**SVEUČILIŠTE U SPLITU**

**PRIRODOSLOVNO-MATEMATIČKI FAKULTET**

**ODJEL ZA INFORMATIKU**

SEMINARSKI RAD

BAZA-PODATAKA-BANKA

STUDENT:

Tomislav Mihaljević

PROFESOR: Monika Mladenović

Split, travanj 2017.

SADRŽAJ:

### UVOD

Opis: Potrebno je osmisliti i realizirati bazu podataka koja može učinkovito poslužiti za potrebe banke. U fazi analize, utvrđeno je da banka nudi mogućnosti štednje u domaćoj i stranoj valuti te korištenje tekućeg računa isključivo u domaćoj valuti. Banka ima potrebne podatke o svojim korisnicima. Svaka štedna knjižica kao i tekući račun ima svoje osnovne podatke (broj računa, vlasnik, valuta, datum otvaranja, kamatna stopa, i sl.) te podatke o prometu (uplate i isplate). U slučaju tekućeg računa postoji više vrsti prometa (uplata redovnog primanja, ostale uplate, isplata) Dozvoljeno prekoračenje stanja tekućeg računa je prosječni iznos uplata redovnog primanja. Na iznos nedozvoljenog prekoračenja plaća se kamata prema stopi koju banka propisuje za svaki mjesec. Osnovni zahtjevi korisnika su slijedeći:

* omogućiti evidenciju stanja i prometa (uplate i isplate, te pripis kamate) na štednim knjižicama
* omogućiti evidenciju stanja i prometa tekućih računa (uplate i isplate, te pripis kamate)
* dobivanje informacije o dozvoljenom minusu na tekućem računu Baza podataka treba biti u što većoj mjeri normalizirana.

### IZRADA MODELA

Prije same izrade baze podataka potrebno je osmisliti model podataka prema kojem će se baza podataka napraviti. Za prikazivanje modela se koriste ER (model-veza) dijagrami. Metoda Entitet-veza je temelj za formiranje modela.

Osnovni koncepti ER modela:

* Entiteti
* Veze među entitetima
* Atributi entiteta

#### Entitet

Entitet je bilo što o čemu želimo prikupljati i pohranjivati informacije (npr. *kuća*, *student*, *auto*, itd.). S prikupljanjem i pohranjivanjem informacija o pojedinom entitetu možemo započeti, ako je poznat identifikator tog entiteta. U modelu „Banka“ smo koristili entitete:

* BANKA - glavni entitet koji sadrži podatke o nazivu banke i kontakt.
* PODRUŽNICA - entitet koji nam govori gdje se nalaze i kako sa njima stupiti u kontakt.
* RAČUN - entitet koji sadrži podatke o datumu otvaranja računa te primarni ključ za račun .
* KLIJENT - entitet koji sadrži potrebne podatke o klijentu banke(ime, prezime, adresu….).
* ZAPOSLENIK - entitet koji sadrži podatke o zaposleniku banke(ime, prezime ,adresu ,odjel….).
* ADRESA – kako se adresa ponavlja više puta napravili smo poseban entitet ADRESA za adrese podružnica, zaposlenika i klijenta.
* TRANSAKCIJA - entitet koji sadrži podatke o transakciji
* TIP\_TRANSAKCIJE - entitet u kojem se nalaza tipovi transakcije(depozit, transfer)
* ZIRO\_RAČUN - entitet koji sadrži podatke o tipu računa tj. žiro račun.
* TEKUCI\_RAČUN- entitet koji sadrži podatke o tipu računa tj. tekući račun.

ŠTEDNA\_KNJIŽICA – sadrži podatke o štednji(vlasnik ,valuta ,iznos ,kamatna stopa…).

Svaki entitet ima svoj primarni ključ(*primary key).* Primarni ključ je polje ili skup polja s vrijednostima koje su jedinstvene cijeloj tablici. Nemoguće je da se u dva različita retka pojavi isti primarni ključ.

Strani ključ (*foreign key*) je atribut koji služi kao primarni ključ drugog entiteta. Strani ključ nam fizički osigurava vezu između tablica, a ujedno je i primarni ključ u nekoj drugoj tablici.

#### Atributi

Atributi fizički predstavljaju stupce u tablicama u bazi podataka.



*Primjer atributa za entitet „banka“*

### RELACIJE

Relacije, odnosno veze, su odnosi između dvije tablice. Entiteti su međusobno povezani

relacijama. Postoje tri vrste relacija:

* One-to-one (jedan na jedan) 1:1
* One-to-many (jedan na više) 1:n
* Many-to-many (više na više) m:n

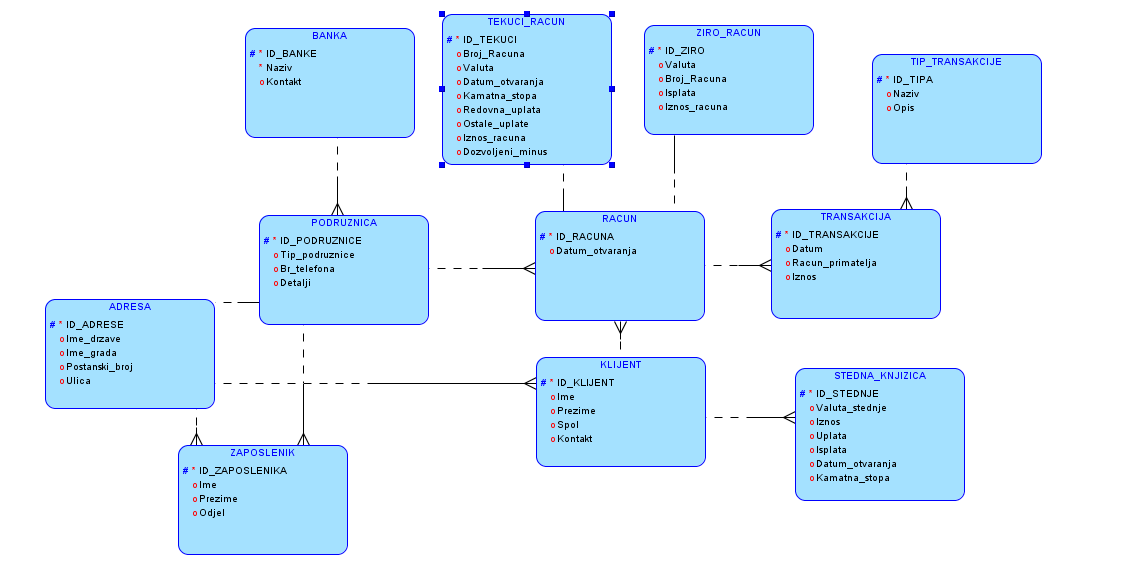
U ovom modelu smo koristili relacije jedan na jedan-1:1 i jedan na više-1:n.

#### Relacija jedan na jedan

* *RACUN-TEKUCI\_RACUN* imamo dva entiteta-račun korisnika i tip računa. Veza između njih je jedan naprema jedan jer klijent banke koji ima jedan račun u banci može imati samo jedan tekući račun, te dani tekući račun može imati samo jedan vlasnik računa u banci.
* *RACUN-TEKUCI\_RACUN* vrijedi isto kao i za prethodni primjer.
* *PODRUŽNICA-ADRESA* jedna podružnica mora imati jednu adresu te se na jednoj adresi može nalaziti samo jedna podružnica.

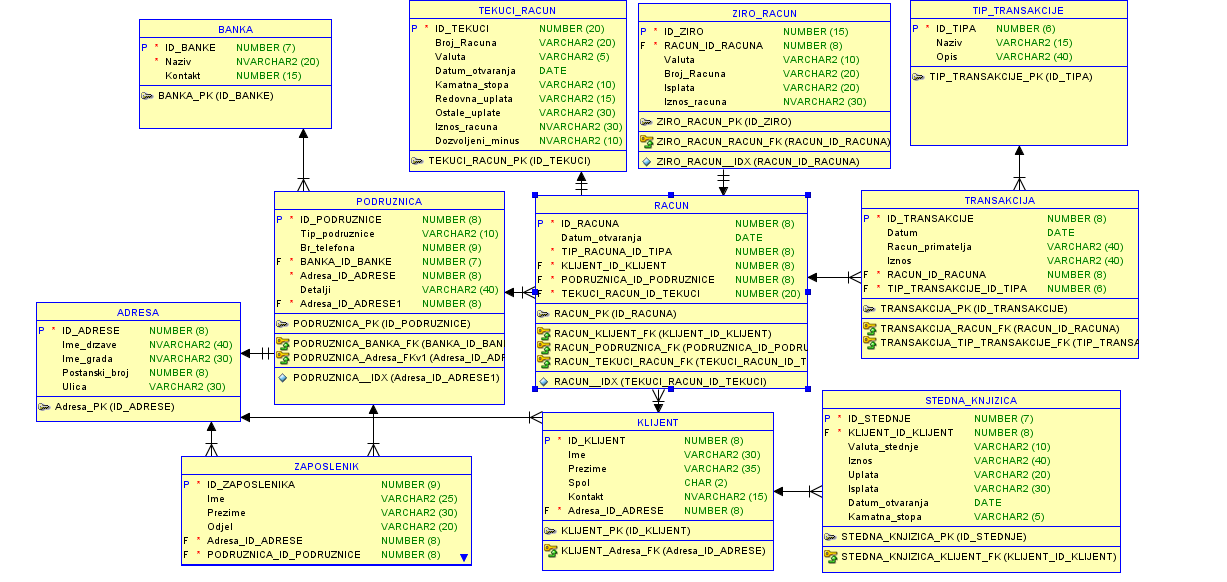
#### Relacija jedan na više

* *PODRUZNICA-BANKA* jedna banka može imati više podružnica dok jedna podružnica može imati samo jodnog vlasnika-banku.
* *RAČUN-PODRUŽNICA* jedan račun može biti otvoren u jednoj podružnici dok jedna podružnica može otvarati više računa.
* *RAČUN-KLIJENT* jedan račun može imati samo jednog vlasnika-klijenta dok jedan klijent može imati više računa .
* *KLIJENT-ADRESA* jedan klijent mora imati jednu adresu na koju je prijavljen , dok se na jednoj adresi može nalaziti više klijenata(npr. ako su na 2 člana obitelji članovi iste banke).
* *ZAPOSLENIK-ADRESA* isto kao u prethodnom primjeru vrijedi i za zaposlenike banke.
* *ZAPOSLENIK-PODRUŽNICA* jedan zaposlenik može raditi u jednoj podružnici dok jedna podružnica može imati više zaposlenih.
* *RAČUN-TRANSAKCIJA* sa jednog računa možemo obaviti više transakcija dok jednu transakciju obavljamo sa jednog računa(npr. depozit)
* *KLIJENT-STEDNA\_KNJIZICA* jedan klijent može imati više otvorenih štednih knjižica dok štedna knjižica mora imati jednog vlasnika.



*LOGICKI MODEL*

Nakon što smo u datamodeleru napravili logički model generiramo relacijski model prikazan na slici ispod

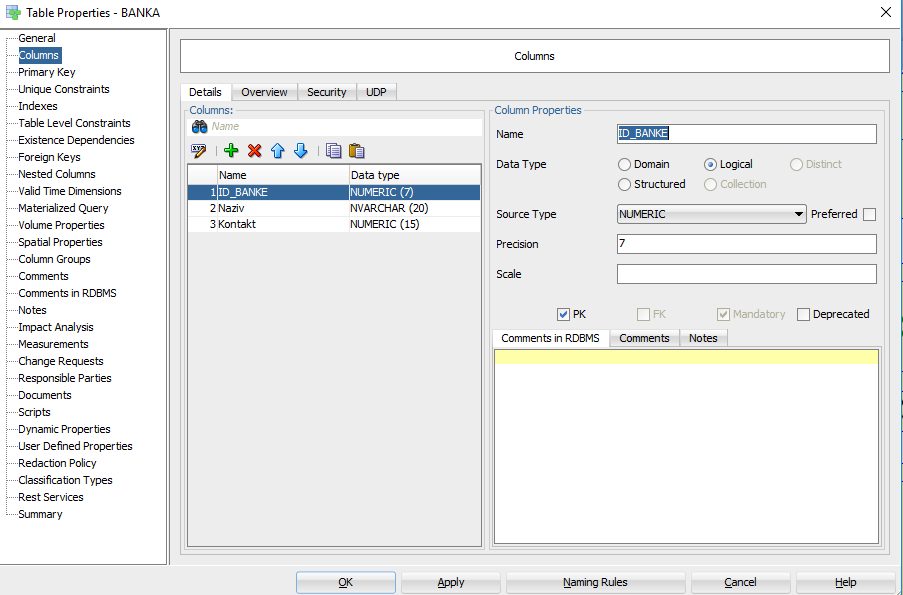


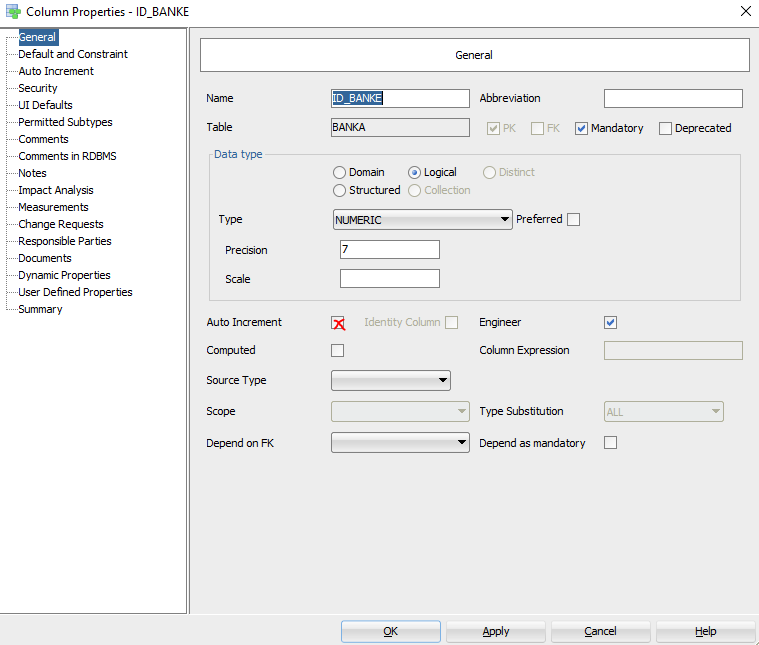
*RELACIJSKI MODEL*

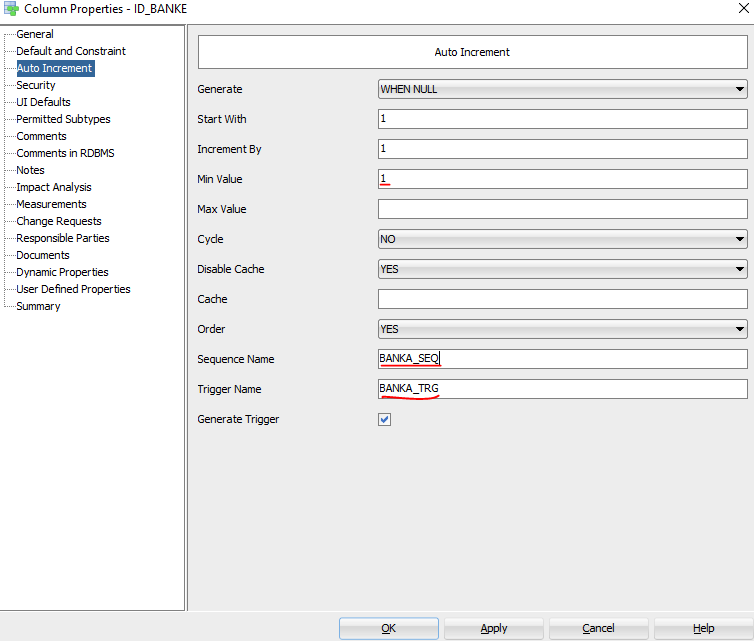
### IZRADA KORISNIKA I VEZE

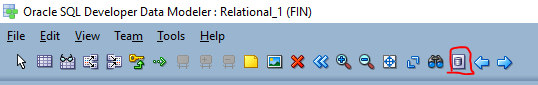
Za primarne ključeve na Oracle bazi ne postoji ugrađeni tip podatka .Potrebno je napraviti sekvencu (sequence) i okidač (trigger). Sekvenca služi za generiranje liste jedinstvenih brojeva za npr. primarne ključeve. Triggeri se pokreću automatski na određeni događaj, npr. prije, na ili nakon unosa, ažuriranja, brisanja.

1. Korak- kliknemo dva puta na primarni ključ ID\_BANKE



1. KORAK-nakon prvog koraka pojavio nam se prozor kao na slici ispod. Na njemu trebamo odabrati *Auto increment*
2. KORAK sa lijeve strane odabrati auto-increment i upisati vrijednosti koje su podvučene crvenom bojom na slici ispod. To je potrebno napraviti za sve ostale primarne ključeve u modelu.

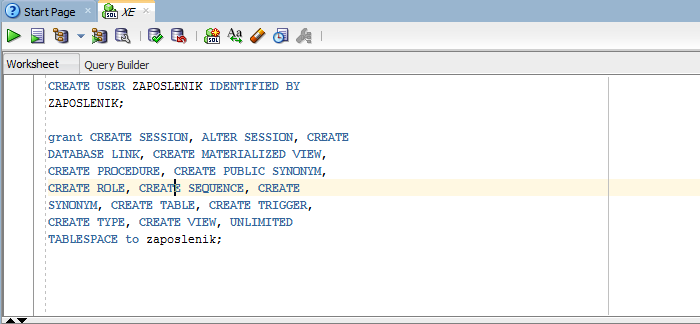


Kada smo uredili relacijski model napravimo DDL .Na početku na izbornoj traci odaberemo ikonu koja je na slici ispod označena crvenom bojom. 

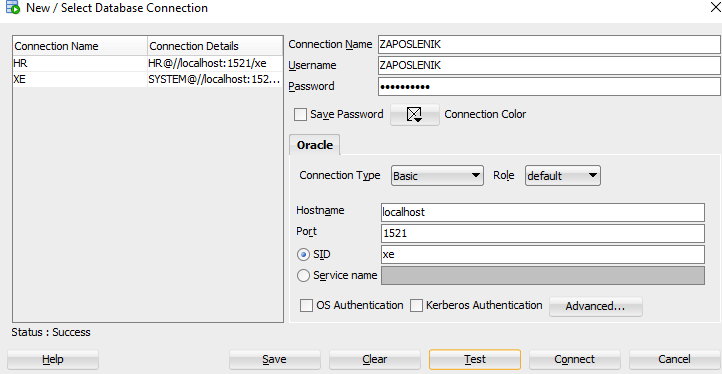
Zatim na prozoru koji na se pojavi odaberemo botun Generate te poslije njega odaberimo OK.

Nakon toga dobijemo skriptu koju kopiramo.

Prije izrade tablice moramo izraditi korisnika.Na slici ispod je prikazano kako smo kreirali korisnika ZAPOSLENIK i dodijelili mu prava.

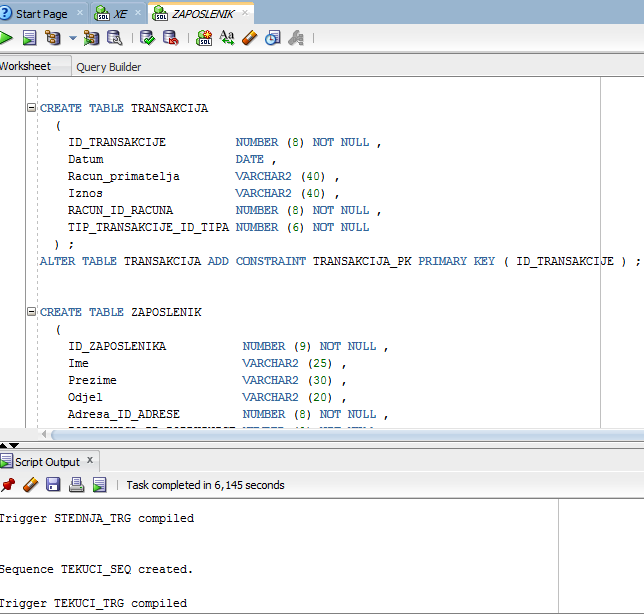


Nakon što smo napravili korisnika trebamo napraviti vezu.



### IZRADA OBJEKATA IZ DDL-a

Tek nakon što smo napravili vezu možemo kopirati DDL skriptu i izvršiti je u sql devloperu na vezi zaposlenik.



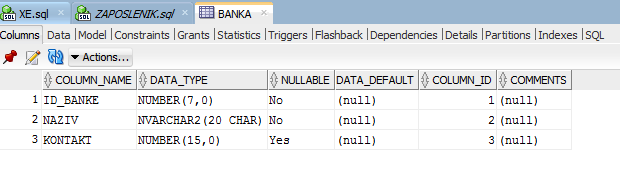
### UPITI

Upiti su komponente baze podataka koji služe za unos podataka u bazu podataka,njihovo pretraživanje, brisanje , sortiranje itd.

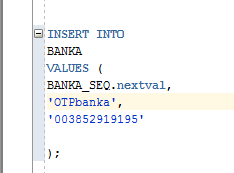
#### INSERT

Upiti koji služi za unos podataka u bazu.Navest ćemo primjer unosa podataka u tablicu BANKA.

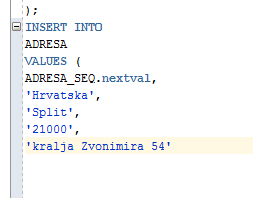
Na slici ispod možemo vidjeti koje atributi se nalaze u tablici banka i koja su njihova ograničenja pri unosu.



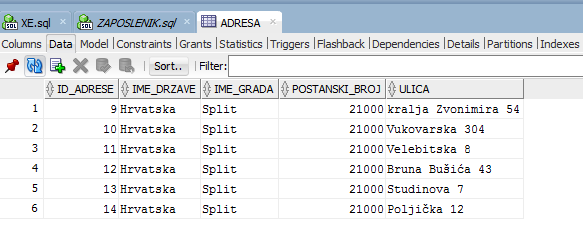
Za unos podataka u tablicu „BANKA“ koristit ćemo naredbu INSERT.Atribut „Naziv“ prima vrijednost koja je ograničena sa 20 znakova.NVARCHAR2 je unicode znak varijabilne veličine. Kontakt je atribut koji prima vrijednost broja ograničenu do 15 znamenki.



Nakon što smo unijeli podatke banke unosimo podatke za tablicu ADRESE, koje imaju klijenti ,podružnice ,zaposlenici.

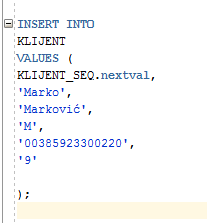


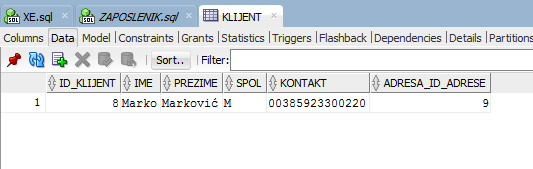
Nakon više zadanih naredbi unijeti podatci izgledaju kao na slici ispod. Možemo ih vidjeti tako da kliknemo na tablicu ADRESA i potom kliknemo na data.



Ukoliko želimo dodati vrijednost atributu koji je strani ključ u tablici pridodajemo mu njegovu vrijednost tj. vrijednost primarnog ključa. Na taj način povezujemo podatke između dvije tablice.

Primjer povezivanja je prikazan ispod.

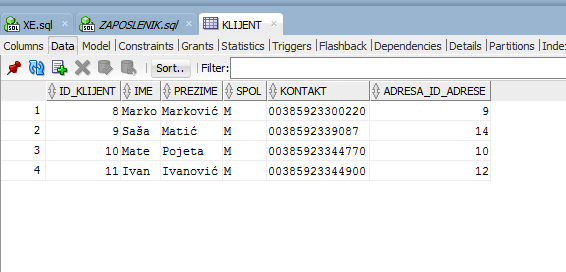




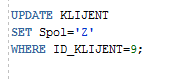
Možemo zaključiti da Marko živi u Splitu u ulici kralja Zvonimira 54.

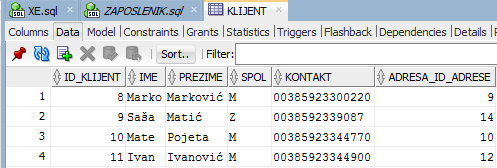
#### UPDATE

Za ažuriranje podataka koristi se UPDATE. Primjer prije naredbe update:



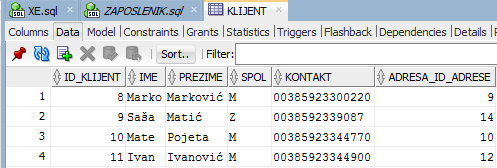
Naredba update mijenja spol u 'Z' klijentu kojem je ID\_KLIJENT=9.



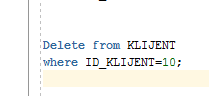
Rezultati naredbe: 

#### DELETE

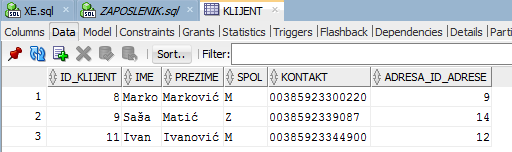
Naredbu delete koristimo za brisanje podataka. Prije brisanja imamo:



Naredba za brisanje:

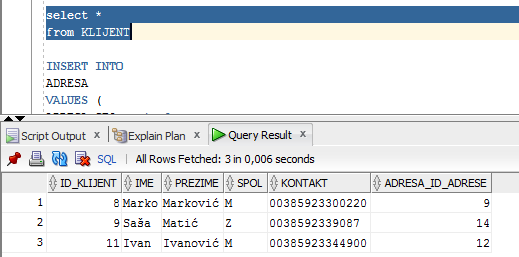


Nakon naredbe za brisanje:

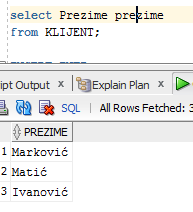


#### SELECT

Naredbu select koristimo kada želimo prikazati određene podatke iz tablice.Kada želimo prikazati sve stupce iz neke tablice/a, koristimo znak \*.Primjer:



Kada želimo prikazati nazive stupaca drugim nazivima koristimo Alijase. Na sljedećem primjeru ćemo iz tablice izvući samo prezime klijenata.



Na primjeru ispod ćemo prikazati kako prikazati broj i ime klijenata.

