Sveučilište Jurja Dobrile u Puli

**Sustav za isplatu plaća u auto salonu**

**Tim 9**

**Mentor:** doc. dr. sc. Goran Oreški

**Predmet:** Baze podataka 2

**Izradili:**

Alan Bubalo

Mihael Blažević

Alen Orbanić

Luka Brodarič

Thomas Siard

Ernest Dudaković

Pula, 17.01.2022.

SADRŽAJ

[1. UVOD 3](#_Toc92913325)

[2. ER DIJAGRAM 4](#_Toc92913326)

[2.1. OPIS ER DIJAGRAMA 5](#_Toc92913327)

[2.2. EER DIJAGRAM 6](#_Toc92913328)

[3. DETALJAN OPIS SVIH TABLICA, ATRIBUTA I DOMENA S KOMENTAROM 7](#_Toc92913329)

[3.1 Tablica – pozicija 7](#_Toc92913330)

[3.2 Tablica – zaposlenik 8](#_Toc92913331)

[3.3. Tablica – kupac 9](#_Toc92913332)

[3.4. Tablica – klasa 10](#_Toc92913333)

[3.5. Tablica – automobil 10](#_Toc92913334)

[3.6. Tablica – servis 11](#_Toc92913335)

[3.7. Tablica – placanje 11](#_Toc92913336)

[3.8. Tablica – racun 12](#_Toc92913337)

[3.9. Tablica – praznici 13](#_Toc92913338)

[3.10. Tablica – prisutnost 13](#_Toc92913339)

[4. UBACIVANJE PODATAKA U BAZU 14](#_Toc92913340)

[5. FUNKCIJE, POHRANJENE PROCEDURE, OKIDAČI I POGLEDI 15](#_Toc92913341)

[5.1. OKIDAČ: 15](#_Toc92913342)

[5.2. OKIDAČ: 16](#_Toc92913343)

[5.3. OKIDAČ 17](#_Toc92913344)

[5.4. FUNKCIJA 18](#_Toc92913345)

[5.5. FUNKCIJA 19](#_Toc92913346)

[5.6. FUNKCIJA 20](#_Toc92913347)

[5.7. FUNKCIJA 21](#_Toc92913348)

[5.8. PROCEDURA 22](#_Toc92913349)

[5.9. PROCEDURA 23](#_Toc92913350)

[5.10. AUTENTIFIKACIJA I AUTORIZACIJA 24](#_Toc92913351)

[5.11. OPTIMIZIRAN PLAN IZVOĐENJA UPITA 25](#_Toc92913352)

[5.12. POGLED 26](#_Toc92913353)

[5.13. POGLED 26](#_Toc92913354)

[5.14. POGLED 27](#_Toc92913355)

[5.15. POGLED 27](#_Toc92913356)

[5.16. POGLED 28](#_Toc92913357)

[6. PROGRAMSKO RJEŠENJE 29](#_Toc92913358)

[7. ZAKLJUČAK 30](#_Toc92913359)

[8. LITERATURA 31](#_Toc92913360)

**SADRŽAJ SLIKA**

[Slika 1 ER dijagram 4](#_Toc92913361)

[Slika 2 EER Dijagram 6](#_Toc92913362)

[Slika 3 pozicija (tablica) 7](#_Toc92913363)

[Slika 4 zaposlenik (tablica) 8](#_Toc92913364)

[Slika 5 kupac (tablica) 9](#_Toc92913365)

[Slika 6 klasa (tablica) 10](#_Toc92913366)

[Slika 7 automobil (tablica) 10](#_Toc92913367)

[Slika 8 servis (tablica) 11](#_Toc92913368)

[Slika 9 placanje (tablica) 11](#_Toc92913369)

[Slika 10 racun (tablica) 12](#_Toc92913370)

[Slika 11 praznici (tablica) 13](#_Toc92913371)

[Slika 12 prisutnost (tablica) 13](#_Toc92913372)

[Slika 13 Umetanje vrijednosti u tablicu "zaposlenik" 14](#_Toc92913373)

[Slika 14 Okidač 1 15](#_Toc92913374)

[Slika 15 Okidač 2 16](#_Toc92913375)

[Slika 16 Okidač 3 17](#_Toc92913376)

[Slika 17 Funkcija 1 18](#_Toc92913377)

[Slika 18 Funkcija 2 19](#_Toc92913378)

[Slika 19 Funkcija 3 20](#_Toc92913379)

[Slika 20 Funkcija 4 21](#_Toc92913380)

[Slika 21 Procedura 1 22](#_Toc92913381)

[Slika 22 Procedura 2 23](#_Toc92913382)

[Slika 23 Autentifikacija i autorizacija 24](#_Toc92913383)

[Slika 24 Optimiziran plan izvođenja upita 25](#_Toc92913384)

[Slika 25 Pogled 1 26](#_Toc92913385)

[Slika 26 Pogled 2 26](#_Toc92913386)

[Slika 27 Pogled 3 27](#_Toc92913387)

[Slika 28 Pogled 4 27](#_Toc92913388)

[Slika 29 Pogled 5 28](#_Toc92913389)

# 1. UVOD

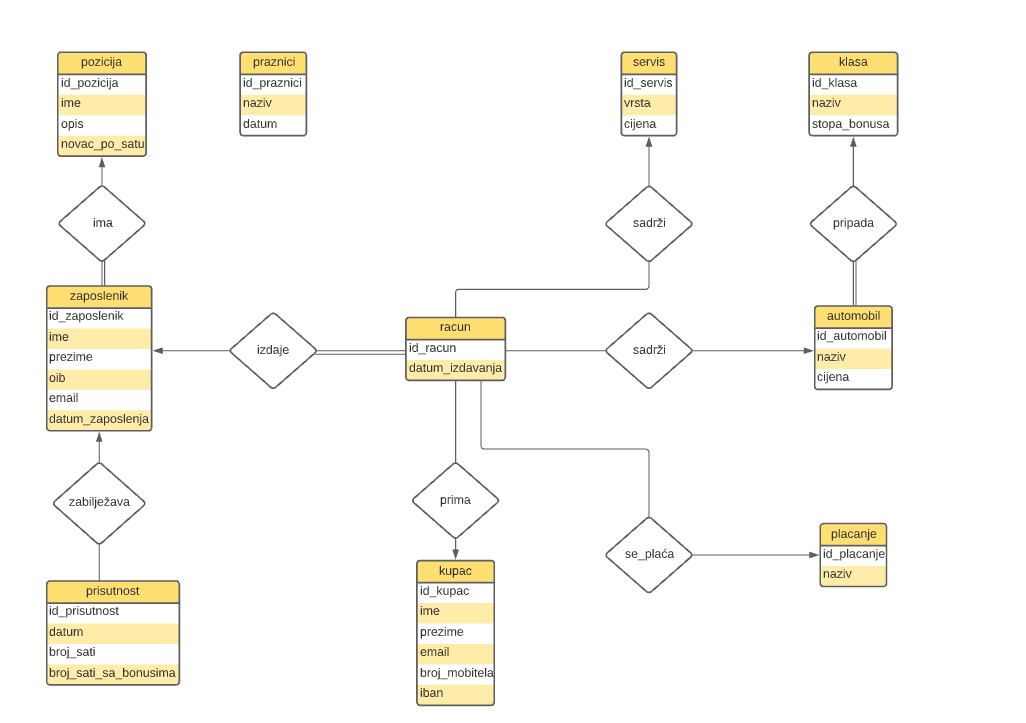
Izabrali smo temu „Sustav za isplatu plaća u auto salonu“. Imali smo članove tima sa iskustvom o ovoj temi pa smo ju odlučili obraditi.

Započeli smo zadavanjem relacija koje opisuju sustav za isplatu plaća u auto salonu . Navedene relacije su: *pozicija, klasa, automobil, kupac, zaposlenik, racun, praznici, placanje, servis i prisutnost.* Svaka pojedina relacija sadrži primarni ključ, tj. odgovarajući ID radi toga da podatci budu jedinstveni. Neke relacije u sebi imaju strane ključeve koji su povezani sa pripadajućim relacijama. U svakoj tablici naravno možemo promijeniti podatke ako bismo htjeli imati više ili manje opcija korištenjem tih relacija.

Cilj našeg projekta je bio da izradimo sustav za isplatu plaća u auto salonu. Pokušali smo da naša baza podatka bude što realnija i praktičnija tako da veze između relacija imaju više logike. Programe koje smo koristili za ovaj projekt je MySQL Workbench za bazu podataka, Lucidchart za izradu ER dijagrama i Visual Studio Code (HTML) za izradu web aplikacije. Za komunikaciju smo koristili Discord i Slack.

# 2. ER DIJAGRAM

Ovdje Vam je prikazan naš ER dijagram. Napravili smo ga vrlo jednostavnim i urednim kako biste Vi što bolje razumjeli. Na slici su Vam prikazani svi entiteti koje smo napravili i njihova povezanost. Neki entiteti su povezani drugačije od ostalih jer postoje različite mogućnosti. U nastavku je detaljno opisan svaki korak našeg ER dijagrama.



Slika 1 ER dijagram

## 2.1. OPIS ER DIJAGRAMA

1. ***zaposlenik 🡨🡪 pozicija:***

Veza „ima“ spaja tablice zaposlenik i pozicija. Spajamo vezom jedan na više zato što svaki zaposlenik ima svoju jednu poziciju, a tu istu poziciju mogu imati više zaposlenika. Svaki zaposlenik ima satnicu koja zapravo ovisi o poziciji. Veza ima dvostruku crtu jer svaki zaposlenik mora imati svoju poziciju.

1. ***zaposlenik 🡨🡪 prisutnost:***

Veza „zabilježava“ spaja tablice zaposlenik i prisutnost. Ova veza se isto spaja vezom jedan na više jer prisutnost zabilježava jedan zaposlenik, no taj isti zaposlenik može dolaziti više puta na posao, tj. zabilježiti svoju prisutnost.

1. ***praznici:***

Tablica praznici ne spaja se sa nijednom tablicom i koristimo ju kako bi dobili broj sati zaposlenika sa bonusima jer tijekom praznika je povećana plaća zaposlenicima.

1. ***zaposlenik🡨🡪 racun:***

Veza „izdaje“ spaja tablice zaposlenik i račun. Ovdje nam se opet pojavljuje veza s dvostrukom crtom jer ne postoji račun bez da ga je zaposlenik izdao. Također, račun izdaje samo jedan zaposlenik, a zaposlenik može izdati više računa.

1. ***racun 🡨🡪 servis:***

Veza „sadrži“ spaja tablice račun i servis. Imamo vezu jedan na više zato što račun sadržava najviše jedan servis, a servis se može pojaviti u više računa.

1. ***racun🡨🡪 kupac:***

Veza „prima“ spaja tablice račun i kupac. Kupac može dolaziti više puta da kupi automobil ili može doći na servis, a taj račun plaća samo jedan kupac. Stoga smo stavili vezu jedan na više.

1. ***racun 🡨🡪 placanje:***

Veza „se\_plaća“ spaja tablice račun i plaćanje. Račun se uvijek mora platiti i tvrtka pruža više plaćanja, no jedan račun se plaća na jedan način, tom logikom smo ustanovili da moramo spojiti vezom jedan na više.

1. ***racun🡨🡪 automobil:***

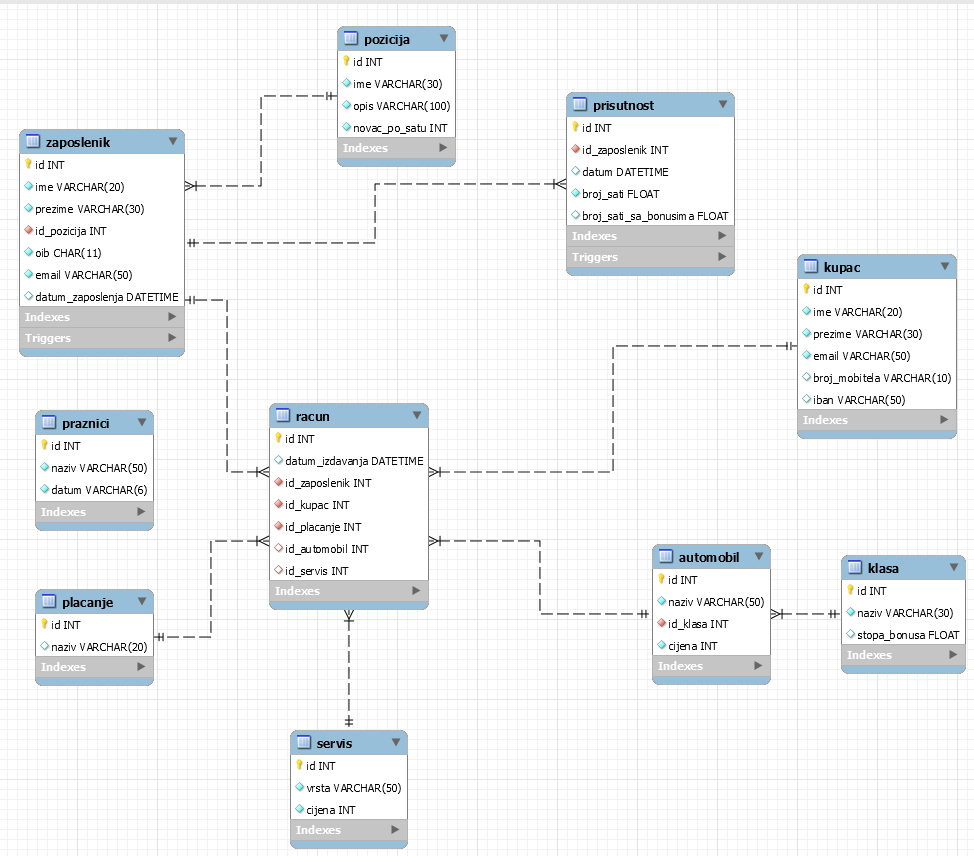
Veza „sadrži“ spaja tablice račun i automobil. Kao kod veze račun-servis, , automobil se može pojaviti na više računa, ali samo jedanput.

1. ***automobil🡨🡪 klasa:***

Veza „pripada“ spaja tablice automobil i klasa. Slično kao kod veze zaposlenik-pozicija, svaki automobil mora pripadati nekoj klasi, ali ima samo jednu klasu.

## 2.2. EER DIJAGRAM

Ovdje Vam je prikazan naš EER dijagram. Izrađen je u MySQL Workbench-u.



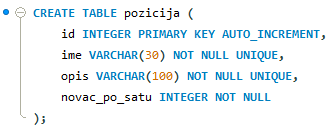
Slika 2 EER Dijagram

# 3. DETALJAN OPIS SVIH TABLICA, ATRIBUTA I DOMENA S KOMENTAROM

## 3.1 Tablica – pozicija

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* ime 🡪 VARCHAR (duljina 30 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* opis 🡪 VARCHAR (duljina znakova 100), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* novac\_po\_satu 🡪 INTEGER, ne smije biti prazan (NOT NULL)

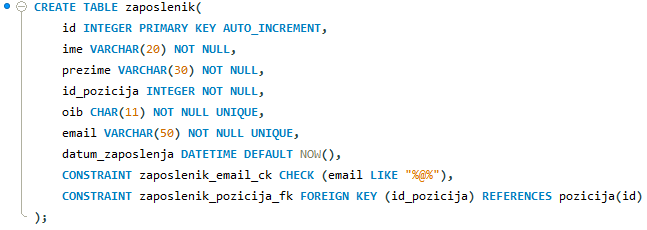


Slika 3 pozicija (tablica)

## 3.2 Tablica – zaposlenik

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* ime 🡪 VARCHAR (duljina 20 znakova), ne smije biti prazan (NOT NULL)
* prezime 🡪 VARCHAR (duljina 30 znakova), ne smije biti prazan (NOT NULL)
* id\_pozicija 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* oib 🡪 CHAR (duljina 11 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* email 🡪 VARCHAR (duljina 50 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* datum\_zaposlenja 🡪 DATETIME, datum kada se zaposlenik zaposlio, DEFAULT NOW(), ako podatak nije unesen automatski se postavi na sadašnje vrijeme

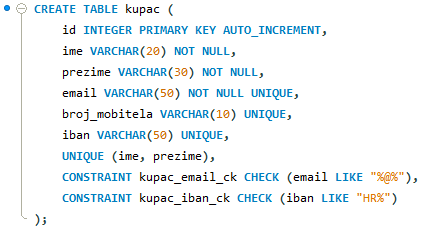


Slika 4 zaposlenik (tablica)

## 3.3. Tablica – kupac

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* ime 🡪 VARCHAR (duljina 20 znakova), ne smije biti prazan (NOT NULL)
* prezime 🡪 VARCHAR (duljina 30 znakova), ne smije biti prazan (NOT NULL)
* email 🡪 VARCHAR (duljina 50 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* broj\_mobitela 🡪 VARCHAR (duljina 10 znakova) i mora biti unikatan (jedinstven)
* iban 🡪 VARCHAR (duljina 50 znakova) i mora biti unikatan (jedinstven)

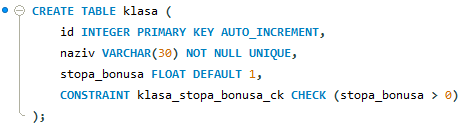


Slika 5 kupac (tablica)

## 3.4. Tablica – klasa

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* naziv 🡪 VARCHAR (duljina 30 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* stopa\_bonusa 🡪 FLOAT DEFAULT 1, automatski će staviti vrijednost 1, mora biti veći od 1

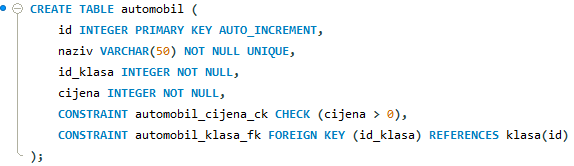


Slika 6 klasa (tablica)

## 3.5. Tablica – automobil

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* naziv 🡪 VARCHAR (duljina 50 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* id\_klasa 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* cijena 🡪 INTEGER, ne smije biti prazan (NOT NULL)

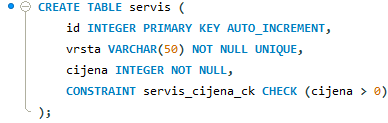


Slika 7 automobil (tablica)

3.6. Tablica – servis

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* vrsta 🡪 VARCHAR (duljina 50 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* cijena 🡪 INTEGER, ne smije biti prazan (NOT NULL)

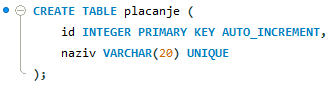


Slika 8 servis (tablica)

## 3.7. Tablica – placanje

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* naziv 🡪 VARCHAR (duljina 20 znakova) i mora biti unikatan (jedinstven)

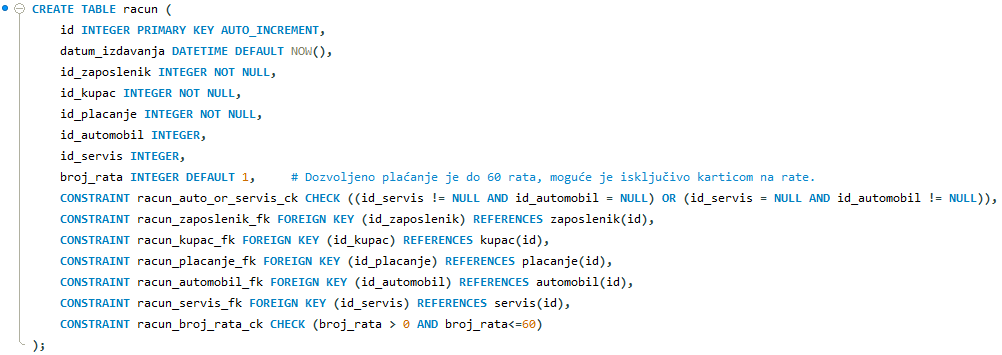


Slika 9 placanje (tablica)

## 3.8. Tablica – racun

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* datum\_izdavanja 🡪 DATETIME, datum kada se zaposlenik izdao račun, DEFAULT NOW(), ako podatak nije unesen automatski se postavi na sadašnje vrijeme
* id\_zaposlenik 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* id\_kupac 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* id\_placanje 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* id\_automobil 🡪 INTEGER
* id\_servis 🡪 INTEGER
* broj\_rata 🡪 INTEGER, DEFAULT 1, automatski će biti vrijednost 1

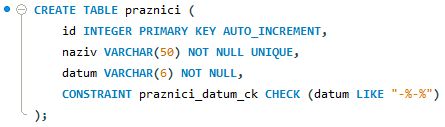


Slika 10 racun (tablica)

## 3.9. Tablica – praznici

Opis:

* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* naziv 🡪 VARCHAR (duljina 50 znakova), ne smije biti prazan (NOT NULL) i mora biti unikatan (jedinstven)
* datum 🡪 VARCHAR (duljina 60 znakova) i ne smije biti prazan (NOT NULL)

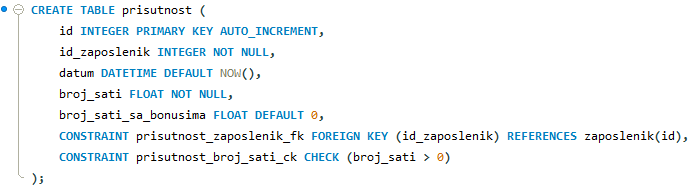


Slika 11 praznici (tablica)

## 3.10. Tablica – prisutnost

Opis:

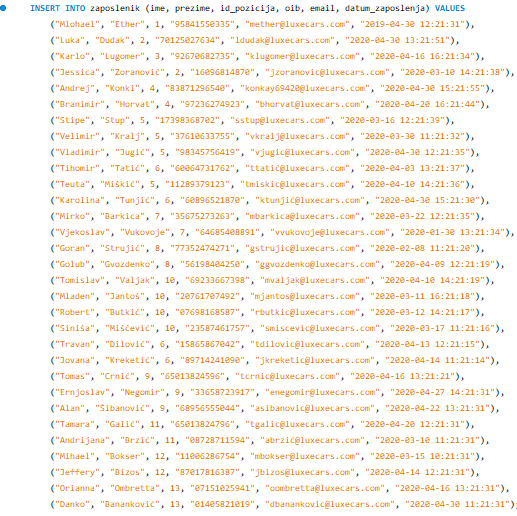
* id 🡪 INTEGER, primarni ključ, automatsko povećanje omogućuje automatsko generiranje jedinstvenog broja kada se novi zapis umetne u tablicu.
* id\_zaposlenik 🡪 INTEGER, ne smije biti prazan (NOT NULL)
* datum 🡪 DATETIME, datum kada je zaposlenik zabilježio prisutnost, DEFAULT NOW(), ako podatak nije unesen automatski se postavi na sadašnje vrijeme
* broj\_sati 🡪 FLOAT, ako zaposlenik nije radio pun sat i ne smije biti prazan (NOT NULL)
* broj\_sati\_sa\_bonusima 🡪 FLOAT, ako zaposlenik nije radio pun sat, DEFAULT 0, automatski je postavljeno na vrijednost 0



Slika 12 prisutnost (tablica)

# 4. UBACIVANJE PODATAKA U BAZU

Unos podataka u relacije vrši se tako da umetnemo naredbu „INSERT INTO“, zatim slijedi ime tablice i nakon nje se piše „VALUES“. Nakon šta ovo napišemo možemo ubaciti podatke koje god hoćemo. Na slici su prikazani naši podatci. U zagradi upisujemo vrijednosti i sa zarezom ih odvajamo jedan od drugog i idemo po redoslijedu kao što smo u tablicama napisali atribute.



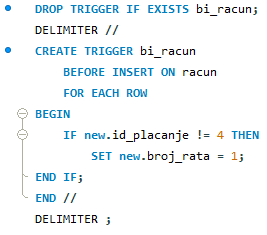
Slika 13 Umetanje vrijednosti u tablicu "zaposlenik"

# 5. FUNKCIJE, POHRANJENE PROCEDURE, OKIDAČI I POGLEDI

## 5.1. OKIDAČ:

Pitanje glasi: Napiši okidač koji će osigurati da u slučaju korisnik ne plaća karticom nije u mogućnosti plaćati na rate, nego mora se platiti u jednom iznosu.

Odgovor:



Slika 14 Okidač 1

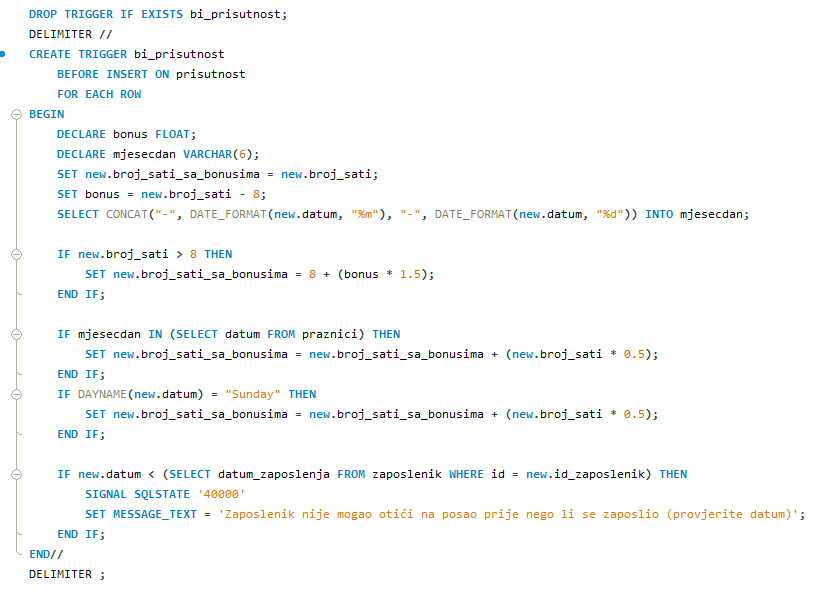
Objašnjenje:

* Prvo smo napisali naredbu drop trigger if exists zbog toga ako već imamo napravljen okidač koji se zove istim imenom da se izbriše, ta linija se obično koristi kada testiramo ako trigger ne radi.
* BEFORE INSERT ON koristimo kada želimo pokrenuti okidač prije nego ubacujemo podatke u neku tablicu koja je u ovome slučaju račun. U okidaču imamo samo jedan uvjet u kojemu je zadan slučaj ako id plaćanja na računu nije jednak broju 4 što je plaćanje karticom na rate, novi broj rata se postavlja na jednu.
* Ukratko, broj rata će uvijek biti postavljen na samo jednu, tj. plaćanje odjednom osim ako kupac ima karticu na rate, onda možemo unesti željeni broj rata (do 60).

## 5.2. OKIDAČ:

Pitanje glasi: Napiši okidač koji nam osigurava da u slučaju, ako je zaposlenik radio preko 8 sati u jednom danu, satnica za prekovremene sate mu se nadodaje na satnicu (+50%) te ako je praznik/nedjelja onda mu se svi sati računaju 50% više.

Odgovor:



Slika 15 Okidač 2

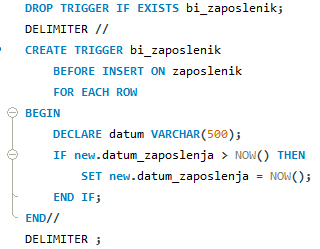
Objašnjenje:

* Prvo smo napravili okidač u kojem u kojem smo definirali četiri uvjeta koja će se izvršiti prije nego ubacimo podatke u tablicu prisutnost te definirali varijable broj sati sa bonusima, bonus i mjesecdan. Uvjet funkcionira tako da se u prethodno definiranu varijablu bonus sprema broj sati koje je zaposlenik radio te se od njih oduzme 8 radnih sati, na taj način smo izvukli broj sati koji se trebaju dodatno nagraditi.
* U broj sati s bonusima spremamo broj sati koji će se isplatiti zaposleniku, na primjer ako je zaposlenik radio 10 sati u jednome danu, isplatiti će mu se normalna cijena satnice, ali za 11 sati odrađenih (8 zadane cijene i dodatna 2 sata sa 50% bonusa (1.5 x 2 = 3)).
* Prvi okidač je uvjet koji se izvršava ako je zaposlenik radio više od 8 sati u danu, što znači da je radio prekovremeno te će dobiti dodatnu isplatu za taj radni dan. Dakle u prvome uvjetu zadajemo da je novi broj sati sa bonusima jednak unesenom broju sati nekog zaposlenika koji podijelimo sa 2 kako bi dobili 50% što su zapravo dodatni sati i na taj broj nadodamo broj sati koji su zaposlenici zapravo odradili.
* Drugi uvjet je jednak prvome osim što se bonus satnica nadodaje na sve sate u tome danu, a uvjet se ispunjava tako da je zaposlenik došao na posao na praznik (jedan od datuma iz tablice praznici). Treći uvjet se temelji na formuli prvoga i drugoga osim što se bonus nadodaje samo ako je zaposlenik bio na poslu u nedjelju i tada mu se bonus nadodaje na cijelu satnicu.
* Četvrti uvjet nam osigurava da zaposlenik nije mogao biti prisutan na poslu prije nego je zaposlen. Datum zaposlenja zaposlenika provjeravamo kroz tablicu zaposlenik.
* U slučaju da imamo takav unos, napravili smo error koji će mysql izbaciti kako bi znali što trebamo izmijeniti i insert se neće izvršiti.

## 5.3. OKIDAČ

Pitanje glasi: Napiši okidač koji će osigurati da datum zaposlenja postane trenutni datum ako pokušamo zaposliti nekoga u budućem vremenu.

Odgovor:



Slika 16 Okidač 3

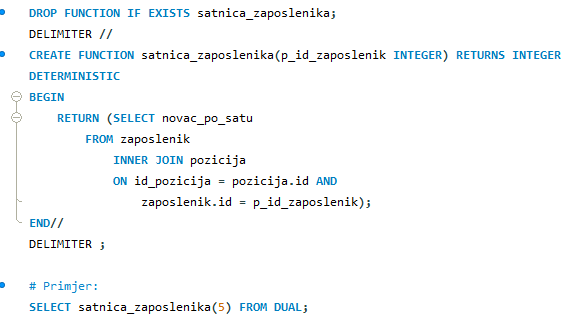
Objašnjenje:

* Napisali smo naredbu drop trigger if exists za slučaj da okidač već postoji i zove istim imenom da se izbriše, tu smo naredbu napisali za slučaj kada testiramo ako trigger ne radi.
* BEFORE INSERT ON se koristi kada želimo da se izvrši okidač prije nego ubacujemo podatke u našu tablicu. U samome okidaču imamo jedan uvjet a to je ako smo unijeli datum zaposlenja nekoga zaposlenika u budućem vremenu, datum zaposlenja će se postaviti na sadašnje vrijeme.

## 5.4. FUNKCIJA

Pitanje glasi: Napiši funkciju koja vraća satnicu određenog zaposlenika.

Odgovor:



Slika 17 Funkcija 1

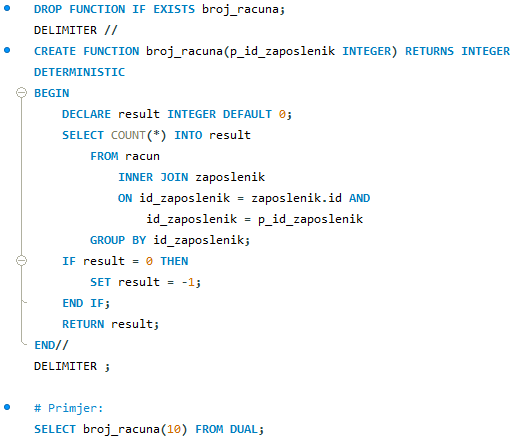
Objašnjenje:

* Prvo provjeravamo ako funkcija s istim imenom već postoji i onda je izbrišemo ako postoji, ovu funkciju smo dodali kako ne bi svaki put morali pronaći i izbrisati funkciju u shemama nego samo pokrenuti isti kod ispočetka.
* Funkcija prima integer kao parametar te ako se taj integer pronalazi kao jedan od id-eva zaposlenika, funkcija će nam vratiti satnicu tog zaposlenika.
* Funkciju pozivamo naredbom select onda ime funkcije i u zagradu upišemo integer koji je zapravo id zaposlenika te na kraj dodajemo FROM DUAL jer select se mora sastojati od from dijela, a dual je jedan specijalni red i stupac koji je prisutan u svakoj tablici.

## 5.5. FUNKCIJA

Pitanje glasi: Napiši funkciju koja vraća broj računa koji je zaposlenik izdao, treba izbaciti -1 ako zaposlenik nije niti jedan račun izdao.

Odgovor:



Slika 18 Funkcija 2

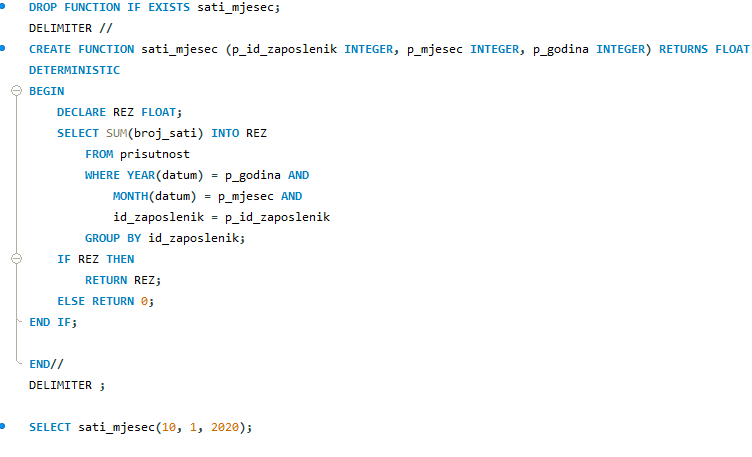
Objašnjenje:

* Provjeravamo ako funkcija s istim imenom već postoji zato da ne moramo svaki put tražiti funkcija u shemama kako bi je izbrisali nego ćemo samo pokrenut isti kod ispočetka i odmah izbrisati i napraviti novu funkciju.
* Funkcija prima kao parametar id zaposlenika i vraća koliko je taj zaposlenik izdao računa na način da smo deklarirali novu varijablu rezultat. Ona sprema broj svakog poklapanja našeg unosa (p\_id\_zaposlenik) sa id-em zaposlenika koji se nalazi u insertovima računa i ako smo unijeli id zaposlenika koji nije izdao niti jedan račun, funkcija će nam vratiti rezultat -1.
* Funkcija se poziva naredbom select zatim ime naše funkcije i u zagrade upišemo broj id-a zaposlenika i na kraj dodajemo FROM DUAL jer select mora imati select i from dio dok je dual neki specijalni red i stupac koji je prisutan u svemu.

## 5.6. FUNKCIJA

Pitanje glasi: Napiši funkciju koja će vratiti zbroj sati rada od određenog zaposlenika u određenom mjesecu

Odgovor:



Slika 19 Funkcija 3

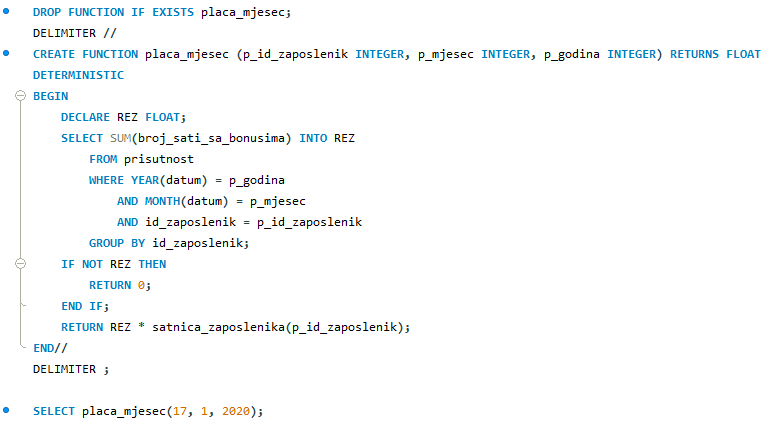
Objašnjenje:

* Funkcija radi na način da prima tri parametra kao ulazna, a to su:
  + id zaposlenika po kojem odabiremo od kojeg zaposlenika želimo vidjeti broj sati,
  + mjesec zaposlenika za koji ćemo zbrojiti sve sate,
  + te godinu u kojoj je zaposlenik radio.
* Zatim smo deklarirali novu varijablu rezultat u koji spremamo sumu sati nekog mjeseca koji se poklapa sa unesenim mjesecom, id-eem zaposlenika i godinom, te if uvjet ako je zaposlenik stvarno radio taj mjesec funkcija nam vraća ukupan broj sati u tome mjesecu.

## 5.7. FUNKCIJA

Pitanje glasi: Napiši funkciju koja vraća izračun plaće određenog zaposlenika u određenom mjesecu

Odgovor:



Slika 20 Funkcija 4

Objašnjenje:

* Funkcija radi na način da prima tri parametra kao ulazna, a to su:
  + id zaposlenika po kojem odabiremo od kojeg zaposlenika želimo izračunati plaću,
  + mjesec zaposlenika za koji ćemo izračunati plaću,
  + te godinu u kojoj je zaposlenik radio
* U funkciji se zbraja broj sati zaposlenika kao i u prošloj funkciji samo što zbrajamo broj sati sa bonusima. Zbog toga što se u plaći računaju bonusi.
* Nakon izračuna broja sati množimo sa satnicom zaposlenika tako da pozovemo funkciju satnica\_zaposlenika.

## 5.8. PROCEDURA

Pitanje glasi: Napiši proceduru koja sprema auto u različite grupe ovisno o cijeni (grupe bitno\_prodati i manje\_bitno)

Odgovor:



Slika 21 Procedura 1

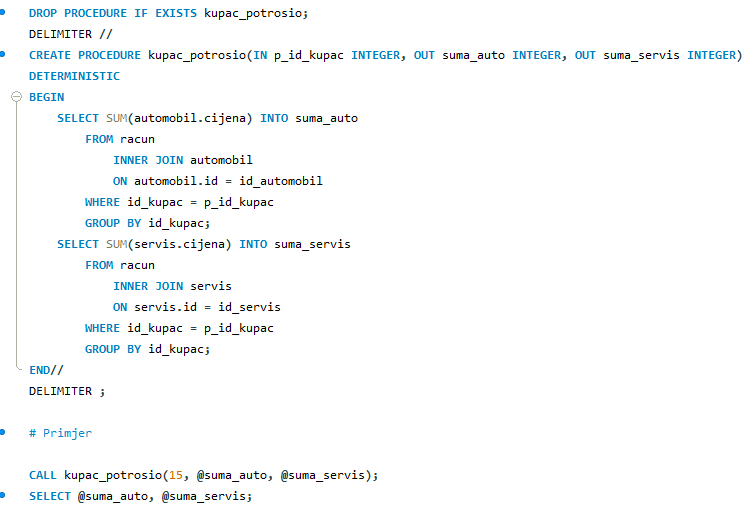
Objašnjenje:

* Procedura prima jedan parametar (limit\_cij) i na izlaz izbacuje dvije varijable (bitno\_prodati i manje\_bitno)
* Dodan je pokazivač koji pokazuje na svaku n-torku tablice automobil i spremamo naziv i cijenu u dvije varijable (auto i dod\_cijena)
* Također se treba koristiti petlja kako bi iterirali kroz tablicu sa pokazivačem i zaustavlja se kada pokazivač pokuša pokazati na n-torku nakon zadnje.
* Na svakoj n-torki provjerava se da li je cijena na koji pokazivač pokazuje manja od našeg ulaznog parametra i ako je, naziv auta dodaje u manje\_bitno, u suprotnom dodaje u bitno\_prodati.

## 5.9. PROCEDURA

Pitanje glasi: Napiši proceduru koja sprema brojeve koliko je sveukupno novaca potrošio kupac na zasebno aute i servis.

Odgovor:



Slika 22 Procedura 2

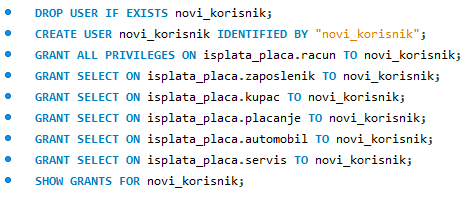
Objašnjenje:

* Procedura prima jedan parametar (p\_id\_kupac) i na izlaz izbacuje dvije varijable (suma\_auto i suma\_servis)
* Napisali smo upit koji nam prikazuje sumu svih cijena automobila koji su prodani i koje je kupio kupac s našim id-em te tu sumu smo spremili u suma\_auto.
* Dodali smo isti upit samo za cijene servisa i tu sumu smo spremili u suma\_servis.

## 5.10. AUTENTIFIKACIJA I AUTORIZACIJA

Pitanje glasi: Napravite novog korisnika koji ima sva prava na tablici račun, dok na tablicama zaposlenik, automobil, servis, korisnik i plaćanje može samo čitati.

Odgovor:



Slika 23 Autentifikacija i autorizacija

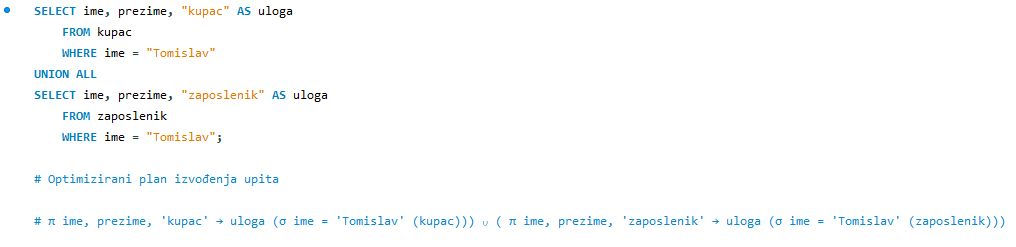
Objašnjenje:

* Kreirali smo korisnika novi\_korisnik i postavili smo mu lozinku novi\_korisnik
* Omogućili smo mu sve privilegije od tablice računa. Dopustili smo mu prikaz ostalih tablica: zaposlenik, kupac, placanje, automobil, servis te smo na kraju provjerili koje mogućnosti ima korisnik

## 5.11. OPTIMIZIRAN PLAN IZVOĐENJA UPITA

Pitanje glasi: Napraviti SQL upit i optimizirani plan izvođenja upita koji će prikazati popis imena i prezimena kupaca i zaposlenika čije je ime "Tomislav" i dodati novi atribut "uloga" u koji sprema riječ kupac ili zaposlenik ovisno koju ulogu ima ta osoba.

Odgovor:



Slika 24 Optimiziran plan izvođenja upita

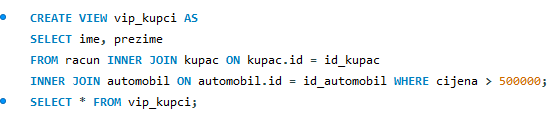
Objašnjenje:

* Iz tablice kupac smo selektirali ime i prezime te smo dodali atribut koji kod svake n-torke ima riječ kupac. Na kraju našeg select-a tražimo ime kupca koji je jednak Tomislavu.
* U drugom select-u smo uzeli tablicu zaposlenik i iz nje imamo opet ime i prezime te smo dodali atribut koji kod svake n-torke ima riječ zaposlenik. Također, pri kraju tražimo zaposlenika s imenom Tomislav.
* Zatim smo ujedinili ta dva upita skupa.
* Na kraju smo taj upit optimizirali.

## 5.12. POGLED

Pitanje glasi: Napiši pogled koji pokazuje imena i prezimena kupaca koji su kupili aute skuplje od 500000 kuna.

Odgovor:



Slika 25 Pogled 1

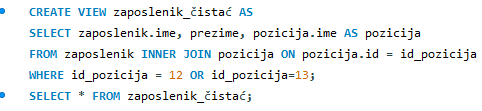
Objašnjenje:

* Kreirali smo pogled s nazivom vip\_kupci. Spojili smo tablice racun, kupac i automobil sa INNER JOIN-om i postavili uvjet da cijena bude veća od pola milijuna.
* Zatim smo spremili ime i prezime kupaca iz te spojene tablice i provjerili da li funkcionira pogled.

## 5.13. POGLED

Pitanje glasi: Napiši pogled koji prikazuje imena i prezimena zaposlenika koji rade kao čistaći.

Odgovor:



Slika 26 Pogled 2

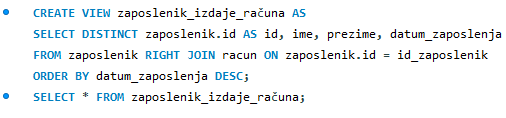
Objašnjenje:

* Napravili smo pogled s nazivom zaposlenik\_cistac i spojili tablice zaposlenik i pozicija sa INNER JOIN-om i postavili uvjet da id\_pozicija mora biti 12 ili 13 tj. da zaposlenik bude čistać.
* Zatim smo spremili ime i prezime zaposlenika zajedno sa imenon pozicije i provjerili da li funkcionira pogled.

## 5.14. POGLED

Pitanje glasi: Napiši pogled koji prikazuje sve zaposlenike koji su izdali račun te ih sortira po datumu zaposlenja toga korisnika.

Odgovor:



Slika 27 Pogled 3

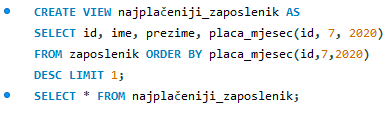
Objašnjenje:

* Napravili smo pogled zaposlenik\_izdaje\_racuna i spojili smo tablice zaposlenik i racun sa RIGHT JOIN-om tako da prikaže sve račune.
* Poredali smo rezultate po datumu zaposlenja od najnovijeg do najstarijeg zaposlenika. Zatim smo spremili id, ime, prezime i datum zaposlenja zaposlenika.

## 5.15. POGLED

Pitanje glasi: Napiši pogled koji prikazuje zaposlenika koji je najviše zaradio u 7 mjesecu 2020.

Odgovor:



Slika 28 Pogled 4

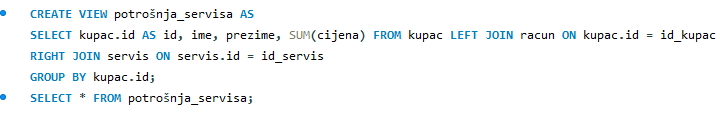
Objašnjenje:

* Napravili smo pogled najplaceniji\_zaposlenik i iz tablice zaposlenik smo poredali rezultate po placi tog zaposlenika u 7. mjesecu 2020. godine od najveće do najmanje plaće.
* Zatim smo spremili id, ime, prezime i samu plaću tog zaposlenika.

## 5.16. POGLED

Pitanje glasi: Napiši pogled koji prikazuje sve kupce i njihovu svotu novcu koju su potrošili na servise auta.

Odgovor:



Slika 29 Pogled 5

Objašnjenje:

* Napravili smo pogled koji se zove potrošnja\_servisa. U njemu smo prikazali id iz tablice kupac koji smo nazvali id i prikazali smo ime i prezime te sumu cijena iz table kupac koju smo left join-ali sa tablicom račun preko id iz tablice kupac i id\_kupac iz tablice račun.
* Rezultat tog left join-a smo right join-ali sa tablicom servis preko id iz tablice servis sa id\_servis iz prijašnjeg join-a te smo ih na kraju grupirali po id-u iz tablice kupac.

# 6. PROGRAMSKO RJEŠENJE

Za izradu naše aplikacije odlučili smo koristiti Node.js i koristili smo nekoliko paketa koji su nam bili potrebni za povezivanje s bazom podataka (mysql, nodemon, express…).

Aplikaciju otvaramo pomoću localhost:5050 veze te kada ju otvorimo imamo grafičko sučelje u kojoj možemo izabrati između prikaza funkcija, procedura ili cijelih tablica. U aplikaciji možemo vidjeti funkcije *satnice\_zaposlenika, broj\_računa, broj\_sati\_u\_mjesec i izračun\_plaće, a* procedure su *sortiranje\_auta i troškovi\_kupca.* Nakon što odaberemo željenu funkciju ili proceduru, gumb “Pokreni funkciju/procedure” nas vodi na novu stranicu gdje unosimo atribute koji su nam potrebni za rad funkcije ili procedure. Kada na primjer upišemo id zaposlenika u funkciji satnica zaposlenika te stisnemo pokreni dobivamo brojčanu vrijednost satnice za tog zaposlenika. Ukoliko želimo prikazati cijelu tablicu onda pomoću izbornika odaberemo određenu tablicu te pritiskom na gumb “Pokreni” dobivamo tu tablicu.

Sve podatke koje unosimo ili mijenjamo u bazi podataka automatski se ažuriraju i u aplikaciji. Aplikaciju pokrećemo komandom „npm start“. Aplikacija se spaja s korisnikom koji ima user: root i password: root.

# 7. ZAKLJUČAK

Cilj ovog projekta je bio što detaljnije opisati primjenu baze podataka u nekom poslovanju, a u našem slučaju sustav za isplatu plaća u auto salonu. Komunikacija između tima je bila vrlo efektivna. Svaki član tima je odradio svoj dio posla kao šta mu je bilo zadano. Ovaj projekt nam je definitivno bio teži nego prethodni, ali smo se na kraju snašli i uspjeli napraviti jednu solidnu bazu podataka i aplikacije. Neke ideje su nam bile korisne poput dodavanju zaposleniku više sati ako je radio prekovremene ili na praznicima, a neke ideje smo odlučili odbaciti. Za aplikaciju smo imali malih poteškoća i mijenjali smo alate za izradu web aplikacije dok se nismo na kraju odlučili za Node.js.

Nakon izrade ovog projekta proširili smo znanje i iskustvo s ovim predmetom. Zadovoljni smo sa krajnjim rezultatom.

# 8. LITERATURA

* Predavanja od mentora: doc. dr. sc. Goran Oreški
* Vježbe od „Baze podataka 1“ i „Baze podataka 2“, asistent: Romeo Šajina
* Slike su sve iz naše baze podataka i ER dijagram je izrađen u Lucidchart-u na temelju tablica iz baze podataka
* Aplikacija – razni Youtube videi o HTML, PHP, JS, Node.js, Django
* [DSC] poglavlje 7. "Database Design and the E-R Model"
* [DSC] poglavlje 8. "Relational Database Design"