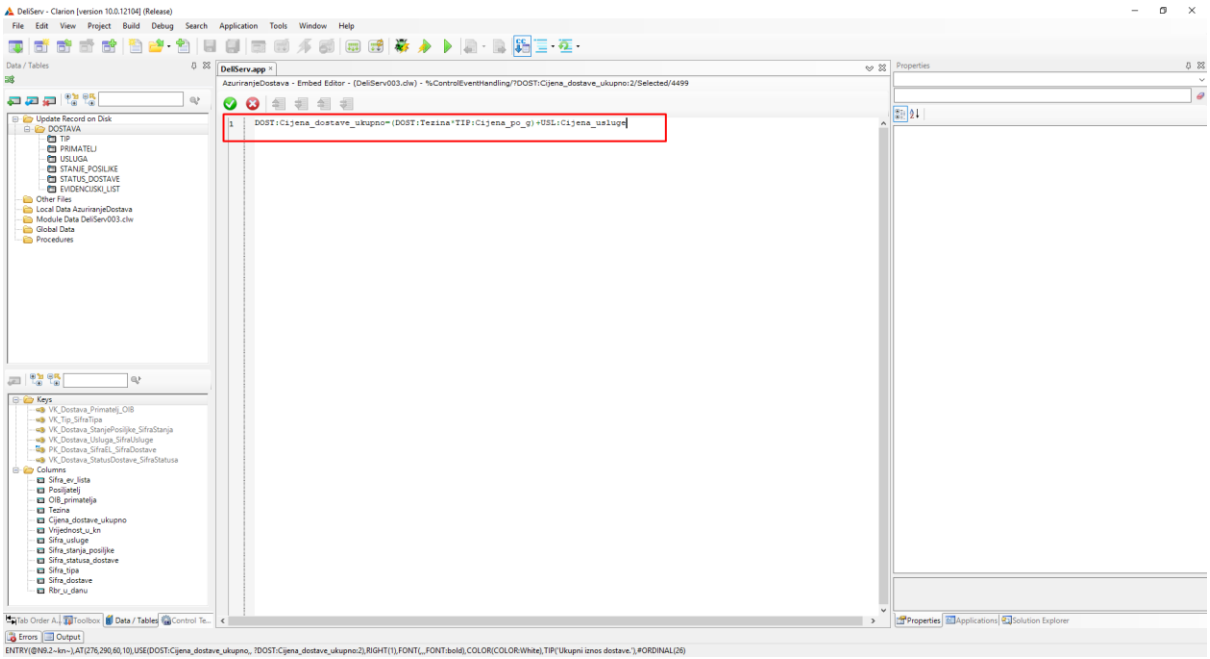
U polju Cijena_dostave_ukupno bit će nam potreban složeniji izračun pa polje biramo desnom tipkom i odabiremo *Embeds*….U prikazu situacija biramo *Selected* te u *Source* upisujemo sljedeći izraz: *DOST:Cijena_dostave_ukupno=(DOST:Tezina*TIP:Cijena_po_g)* +

USL:Cijena_usluge. Na taj način osiguravamo da ukupni iznos te dostave uvijek bude umnožak težine i cijene po gramu plus cijena usluge.

Da ne moramo pisati napamet izraz, Clarion sa strane nudi ponovno dodavanje potrebnih povezanih tablica i njihovih stupaca, koje onda, dvoklikom, dodajemo u polje.

Spremamo sve i vraćamo se u *Window*.

U prozor ponovno dodajemo dinamičke *stringove* za lakše snalaženje sa šiframa te kroz *Properties* uređujemo prozor. Procedura je dovršena.



Slika 54. AzuriranjeDostave - Cijena_dostave_ukupno

2.3.7. Kreiranje procedure *PopisDostavljaca*

PopisDostavljaca kreiramo na isti način kao i ostale popise. Nakon riješenih *Actions…* i dizanja *Windowa* te dodane tablice i njezinog primarnog ključa uređujemo *List Box* preko *List Box Format*….Kad smo dodali sve stupce vraćamo se u *Window,* te dodajemo pretraživanje sa pripadnim gumbom.

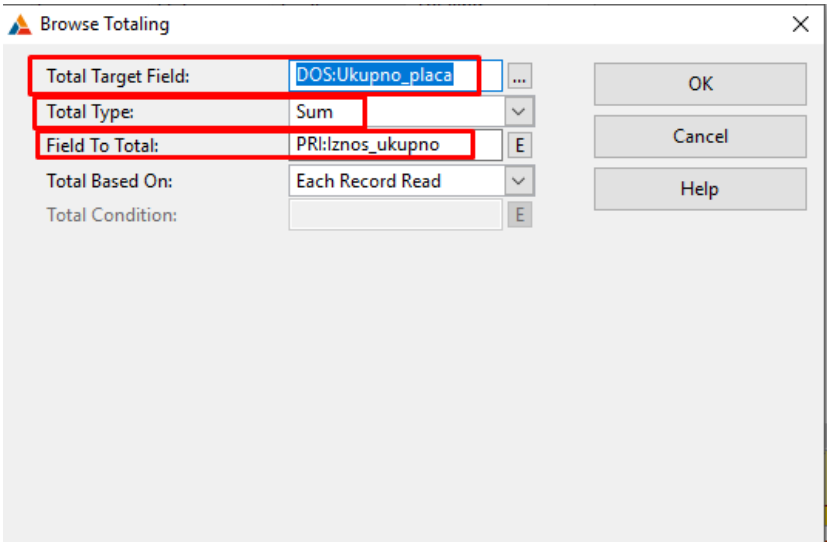
Sada ćemo u produžetku prozora dodati još jedan *List Box.* On će sadržavati samo jedan stupac, a to je Iznos_ukupno tablice PRIMA. Kako je to agregacija biti će prikazana u ovoj proceduri. U taj stupac upisivat će se rezultat množenja odabrane satnice i broja sati dostavljača. Ovo ćemo još dodatno objasniti u opisu kreiranja procedure *AzuriranjePrima*.

List Boxu određujemo tablicu i primarni ključ , dodajemo i oblikujemo taj jedini stupac te po povratku u *Window,* postavljamo *Range Limit* na prethodno opisani način kako bi se podaci prikazivali samo za odabranog dostavljača.

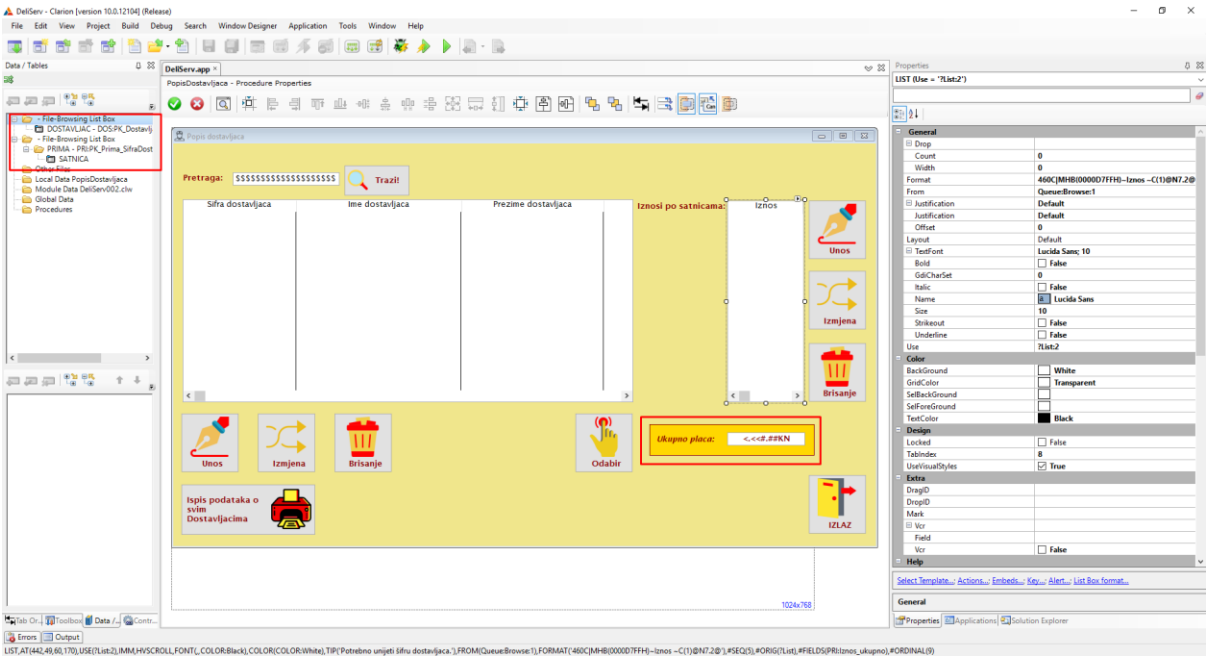
Sada još moramo, iz kartice *Control Templates*, povući *BrowseUpdateButtons,* pokraj samog *List Boxa,* te ih postaviti da pozivaju proceduru *AzuriranjePrima*.

Ispod *List Boxa* dodajemo u prozor, iz tablice DOSTAVLJAC, stupac Ukupno_placa. Ovdje će se upisivati zbroj iznosa po satnicama odabranog dostavljača. Desnim klikom biramo drugi *List Box,* otvaramo *Actions…*, u kartici *Totaling* biramo *DOS:Ukupno_placa* te tip postavljamo na *Sum.* Ovdje još moramo odabrati koji stupac će se zbrajati, pa u polje *Field to Total* postavljamo PRI:Iznos_ukupno. Kroz *Properties* ćemo još, za polje *Ukupno_placa*,

označiti *checkbox Read Only* kako bi osigurali da se vrijednost ne može ručno promijeniti. Spremamo sve i procedura je dovršena.



Slika 55. *PopisDostavljaca* - Ukupno_placa - *Totaling*



Slika 56. *PopisDostavljaca* - *Window*

2.3.8. Kreiranje procedure *AzuriranjePrima*

Finalno, dolazimo do procedure *AzuriranjePrima*. Iako sam prozor ima vrlo malo elemenata, traži dosta zahtjevan izračun u pozadini.

Nakon standardnog kreiranja *Update* procedure, u prozor dodajemo stupce *Sifra_satnice* i *Iznos_ukupno*. Stupac Sifra_dostavljaca se ne dodaje zbog *Range Limit* opcije koju smo odabrali u proceduri *PopisDostavljaca*.

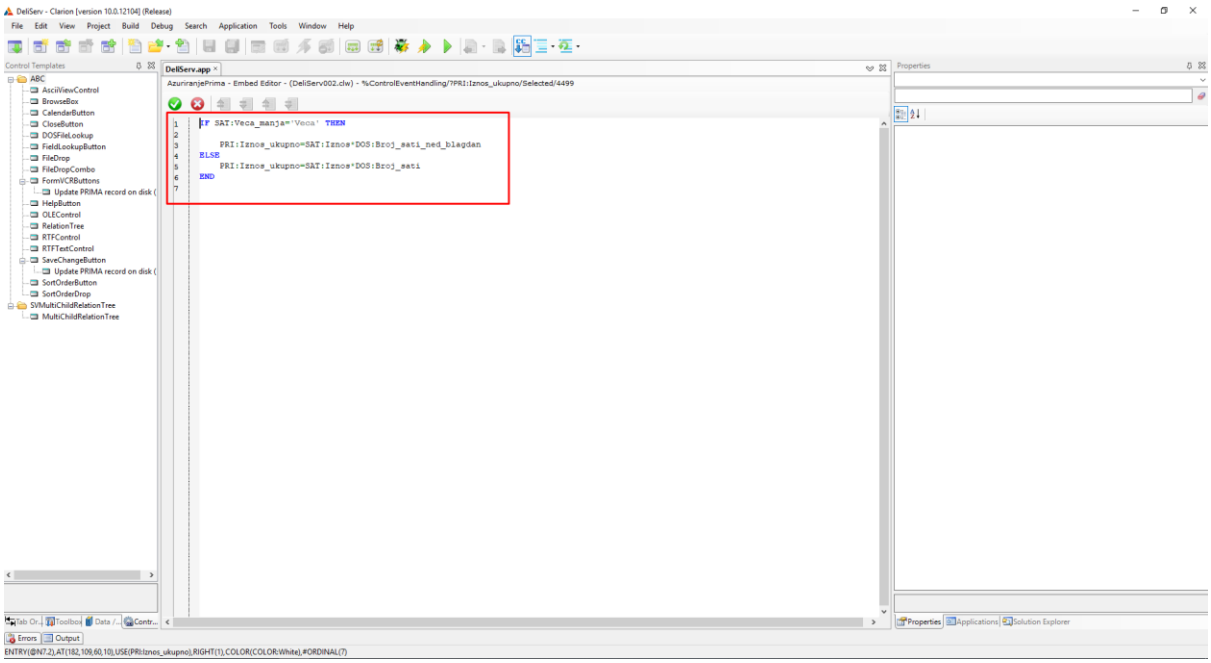
Na polje *Sifra_satnice* dodajemo *Lookup,* te pripadni gumb sa strane. Nakon što je odabrana satnica prikazat će se, kroz dinamičke s*tringove*, naziv i iznos satnice.

Sada je potrebno definirati izračun. Polje Iznos_ukupno odabiremo desnim klikom i ulazimo u *Embeds….* Pod situacijom *Selected* (jer će se izračun obaviti nakon odabira polja) u *Source* sučelju kreiramo kod za izračun u Clarion programskom jeziku.

```
IF SAT:Veca_manja='Veca' THEN

PRI:Iznos_ukupno=SAT:Iznos*DOS:Broj_sati_ned_blagdan
ELSE
PRI:Iznos_ukupno=SAT:Iznos*DOS:Broj_sati
END
```

Izračun se sastoji od toga da će aplikacija provjeriti je li *checkbox Veca_manja* u tablici SATNICA označen tj. ima li stupac za tu satnicu vrijednost „Veca“. Ako ima, množit će broj sati dostavljača nedjeljom i praznikom i iznos odabrane satnice. Ako nema, tj. *checkbox* nije označen, ukupni iznos će biti umnožak broja redovnih sati i iznosa satnice. Na taj način jedan dostavljač može imati više satnica, a iznosi se prikazuju u *List Boxu,* u proceduri *PopisDostavljaca*, te se zbrajaju i ukupni iznos se prikazuje u polju *Ukupno_placa*.



Slika 57. Iznos_ukupno - *Embeds...*

Sada su sve *Browse* i *Update* procedure dovršene. Još nam ostaje dodati tri vrste izvještaja te jednu malu proceduru sa opisom aplikacije kako bi projekt bio dovršen.

2.3.9. Kreiranje jednostavnog izvještaja – procedura *PopisDostavljaca*

Izvještaj je prikaz određenih podataka u određenom obliku obično u formatu za ispis na A4 papir. Kreirat ćemo jedan jednostavan izvještaj u proceduri *PopisDostavljaca*, jedan „preko gumba“ u *PopisVozila*, te jedan složeni u *PopisEL*.

Za početak ćemo, u alatnoj traci Glavnog Izbornika, dodati padajući izbornik *Izvještaj* te elemente izbornika *Ispis dostavljaca* i *Ispis Evidencijskih listi po Dostavljacima*. Kao i za prijašnje procedure, otvaramo na svakom elementu *Actions…* te podešavamo da se pozove procedura (*Call a Procedure),* imena (*Procedure Name*) *ReportDostavljac* i da se za nju opet otvara novi *thread* sa 50 000 procesorskih niti. Spremamo i izlazimo.

Sada se u *Application Tree* pojavljuje ta nova procedura (*ToDo).* Biramo je dvoklikom, i odabiremo tip procedure *Report (Paper size A4 – Portrait)*. Ovdje ne diramo *Actions…* već prvo *Window* u kojem oblikujemo dizajn prozora koji će se kratkotrajno prikazati dok aplikacija bude kreirala novi izvještaj. Kada je to dovršeno, idemo na gumb *Report*.

Otvara se Clarionovo sučelje za oblikovanje izvještaja. Softver nudi mnogo opcija, načina i predložaka ali mi ćemo se držati jednostavnog.

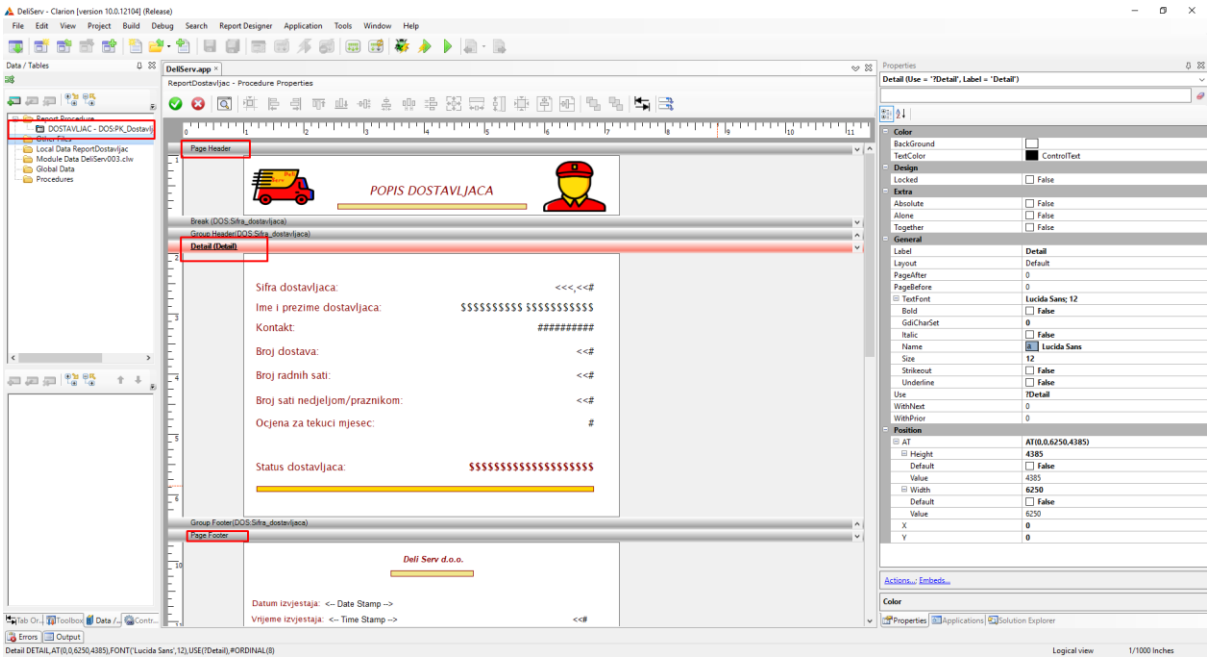
Za početak, kao u *Browse* procedurama, moramo, u kartici *Data/Tables,* odabrati tablicu za koju radimo izvještaj te njezin primarni ključ.

Zatim ćemo oblikovati zaglavlje i podnožje izvještaja. U dijelu *Page Header* kreiramo naslov dokumenta i ukrašavamo ikonama iz aplikacije. U *Page Footer* dodajemo još jednom naziv firme, te datum i vrijeme izrade izvještaja i broj stranice. To ćemo napraviti tako da iz kartice *Control Templates* povučemo elemente *ReportDateStamp, ReportTimeStamp* i *ReportPageNumber.* Uz njih ćemo još, iz kartice *Toolbox*, dodati *stringove* koji će ih opisivati (Datum izvještaja, Vrijeme izvještaja).

Finalno, prelazimo na *Detail*. Tu ćemo isto kreirati *stringove* koji će opisivati podatke u izvještaju. Oni su, dakle, statični i ostaju nepromijenjeni, a onda ćemo na odgovarajuća mjesta povući stupce iz tablice. To su zapravo dinamički *stringovi* koji će se izmjenjivati prema podacima o dostavljaču.

Oblikujemo još dizajn i spremamo sve.

Još je potrebno u proceduri *PopisDostavljaca*, ispod *List Boxa*, dodati gumb za ispis. Iz *Toolboxa* vučemo *button*, oblikujemo mu dizajn, u polje *Use (Properties)* upisujemo *?Print,* odabiremo ga desnom tipkom, otvaramo *Actions…* te postavljamo da poziva proceduru *ReportDostavljac*. Spremamo proceduru i prvi izvještaj je dovršen.



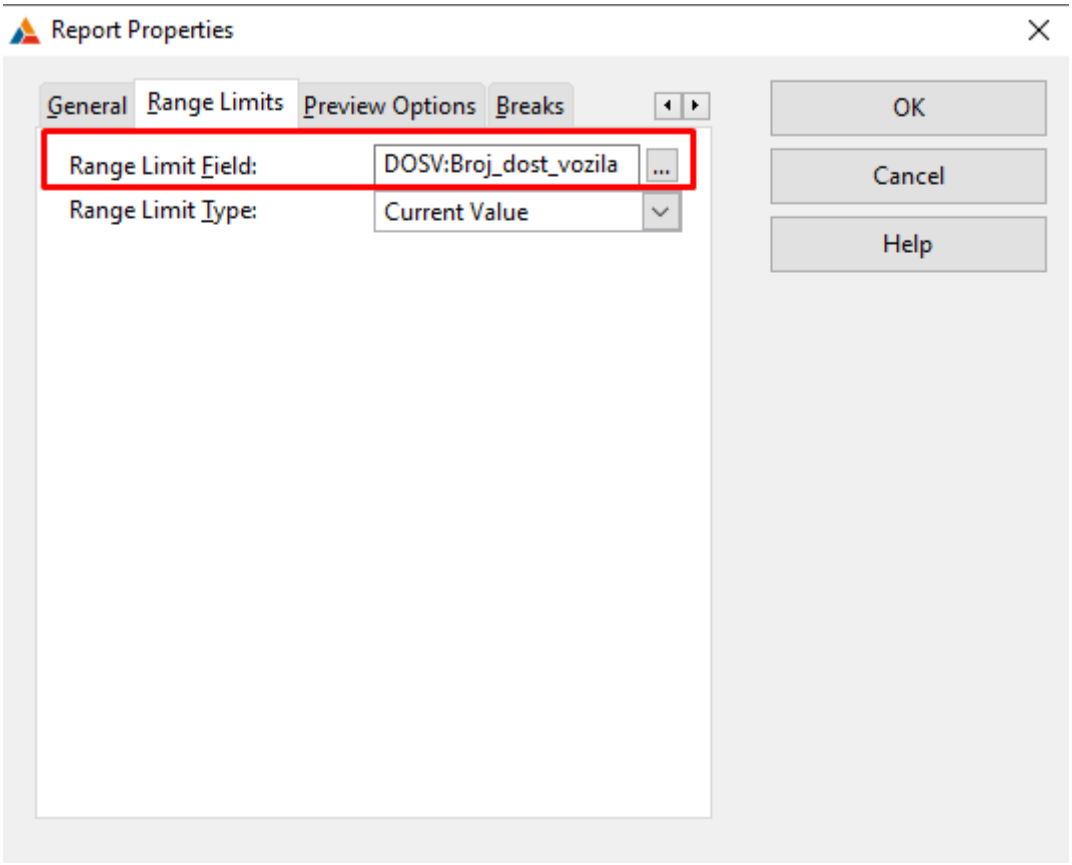
Slika 58. *PopisDostavljaca* - jednostavni izvještaj

2.3.10. Kreiranje izvještaja „preko gumba“ – procedura *PopisVozila*

Izvještaj „preko gumba“ znači da se pritiskom gumba za ispis pokraj *List Boxa*, ne kreira izvještaj za sve zapise u tablici, nego samo za onaj koji je trenutno odabran. Iz tog razloga se ovaj izvještaj ne uključuje u padajući izbornik alatne trake na početnom zaslonu.

Da si olakšamo, kopirat ćemo proceduru *ReportDostavljac*, preimenovati je u *ReportOdabranoVozilo*, sam *Report* izmjenjujemo na način da, u *Data/Tables,* odabiremo tablicu DOSTAVNO_VOZILO, i njezin ključ. Mijenjamo *stringove* da pašu novim stupcima te dodajemo dinamičke *stringove* iz tablice. Spremamo i vraćamo se u *Procedure Properties*. Ovdje ćemo koristiti *Actions..*, u novom prozoru biramo tipku *Report Properties* te, pod karticom *Range Limit*, u polje *Range Limit Field* biramo stupac *DOSV:Broj_dost_vozila*. Na taj način osiguravamo da se prikazuju podaci samo za odabrano vozilo. Spremamo sve i izlazimo.

U proceduri *PopisVozila* ponovno dodajemo gumb za ispis koji će pozvati proceduru *ReportOdabranoVozilo*. Spremamo sve kreirano.



Slika 59. ***Report Properties - Range Limit***

2.3.11. Kreiranje složenog izvještaja – procedura *PopisEL*

Složeni izvještaj kreirat ćemo na način da se ispišu sve Evidencijske liste, ali grupirane po dostavljačima. Ponovno kopiramo proceduru *ReportDostavljac* da ne gubimo vrijeme na dizajn. Preimenujemo je i otvaramo *Report*.

U kartici *Data/Tables* prvo biramo tablicu DOSTAVLJAC i njezin primarni ključ, a onda još dodajemo tablicu EVIDENCIJSKI_LIST te pripadni primarni ključ.

Odabiremo područje *Detail* desnim klikom te biramo *Sorrounding Break.* Otvara se novi prozor u kojem, u polju *Variable,* biramo *DOS:Sifra_dostavljaca* te dajemo naziv „Break1DostavljacSifra“.To je prvi prijelom ovog izvještaja. Zatim ponovno biramo *Detail* i ponovno kroz *Sorrounding Break* otvaramo prozor te dodajemo varijablu (*Variable*) *EVI:SifraEL*. Prema tome i dajemo naziv „break1ELSifra“. Sada imamo i drugi prijelom izvještaja.

Značenje prijeloma je da se to određeno područje ponavlja tj. ponovno prikaže svaki puta kada se vrijednost promjeni. Sada kad imamo definirane *breakove* (prijelome), prvo desnim klikom na oznaku područja *Break (EVI:SifraEL)* dodajemo *Group Header* i *Group Footer.* To će biti zaglavlje i podnožje koje će se ponavljati sa svakom iteracijom. U *Header* dodajemo *stringove* „Sifra Evidencijske liste:“ i „Datum Evidencijske liste:“ . pored njih, iz *Data/Tables,* povlačimo dinamičke *stringove Sifra_ev_lista* i *Datum_ev_lista.*

Niže ćemo oblikovati zaglavlje podataka koji će se prikazivati o svakoj dostavi. Dodajemo ponovno statičke *stringove*: Sifra dostave, Sifra tipa, Sifra usluge, Cijena ukupno, Stanje, Status. Ispod još dodajemo debelu liniju kreiranu od *Box* elementa kartice *Toolbox.*

Sada u *Detail* ubacujemo stupce (dinamičke *stringove*) ispod pripadajućih naslova. Dakle *Sifra_dostave* ispod Sifra dostave itd. Dodajemo povezane tablice kako bi mogli povući sve potrebne stupce.

Potrebno je još dodati *Group Header* i *Group Footer* za nadgrupu tj. za *Break (DOST:Sifra_dost).* U *header* ćemo dodati dinamičke *stringove* za stupce *Sifra_dostavljaca*, *Ime_dostavljaca*, *Prezime_dostavljaca*, *Kontakt* te *Status_dostavljaca*. Sve to uokvirujemo elementom *Box* iz *Toolboxa* kojeg uređujemo na način da se vidi samo vanjski rub.

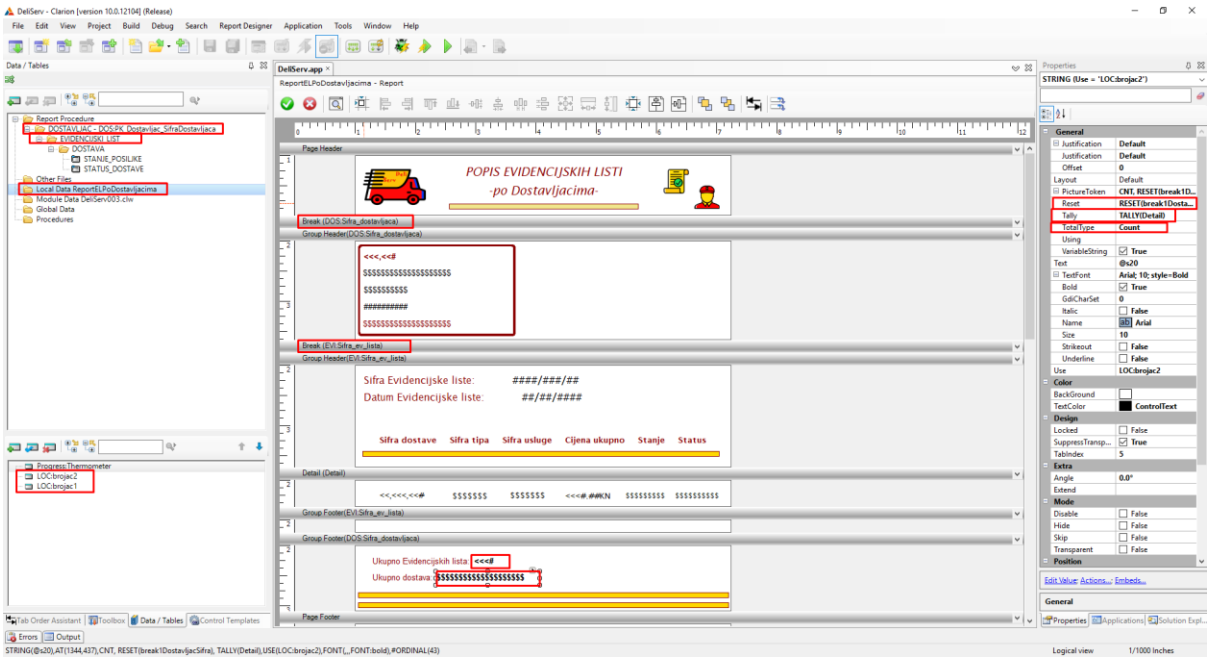
U *Footer* dodajemo statične *stringove* „Ukupno Evidencijskih lista:“ i „Ukupno dostava:“ . tu će se prikazivati brojevi Evidencijskih lista za koje je prijavljen pojedini dostavljač te ukupan broj dostava koje je općenito pojedini dostavljač obavio. Međutim, kako za to ne postoje stupci niti u jednoj tablici, dodajemo lokalne stupce. Kao što smo u aplikaciji dodavali lokalni stupac za pretraživanje, tako ćemo i ovdje dodati *LOC:brojac1* i *LOC:brojac2*. Povlačimo ih u *Footer*. Za ukupan broj Evidencijskih lista, u *Propertiesu* dinamičkog *stringa* postavljamo polje *TotalType* na *Count*, iznad toga polje *Tally* na *break1ELSifra* (Svaki put kada aplikacija naiđe na prijelom po šifri Evidencijskog lista, inkrementirati će se Ukupan broj Evidencijskih lista za 1) te konačno polje *Reset* na *break1DostavljacSifra* (brojanje kreće ispočetka kada aplikacija naiđe na prijelom po šifri dostavljača).

Za Ukupno dostava, ponovno *TotalType* postavljamo na *Count, Tally* ovaj put na *Detail* (prebrojati će se svi zapisi u *Detail* dijelu), a *Reset* na *break1DostavljacSifra* (i ovdje se broj vraća na nulu kada se promjeni dostavljač).

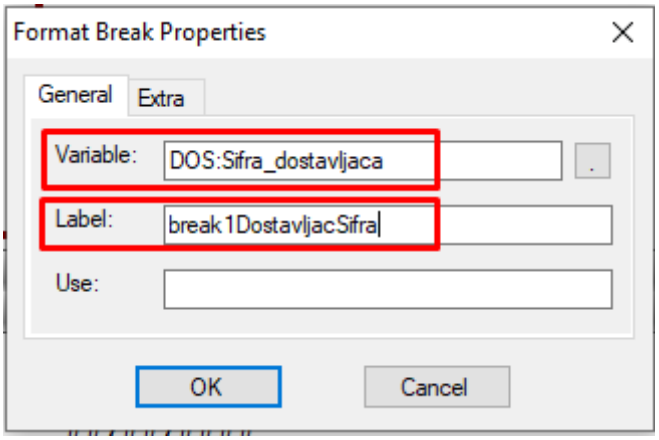
Zadnji detalj koji treba urediti je postavka *Inner Join* koja će spriječiti pojavljivanje nultih zapisa tj. prikazivanje dostavljača koji nema svojih Evidencijskih lista ni dostava.

U *Data/Tables* označavamo tablicu EVIDENCIJSKI_LIST te biramo tipku *Change* nas vrhu. U novom prozoru označavamo *checkbox Inner Join*. Biramo OK i spremamo sve.

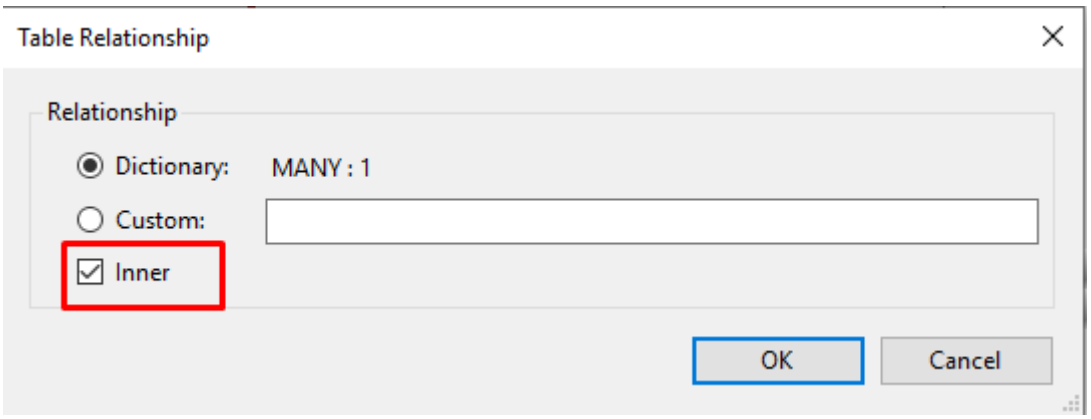
U proceduri *PopisEL* dodajemo gumb za ispis koji poziva proceduru *ReportELPoDostavljacima*. Izvještaj je gotov.



Slika 60. ***PopisELPoDostavljacima*** **- složeni izvještaj –** ***Report***



Slika 61. ***Sorrounding Break***



Slika 62. ***Inner Join***

2.3.12. Kreiranje procedure *OAplikaciji*

Za kraj, dodat ćemo jednostavnu proceduru koja će prikazivati prozor s kratkim tekstom o samom projektu. U *Application Tree* dodajemo novu proceduru, tip postavljamo na *FORM for View-Only* te otvaramo *Window.* Prozor uređujemo standardno kao i do sada te dodajemo *Prompt* iz kartice *Toolbox*. U njega upisujemo kratki tekst te dodajemo slike s grbovima kao ukras. Spremamo sve.

U proceduri *GlavniIzbornik* u alatnoj traci dodajemo element *O aplikaciji* kojem, u *Actions…,* postavljamo da poziva proceduru *OAplikaciji*. Također, dodajemo nekoliko *separatora* da odvojimo ovaj element od ostalih. Spremamo i izlazimo.

Aplikacija je ovime dovršena. Sadrži sve potrebne elemente i radi logički ispravno.

3. Zaključak

U ovom se radu nastojalo na detaljan, a opet ne preširok način, objasniti izradu poslovne aplikacije nad relacijskom bazom podataka. Kroz prvotno izlaganje teorije koja je u pozadini pa onda kroz samo praktično korištenje alata Clarion, cilj je bio približiti proces izrade baze podataka. Rad sadrži veliki broj slika upravo iz razloga da se svi važni koraci prikažu u samom softveru, te na taj način osigura shvaćanje učinjenog.

Kao što je navedeno u uvodu, ovakve poslovne aplikacije su od presudne važnosti za poslovne sustave. Za dobro poslovanje, ključno je imati dobro razvijenu, detaljnu i temeljitu aplikaciju koja će biti posrednik između sustava i njegove baze podataka. Naravno, kao i većina računalnih programa, aplikacija može biti žrtva hakerskog napada ili modificirana od strane zloćudnog softvera (*malware)* ili virusa. Također, problem može biti u samoj aplikaciji, ako je ona napravljena nekvalitetno. Na taj način, zbog programerskih pogrešaka ili logički neispravnih postavki, može doći do gubitka podataka, a s time i poslovnih i financijskih problema samoj tvrtki.

Aplikacija koje je izrađena za potrebe ovoga rada može se smatrati solidnom. Naravno, u puno aspekata ona se može nadograditi i poboljšati. Možda bi trebalo dodati još nekoliko tablica da obuhvaća veći spektar podataka. Zatim bi se mogli poboljšati izračuni. Trenutni izračun plaće temeljen je na studentskoj plaći gdje se izračun sastoji samo od broja sati i satnice. Poznato je, dakako, da je pravi izračun plaće daleko složeniji i kompliciraniji. Zatim postoje detalji u samoj aplikaciji koje bi trebalo napraviti bolje i praktičnije. Primjer su polja koja sadrže broj nečega (Broj vozila, Broj dostavljača…). Vrijednosti u tim poljima trebala bi se mijenjati dinamički, ovisno o zapisima u drugim tablicama. To opet traži bolje i dublje poznavanje Clarionovih mogućnosti i funkcionalnosti.

Što se samog softvera tiče, studenti većinom njime nisu zadovoljni, smetaju im povremena rušenja programa ili neshvatljive greške koje sam spomenuo ranije. U izradi ovog projekta, Clarion se nije pokazao toliko lošim alatom. Kakva mu je i zadaća, on ubrzava i olakšava proces izrade aplikacije, i kroz korištenje za izradu ove aplikacije, u nekim mogućnostima čak pozitivno iznenađuje. Istina je da i on treba nekoliko nadogradnji, ali kao alat ispunjava očekivanja i potrebe *developera*.

Na kraju, vidljivo je koliki dio posla obavljaju RAD alati. Najdugotrajniji dio izrade ove aplikacije bio je onaj prvi, „misaoni“. Za razliku od timova *developera* koji inače rade na ovakvim projektima, ovdje je radila jedna osoba. Zato je proces stvaranja ideje, nacrta relacijskog modela i dijagrama EV uzeo puno vremena. Međutim, jednom kad je to bilo gotovo, alat Clarion je preostali dio posla veoma ubrzao. Zato se RAD alati smatraju ključnim pomagačem svakome tko želi kreirati kvalitetnu aplikaciju u razumnom roku.

Još je potrebno istaknuti i da se, kroz izradu ove aplikacije, neke stvari i naučilo. Kao što je navedeno, od nekih do sad nepoznatih funkcionalnosti Clarion alata koji se nisu koristili u nastavi, a vrlo su korisne, do načina razmišljanja potrebnog za kvalitetnu izradu aplikacije. Zato ovaj rad nije samo predstavljao zadatak koji se mora obaviti, nego i poučno iskustvo. Bez obzira na osobne profesionalne interese, korisno je dodatno izučiti se u radu sa bazama podataka i relacijskim aplikacijama koje se njima koriste.

Literatura

[1] Pavlić Mile, Oblikovanje baze podataka, Odjel za informatiku Sveučilišta u Rijeci, Rijeka, 2011.godina

[2] Rapid Application Development Tools – 5 Popular RAD Tools in 2019, Kissflow. preuzeto 18.9.2019. sa https://kissflow.com/rad/rapid-application-development-tools-most-popular-in-2018/

Popis slika

Prilozi

Zadatak za završni rad

Rijeka, 7.6.2019

Zadatak za završni rad

Pristupnik: Luka Štorić

Naziv završnog rada: Poslovna aplikacija za dostavnu službu nad relacijskom bazom podataka - Clarion

Naziv završnog rada na eng. jeziku: Delivery service business application with relational database - Clarion

Sadržaj zadatka:

BP predstavlja kolekciju podataka, ograničenja i operacija koji reprezentiraju neke aspekte realnoga svijeta. Dakle, BP je model neke aplikacijske domene. Od studenta se očekuje da kreira poslovnu aplikaciju za obradu određenih poslovnih dokumenata vezanih uz rad dostavne službe. Aplikacija mora raditi nad relacijskom BP u navedenoj aplikacijskoj domeni (dostavna služba). Potrebno je napraviti model podataka i odgovarajuću BP u programskom alatu Clarion (8, 9 ili 10).

Mentor

prof. dr. sc. Patrizia Poščić

Voditelj za završne radove

doc. dr. sc. Miran Pobar

Zadatak preuzet: 27.6.2019.

(potpis pristupnika)