*Elektrotehnički fakultet Sarajevo*

*PROJEKAT IZ PREDEMTA*

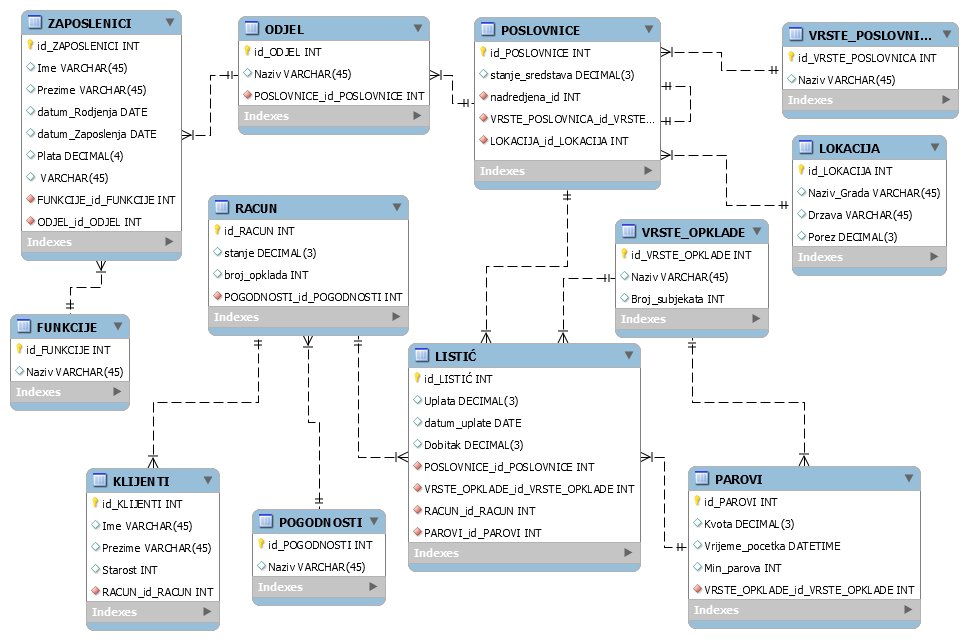
*OSNOVE BAZA PODATAKA*

*SANIDA ĐUMIŠIĆ*

*Index: 17849*

*Grupa: RI 2*

*2017/2018.*

**

*Kreiranje tabela*

---tabela zaposlenika vrsi evidenciju svih radnika koji rade u kladionicama

CREATE TABLE zaposleni (zaposlenik\_id NUMBER NOT NULL,  
 Ime VARCHAR2(45),  
 Prezime VARCHAR2(45),  
 datum\_Rodjenja DATE NOT NULL,  
 datum\_Zaposlenja DATE NOT NUL,  
 plata NUMBER,  
 funkcija\_id NUMBER NOT NULL,  
 odjel\_id NUMBER NOT NULL,   
CONSTRAINT zaposleni\_id\_pk PRIMARY KEY(id),  
CONSTRAINT z\_funkcija\_id\_fk FOREIGN KEY(funkcija\_id) REFERENCES funkcije(funkcija\_id),   
CONSTRAINT z\_odjel\_id\_fk FOREIGN KEY(odjel\_id) REFERENCES odjel(odjel\_id));

---tabela odjeli vrsi evidenciju svih odjela koji koji postoje u odredjenim kladionicama

CREATE TABLE odjel (odjel\_id NUMBER NOT NULL,  
 naziv VARCHAR2(45),  
 poslovnica\_id NUMBER NOT NULL,  
CONSTRAINT odjel\_id\_pk PRIMARY KEY(id)  
CONSTRAINT z\_poslovnica\_id\_fk FOREIGN KEY(poslovnica\_id) REFERENCES poslovnice(poslovnica\_id));

---tabela funkcije daje spisak mogucih funkcija zaposlenika

CREATE TABLE funkcije (funkcija\_id NUMBER NOT NULL,  
 Naziv VARCHAR2(45),  
CONSTRAINT funkcija\_id\_pk PRIMARY KEY(funkcija\_id));

---tabela vrste\_poslovnica prikazuje koje vrste kladionica postoje

CREATE TABLE vrsta\_poslovnice (vrsta\_id NUMBER NOT NULL,  
 Naziv VARCHAR2(45),   
CONSTRAINT vrsta\_poslovnice\_id\_pk PRIMARY KEY(vrsta\_id),   
CONSTRAINT kartica\_naziv\_ck CHECK (Naziv IN ('Centralna','Regionalna', 'Obicna', 'Glavna'));

---tabela lokacija daje naziv svih lokacije gdje su otvorene poslovnice

CREATE TABLE lokacija(lokacija\_id NUMBER NOT NULL,  
 Naziv\_grada VARCHAR2(45),   
 drzava VARCHAR2(45),   
 porez NUMBER,  
CONSTRAINT lokacija\_id\_pk PRIMARY KEY(lokacija\_id));

---tabela poslovnice vrsi evidenciju jedne kladionice

CREATE TABLE poslovnice (poslovnica\_id NUMBER NOT NULL,  
 stanje\_sredstava NUMBER,  
 nadredjena\_id NUMBER,  
 vrsta\_id NUMBER NOT NULL,  
 lokacija\_id NUMBER NOT NULL,  
CONSTRAINT poslovnice\_id\_pk PRIMARY KEY(poslovnica\_id),   
CONSTRAINT p\_vrsta\_poslovnice\_id\_fk FOREIGN KEY(vrsta\_id) REFERENCES vrsta\_poslovnice(vrsta\_id),  
CONSTRAINT p\_lokacija\_id\_fk FOREIGN KEY(lokacija\_id) REFERENCES lokacija(lokacija\_id));

---tabela racun posjeduje razne informacije o stanju klijenta

CREATE TABLE racun(racun\_id NUMBER NOT NULL,  
 stanje NUMBER,   
 broj\_opklada NUMBER,   
 pogodnosti\_id NUMBER NOT NULL,  
CONSTRAINT racun\_id\_pk PRIMARY KEY(racun\_id)   
CONSTRAINT p\_pogodnosti\_id\_fk FOREIGN KEY(pogodnosti\_id) REFERENCES pogodnosti(pogodnosti\_id));

---tabela klijenti cuva informacije o klijentu kladionice

CREATE TABLE klijenti (klijent\_id NUMBER NOT NULL,  
 Ime VARCHAR2(45),  
 Prezime VARCHAR2(45),   
 racun\_id NUMBER NOT NULL,   
 starost NUMBER NOT NULL,  
CONSTRAINT klijent\_id\_pk PRIMARY KEY(klijent\_id),   
CONSTRAINT k\_racun\_id\_fk FOREIGN KEY(racun\_id) REFERENCES racun(racun\_id), CONSTRAINT k\_starost\_ck CHECK (starost>= 18));

---tabela vrste\_opklade cuva informacije o tipovima opklada

CREATE TABLE vrsta\_opklade (vrsta\_opklade\_id NUMBER NOT NULL,  
 Naziv VARCHAR2(15),   
 broj\_subjekata VARCHAR2(15),  
CONSTRAINT vrsta\_opklade\_id\_pk PRIMARY KEY(vrsta\_opklade\_id));

---tabela listica vrsi evidenciju svim opkladama

CREATE TABLE listici (listic\_id NUMBER NOT NULL,  
 vrsta\_opklade\_id NUMBER NOT NULL,   
 poslovnica\_id NUMBER NOT NULL,   
 uplata NUMBER NOT NULL,   
 racun\_id NUMBER NOT NULL,   
 datum\_uplate DATE,   
 dobitak NUMBER,   
 parovi\_id NUMBER,  
CONSTRAINT listici\_id\_pk PRIMARY KEY(listic\_id),   
CONSTRAINT l\_poslovnica\_id\_fk FOREIGN KEY(poslovnica\_id) REFERENCES poslovnice (poslovnica\_id),   
CONSTRAINT l\_racun\_id\_fk FOREIGN KEY(racun\_id) REFERENCES racun (racun\_id),   
CONSTRAINT l\_parovi\_id\_fk FOREIGN KEY(parovi\_id) REFERENCES parovi (parovi\_id),  
CONSTRAINT l\_vrsta\_opklade\_id\_fk FOREIGN KEY(vrsta\_opklade\_id) REFERENCES vrsta\_opklade (vrsta\_id));

---tabela pogodnosti cuva informacije o tipovima mogucih pogodnosti

CREATE TABLE pogodnosti (pogodnost\_id NUMBER NOT NULL,  
 naziv VARCHAR2(15),   
CONSTRAINT pogodnostii\_id\_pk PRIMARY KEY(pogodnost\_id));

---tabela za sve moguce na raspolanju parove za kladjenje

CREATE TABLE parovi (parovi\_id NUMBER NOT NULL,  
 kvota NUMBER NOT NULL,   
 vrsta\_opklade\_id NUMBER NOT NULL,   
 vrijemePocetka DATE,   
 minParova NUMBER NOT NULL,   
CONSTRAINTparovi\_id\_pk PRIMARY KEY(parovi\_id),   
CONSTRAINT pp\_vrsta\_opklade\_id\_fk FOREIGN KEY(vrsta\_opklade\_id) REFERENCES vrste\_opklade (vrste\_opklade\_id));

*Kreiranje sekvenci*

CREATE SEQUENCE zaposleni\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE odjel\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE funkcija\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE vrsta\_id\_seq  
INCREMENT BY 1  
START WITH 0  
MAXVALUE 3  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE poslovnica\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE lokacija\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE racun\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE klijent\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE listic\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE vrsta\_opklade\_id\_seq  
INCREMENT BY 1  
START WITH 0  
MAXVALUE 4  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE pogodnosti\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

CREATE SEQUENCE parovi\_id\_seq  
INCREMENT BY 1  
START WITH 0  
NOMAXVALUE  
MINVALUE 0  
NOCYCLE

***Unos podataka***

**NSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Mujo' , 'Mujic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('11.01.2018', dd.mm.yyyy'), 1200, 2, 10);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Kemo' , 'Kemic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('15.04.2012', dd.mm.yyyy'), 1800, 1, 10);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Hamo' , 'Hamic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('11.02.2017', dd.mm.yyyy'), 1400, 3, 7);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Mara' , 'Maric', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('19.12.2005', dd.mm.yyyy'), 1200, 4, 6);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Sara' , 'Saric', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('20.11.2009', dd.mm.yyyy'), 800, 5, 5);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Daro' , 'Daric', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('22.05.2003', dd.mm.yyyy'), 600, 6, 4);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Fikro' , 'Fiktic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('01.03.2008', dd.mm.yyyy'), 1300, 7, 4);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Ibro' , 'Ibric', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('06.07.2014', dd.mm.yyyy'), 600, 9, 3);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Una' , 'Unic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('19.10.2015', dd.mm.yyyy'), 1200, 10, 2);  
**INSERT INTO** zaposleni **VALUES** (zaposleni\_id\_seq.**NEXTVAL**, 'Fata' , 'Fatic', TO\_DATE('17.12.1987', dd.mm.yyyy'), TO\_DATE('17.02.2015', dd.mm.yyyy'), 1000, 2, 1);

**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Prijem' , 1);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Prijem' , 2);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Postavka', 1);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Finansije' , 2);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Finansije', 3);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Podrska' , 2);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Podrska', 3);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Savjetovanje' , 3);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Isplata', 3);  
**INSERT INTO** odjel **VALUES** (odjel\_id\_seq.**NEXTVAL**, 'Isplata' , 1);

**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Direktor');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Cistac/ica');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Konsultant');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Salterusa');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Zamjenik direktora');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Sekretarica');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Racunovodja');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Blagajnik');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Cuvar');  
**INSERT INTO** funkcija **VALUES** (funckija\_id\_seq.**NEXTVAL**, 'Menadzer');

**INSERT INTO** vrsta\_poslovnice **VALUES** (vrsta\_poslovnice \_id\_seq.**NEXTVAL**, 'Obicna');  
**INSERT INTO** vrsta\_poslovnice **VALUES** (vrsta\_poslovnice \_id\_seq.**NEXTVAL**, 'Regionalna ');  
**INSERT INTO** vrsta\_poslovnice **VALUES** (vrsta\_poslovnice \_id\_seq.**NEXTVAL**, 'Glavna');   
**INSERT INTO** vrsta\_poslovnice **VALUES** (vrsta\_poslovnice \_id\_seq.**NEXTVAL**, 'Centralna');

**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1200000, 3, 1, 9);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, 9, 2, 5);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1100000, 4, 2, 9);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, 5, 2, 6);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, 3, 1, 9);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 900000, 9, 2, 9);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, 5, 2, 7);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 800000, 3, 1, 9);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, NULL, 3, 3);  
**INSERT INTO** poslovnica **VALUES** (poslovnica\_id\_seq.**NEXTVAL**, 1000000, 4, 2, 9);

**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Sarajevo','BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Sarajevo', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Beograd','Srbija',0.20);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Tuzla', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Sarajevo', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Sarajevo', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Zenica', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Kakanj', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Sarajevo', BiH',0.17);  
**INSERT INTO** lokacija **VALUES** (lokacija\_id\_seq.**NEXTVAL**, 'Budva', 'Crna Gora',0.21);

**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 1000, 2, 1);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 1200, 2, 1);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 1400, 10,2);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 40, 14,2);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 400, 18,2);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 22, 20,3);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 87, 22,3);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 300, 25,3);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 250, 100,4);  
**INSERT INTO** racun **VALUES** (racun\_id\_seq.**NEXTVAL**, 500, 114,4);

**INSERT INTO** klijenti **VALUES** (klijent\_id\_seq.**NEXTVAL**, 'Hana' , 'Hanic', 22, 3);  
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Maja' , 'Majic', 27, 1); **INSERT INTO** klijenti **VALUES** (klijent\_id\_seq.**NEXTVAL**, 'Kana' , 'Kanic', 52, 2);  
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Eso' , 'Esic', 19, 7);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Keso' , 'Kesic', 21, 4);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Damir' , 'Damiric', 45, 6);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Veljo' , 'Veljic', 27, 5);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Zaro' , 'Zaric', 30, 9);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Hato' , 'Hatic', 37, 8);   
**INSERT INTO** klijenti **VALUES** (klijenti\_id\_seq.**NEXTVAL**, 'Bino' , 'Binic', 35, 10);

**INSERT INTO** pogodnosti **VALUES** (pogodnosti \_id\_seq.**NEXTVAL**, 'Bez pogodnosti');  
**INSERT INTO** pogodnosti **VALUES** (pogodnosti\_id\_seq.**NEXTVAL**, 'Smanjenje poreza');  
**INSERT INTO** pogodnosti **VALUES** (pogodnosti \_id\_seq.**NEXTVAL**, 'Povecana kamatna stopa');  
**INSERT INTO** pogodnosti **VALUES** (pogodnosti \_id\_seq.**NEXTVAL**, 'Ukidanje poreza');

**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklade\_id\_seq.**NEXTVAL**, 'Kosarka',1);  
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Fudbal', 1);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Rukomet', 1) ;  
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Trka konja', 2);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Trka zeceva', 2);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Trka kornjaca', 2);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Trka pasa', 2);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Atletika', 1);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Skijanje', 1);   
**INSERT INTO** vrste\_opklade **VALUES** (vrste\_opklada\_id\_seq.**NEXTVAL**, 'Formula1', 2);

**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 1 , 1, 100, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),20000, 5 );  
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 2 , 1, 150, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),20650, 4);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 3 , 5, 1000, 5, TO\_DATE('17.12.2017', dd.mm.yyyy'),246500, 3);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 4 , 5, 1002, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),50000, 5);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 5 , 9, 10, 3, TO\_DATE('17.12.2017', dd.mm.yyyy'),200, 5);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 6 , 11, 900, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),6540000, 2);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 7 , 12, 10, 4, TO\_DATE('17.12.2017', dd.mm.yyyy'),50000, 7);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 8 , 9, 150, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),400, 5);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 9 , 8, 180, 1, TO\_DATE('17.12.2017', dd.mm.yyyy'),50000, 1);   
**INSERT INTO** listic **VALUES** (listic\_id\_seq.**NEXTVAL**, 10 , 5, 40, 2, TO\_DATE('17.12.2017', dd.mm.yyyy'),5550000, 5);

***Upiti***

**--Vraca sve zaposlene cije ime pocinje sa 'K'**

SELECT Ime, Prezime   
FROM zaposleni z  
WHERE z.Ime = 'K%';

**--Vraca sve zaposlene cije prezime sadrzi 'a'**

SELECT Ime, Prezime   
FROM zaposleni z  
WHERE z.Prezime = '%a%';

-–**Vraca sve zaposlene, koji su se zaposlili poslije 2013**

SELECT Ime, Prezime  
FROM zaposleni z  
WHERE To\_Char(z.datumZaposlenja, 'YYYY') > '2013';

-–**Vraca sve zaposlene, koji su se zaposlili prije 1996**

SELECT Ime, Prezime  
FROM zaposleni z  
WHERE To\_Char(z.datumZaposlenja, 'YYYY') < '1996';

-–**Vraca sve listice cija je uplata veca od 1000 KM**

SELECT \*  
FROM listici   
WHERE uplata>1000;

-–**Vraca ime klijenta i broj racuna za klijente koji imaju na racunu 0 KM**

SELECT k.ime, k.racun\_id  
FROM klijenti k, racuni r  
WHERE k.racun\_id=r.racun\_id AND r.stanje=0;

-–**Vraca glavnu poslovnicu**

SELECT poslovnica\_id  
FROM poslovnice p  
WHERE p.nadredjeni\_id IS NULL;

-–**Vraca lokacije gdje se sve nalaze poslovnice u Bosni**

SELECT nazivGrada  
FROM lokacije l  
WHERE l.nazivGrada='BiH' ;

-–**Vraca imena svih direktora**

SELECT z.ime  
FROM zaposleni z, funkcije f  
WHERE f.funkcije\_id=z.funkcije\_id AND f.Naziv = 'Direktor';

-–**Vraca imena klijenata u dobi od 20 do 30 godina**

SELECT ime  
FROM klijenti  
WHERE starost BETWEEN(20,30);

***Grupne funkcije***

-- **Broj zaposlenih**

SELECT Count(\*) brojZaposlenih  
FROM zaposleni;

**-- Maksimalna plata po odjelima**

SELECT o.naziv, Max(plata) najvecaPlata  
FROM zaposleni z, odjel o  
WHERE z.odjel\_id = o.odjel\_id  
GROUP BY o.naziv;

**-- Prosjecna plata veca od 1000 po odjelima i funkcijama**

SELECT z.odjel\_id, o.naziv, Avg(z.plata)  
FROM zaposleni z, funkcija f, odjel o  
WHERE z.funkcija\_id = f.funkcija\_id AND z.odjel\_id = o.odjel\_id  
GROUP BY z.odjel\_id, o.naziv  
HAVING Avg(z.plata) > 1000  
ORDER BY Avg(z.plata);

**-- Broj poslovnica po lokacijama**

SELECT l.naziv, Count(\*)  
FROM lokacija l, poslovnica p  
WHERE p.lokacija\_id = l.lokacija\_id  
GROUP BY l.naziv;

**-- Prosjecno stanje na racunu vece od 500 po pogodnostima**

SELECT pp.pogodnosti\_id, r.racun\_id, Avg(r.stanje)  
FROM pogodnosti pp, racun r  
WHERE r.pogodnosti\_id = pp.pogodnosti\_id   
GROUP BY r.racun\_id, pp.naziv  
HAVING Avg(r.stanje) > 500  
ORDER BY Avg(r.stanje);

***Podupiti***

**-- Svi zaposleni koji imaju vecu platu od zaposlenog 'Kemo'**

SELECT zaposleni\_id, Ime, Prezime  
FROM zaposleni  
WHERE plata > (SELECT plata  
 FROM zaposleni  
 WHERE Prezime = 'Kemo');

**-- Ime, prezime i odjel zaposlenika čija je plata veca od prosjecne plate svih zaposlenika iz 'Podrska'**

SELECT z.zaposleni\_id , z.Ime, z.Prezime, z.odjel\_id  
FROM zaposleni z  
WHERE z.plata > (SELECT Avg(z1.plata)  
 FROM zaposleni z1, odjel o  
 WHERE z1.odjel\_id = o.id AND o.Naziv = 'Podrska');

**-- Ime, prezime klijenta čija je broj opklada veci od prosjecnog broja za ljude koji imaju iste pogodnosti kao i on**

SELECT k.klijenti\_id , k.Ime, k.Prezime  
FROM klijenti k, racun r  
WHERE k.racun\_id=r.racun\_id AND k.broj\_opklada > ( SELECT Avg(r1.broj\_opklada)  
 FROM racun r1  
 WHERE r1.pogodnost =r.pogodnost);

**-- Ime, prezime klijenta ciji listic ima veci obitak od prosjecnog dobitka za istu vrstu opklade**

SELECT k.klijenti\_id , k.Ime, k.Prezime  
FROM klijenti k, racun r  
WHERE k.racun\_id=r.racun\_id AND r.dobitak > ( SELECT Avg(r1.dobitak)  
 FROM racun r1  
 WHERE r1.vrsta\_opklade\_id =r.vrsta\_opklade\_id);

**-- Napisati sve poslovnice cije je stanje vece od stanja centralne poslovnice u BiH**

SELECT p.poslovnica\_id , p.stanje\_sredstava, vp.naziv, l.drzava  
FROM poslovnice p, vrste\_poslovnica vp, lokacija l  
WHERE p.vrsta\_poslovnice\_id=vp.vrsta\_id AND p.lokacija\_id=l.lokacija\_id AND p.stanje\_sredstava > ( SELECT Avg(p1.stanje\_sredstava)  
 FROM poslovnica p1, vrste\_poslovnica vp1, lokacija l1  
 WHERE p1.vrsta\_poslovnice\_id=vp1.vrsta\_id AND p.lokacija\_id=l1.lokacija\_id AND vp1.Naziv LIKE 'Centralna' AND l1.drzava LIKE 'BiH');

***Subtotali***

**-- ROLLUP**

SELECT Ime, Prezime, Sum(plata)  
FROM zaposleni  
GROUP BY ROLLUP(Ime, Prezime);

**-- CUBE**

SELECT k.Ime, k.Prezime, Sum(r.stanje)  
FROM klijent k, racun r   
WHERE r.racun\_id=k.racun\_id  
GROUP BY CUBE(Ime, Prezime);

***Upiti sa TOP N analizom***

-–**TOP 3 zaposlenih sa najvecom platom**

**SELECT** **ROWNUM** **AS** redni\_broj, Ime, Prezime, Plata  
**FROM** (**SELECT** Ime, Prezime, Plata  
 **FROM** zaposleni  
 **ORDER** **BY** plata **DESC**)  
**WHERE** **ROWNUM** <= 3;

-–**TOP 4 poslovnice sa najvecim stanjem sredstava**

**SELECT** **ROWNUM** **AS** redni\_broj, vp.naziv, l.lokacija  
**FROM** (**SELECT** vp.naziv, l.lokacija  
 **FROM** poslovnica p, vrsta\_poslovnice vp, lokacija l  
 **WHERE** p.lokacija\_id = l.lokacija\_id **AND** p.vrsta\_poslovnice\_id = vp.vrsta\_poslovnice\_id  
 **ORDER** **BY** p.Stanje\_sredstava **DESC**)  
**WHERE** **ROWNUM** <= 4;

***Upit sa UNION***

SELECT z.Ime, z.Prezime  
FROM zaposleni z, funkcija f  
WHERE z.funkcija\_id = f.funkcija\_id  
**UNION**  
SELECT f.Naziv  
FROM zaposleni z, funkcija f  
WHERE z.funkcija\_id = f.funkcija\_id;

***Indexi***

Indekse je potrebno kreirati za kolone koje se učestalo koriste u WHERE uvjetu ili JOIN-u.  
Indeks ćemo kreirati za kolonu 'Prezime' u tabeli zaposleni, s obzirom na to da se cesto uzimaju podaci iz ove tabele, koristeci kolonu 'Prezime'.

CREATE INDEX zaposleni\_prezime\_index ON zaposleni(Prezime);

Indeks je kreiran i za kolonu 'Ime' u tabeli klijenti, s obzirom na to da se cesto uzimaju podaci iz ove tabele, koristeci kolonu 'Ime'.

CREATE INDEX klijenti\_ime\_index ON klijenti(Ime);

U where uvjetu cesto ispitujemo situacije koje se 'vrte' oko plate i stanja, pa je kreiram i indeks na kolonu plata iz tabele zaposleni, kao i za kolonu stanje iz tabele racun;

CREATE INDEX zaposleni\_plata\_index ON zaposleni(plata);   
CREATE INDEX racun\_stanje\_index ON racun(stanje);

Indeks je kreiran i za kolonu 'Naziv' u tabeli funkcija, s obzirom na to da se cesto uzimaju podaci iz ove tabele, koristeci kolonu 'Naziv'.

CREATE INDEX funkcija\_naziv\_index ON funkcija(Naziv);

***Funcije***

**-- Vraca platu nekog zaposlenika**

**CREATE OR REPLACE FUNCTION** dajPlatu  
(z\_id IN zaposleni.zaposleni\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
z\_plata zaposleni.plata%TYPE := 0 ;  
**BEGIN**  
**SELECT** z.plata  
**INTO** z\_plata  
**FROM** zaposleni z  
**WHERE** z.zaposleni\_id = z\_id;  
**RETURN** (z\_plata);  
**END** dajPlatu;

**-- Vraca ime nekog zaposlenika**

**CREATE OR REPLACE FUNCTION** dajIme  
(z\_id IN zaposleni.zaposleni\_id%TYPE)  
**RETURN** VARCHAR2  
**IS**  
z\_ime zaposleni.ime%TYPE := 0 ;  
**BEGIN**  
**SELECT** z.ime  
**INTO** z\_ime  
**FROM** zaposleni z  
**WHERE** z.zaposleni\_id = z\_id;  
**RETURN** (z\_ime);  
**END** dajIme;

**-- Vraća vrstu poslovnice, na kojoj se nalazi određeni odjel**

**CREATE OR REPLACE FUNCTION** dajVrstuPoslovnice  
(odjel\_id IN odjel.odjel\_id%TYPE)  
**RETURN** VARCHAR2  
**IS**  
p\_naziv vrsta\_poslovnice.naziv%TYPE := 0 ;  
**BEGIN**  
**SELECT** vrsta\_poslovnice.naziv  
**INTO** p\_naziv  
**FROM** odjel o, poslovnica p, vrsta\_poslovnice vp  
**WHERE** o.poslovnica\_id = p.poslovnica\_id **AND** p.vrsta\_poslovnice\_id = vp.vrsta\_poslovnice\_id **AND** o.odjel\_id = odjel\_id;  
**RETURN** (p\_naziv);  
**END** dajVrstuPoslovnice;

**-- Vraca najvece stanje nekog racuna**

**CREATE OR REPLACE FUNCTION** dajMaxIznos  
(r\_id IN racun.racun\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
r\_stanje racun.stanje%TYPE := 0 ;  
**BEGIN**  
**SELECT** r. stanje  
**INTO** r\_stanje   
**FROM** racun r  
**WHERE** r.racun\_id = r\_id;  
**RETURN** (r\_stanje);  
**END** dajMaxIznos;

**-- Vraca minimalan broj parova**

**CREATE OR REPLACE FUNCTION** dajMinBrojParova  
(p\_id IN parovi.parovi\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
p\_min parovi.minParova%TYPE := 0 ;  
**BEGIN**  
**SELECT** p. minParova  
**INTO** p\_min   
**FROM** parovi p  
**WHERE** p.parovi\_id = p\_id;  
**RETURN** (p\_min);  
**END** dajMinBrojParova;

**-- Vraca dobitak nekog listica**

**CREATE OR REPLACE FUNCTION** dajDobitak  
(l\_id IN listic.listic\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
l\_dobitak listic.dobitak%TYPE := 0 ;  
**BEGIN**  
**SELECT** l.dobitak  
**INTO** l\_dobitak   
**FROM** listic l  
**WHERE** l.listic\_id = l\_id;  
**RETURN** (l\_dobitak);  
**END** dajDobitak;

**-- Vraca stanje neke poslovnice**

**CREATE OR REPLACE FUNCTION** dajStanje  
(p\_id IN poslovnica.poslovnica\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
p\_stanje poslovnica.Stanje\_sredstava%TYPE := 0 ;  
**BEGIN**  
**SELECT** p.Stanje\_sredstava  
**INTO** p\_stanje   
**FROM** poslovnica p  
**WHERE** p.poslovnica\_id = p\_id;  
**RETURN** (p\_stanje);  
**END** dajStanje;

**-- Vraca naziv neke lokacije**

**CREATE OR REPLACE FUNCTION** dajNaziv  
(l\_id IN zaposleni.zaposleni\_id%TYPE)  
**RETURN** VARCHAR2  
**IS**  
l\_naziv lokacija.naziv%TYPE := 0 ;  
**BEGIN**  
**SELECT** l.naziv  
**INTO** l\_naziv   
**FROM** lokacija l  
**WHERE** l.lokacija\_id = l\_id;  
**RETURN** (l\_naziv);  
**END** dajNaziv;

**-- Vraca prezime nekog zaposlenika**

**CREATE OR REPLACE FUNCTION** dajPrezime  
(z\_id IN zaposleni.zaposleni\_id%TYPE)  
**RETURN** VARCHAR2  
**IS**  
z\_prezime zaposleni.prezime%TYPE := 0 ;  
**BEGIN**  
**SELECT** z.prezime  
**INTO** z\_prezime   
**FROM** zaposleni z  
**WHERE** z.zaposleni\_id = z\_id;  
**RETURN** (z\_prezime);  
**END** dajPrezime;

**-- Vraca trenutno stanje neke poslovnice**

**CREATE OR REPLACE FUNCTION** dajStanje  
(p\_id IN poslovnice.poslovnice\_id%TYPE)  
**RETURN** NUMBER  
**IS**  
p\_stanje poslovnice.stanje%TYPE := 0 ;  
**BEGIN**  
**SELECT** p. stanje  
**INTO** p\_stanje   
**FROM** poslovnice p  
**WHERE** p.poslovnice\_id = p\_id;  
**RETURN** (p\_stanje);  
**END** dajStanje;

***Procedure***

**-- Vraca platu nekog zaposlenika**

**CREATE OR REPLACE PROCEDURE** dajPlatu  
(z\_id IN zaposleni.zaposleni\_id%TYPE, z\_plata OUT zaposleni.plata%TYPE)  
**AS**  
**BEGIN**  
**SELECT** zaposleni.plata  
**INTO** z\_plata  
**FROM** zaposleni  
**WHERE** zaposleni\_id = z\_id;  
**END** dajPlatu;

**-- Vraca ime nekog zaposlenika**

**CREATE OR REPLACE PROCEDURE** dajIme  
(z\_id IN zaposleni.id%TYPE, z\_ime OUT zaposleni.ime%TYPE)  
**AS**  
**BEGIN**  
**SELECT** zaposleni.ime  
**INTO** z\_ime  
**FROM** zaposleni  
**WHERE** id = z\_id;  
**END** dajIme;

**-- Vraća vrstu poslovnice, na kojoj se nalazi određeni odjel**

**CREATE OR REPLACE PROCEDURE** dajVrstuPoslovnice  
(odjel\_id IN odjel.odjel\_id%TYPE, p\_naziv OUT vrsta\_poslovnice.naziv%TYPE)  
**AS**  
**BEGIN**  
**SELECT** vrsta\_poslovnice.naziv  
**INTO** p\_naziv  
**FROM** odjel o, poslovnica p, vrsta\_poslovnice vp  
**WHERE** o.poslovnica\_id = p.poslovnica\_id **AND** p.vrsta\_poslovnice\_id = vp.vrsta\_poslovnice\_id **AND** o.odjel\_id = odjel\_id;  
**END** dajVrstuPoslovnice;

**-- Unesi novu vrstu poslovnice**

**CREATE OR REPLACE PROCEDURE** unesiPoslovnicu  
(v\_naziv IN VARCHAR2(45))  
**IS**  
**BEGIN**  
**INSERT INTO** Vrsta\_poslovnice(vrsta\_poslovnice\_id\_seq.NEXTVAL,v\_naziv);   
**COMMIT;**  
**END** unesiPoslovnicu;

**-- Unesi novi racun**

**CREATE OR REPLACE PROCEDURE** UnesiRacun  
(st IN NUMBER, br\_op IN NUMBER, id\_p IN NUMBER)  
**IS**  
**BEGIN**  
**INSERT INTO** racun(racun\_id.seq.NEXTVAL,st,br\_op,id\_p);   
**COMMIT;**  
**END** unesiRacun;

**-- Vraca stanje nekog racuna**

**CREATE OR REPLACE PROCEDURE** dajStanje  
(r\_id IN racun.id%TYPE, r\_stanje OUT racuni.stanje%TYPE)  
**AS**  
**BEGIN**  
**SELECT** r.stanje  
**INTO** r\_stanje   
**FROM** racuni  
**WHERE** r.racun\_id = r\_id;  
**END** dajStanje;

**-- Vraca stanje neke poslovnice**

**CREATE OR REPLACE PROCEDURE** dajStanje  
(p\_id IN poslovnica.poslovnica\_id%TYPE, p\_stanje OUT poslovnica.Stanje\_sredstava%)  
**AS**  
**BEGIN**  
**SELECT** p.Stanje\_sredstava  
**INTO** p\_stanje   
**FROM** poslovnica p  
**WHERE** p.poslovnica\_id = p\_id;  
**END** dajStanje;

**-- Vraca naziv neke lokacije**

**CREATE OR REPLACE PROCEDURE** dajNaziv  
(l\_id IN lokacija.lokacija\_id%TYPE, l\_naziv OUT lokacija.naziv%TYPE)  
**AS**  
**BEGIN**  
**SELECT** l.naziv  
**INTO** l\_naziv   
**FROM** lokacija  
**WHERE** l.id = l\_id;  
**END** dajNaziv;

**-- Vraca prezime nekog zaposlenika**

**CREATE OR REPLACE PROCEDURE** dajPrezime  
(z\_id IN zaposleni.zaposleni\_id%TYPE, z\_prezime OUT zaposleni.prezime%TYPE)  
**AS**  
**BEGIN**  
**SELECT** z.prezime  
**INTO** z\_prezime   
**FROM** zaposleni z  
**WHERE** z.zaposleni\_id = z\_id;  
**END** dajPrezime;

**-- Unosi naziv neke lokacije**

**CREATE OR REPLACE PROCEDURE** unesiNaziv  
( l\_naziv IN lokacija.naziv%TYPE, drz IN VARCHAR2(45), porez IN DECIMAL(3))  
**AS**  
**BEGIN**  
**INSERT INTO** lokacija(lokacija\_id\_seq.NEXTVAL,l\_naziv, drz,porez);   
**COMMIT;**  
**END** unesiNaziv;

***Trigeri***

-- Provjera datuma zaposlenog

CREATE OR REPLACE TRIGGER ProvjeraDatuma  
BEFORE UPDATE OF id ON funkcija  
FOR EACH ROW   
WHEN (NEW.funkcija\_id = 1)  
BEGIN  
IF (To\_Char(SYSDATE, 'yyyy') - To\_Char(:old.datum\_Zaposlenja, 'yyyy')) < 5  
THEN   
Raise\_Application\_Error (-20000, 'Osoba koja je zaposlena manje od 5 godina, ne može postati direktor!');  
END IF;   
END;

-- Provjera negativne plate

CREATE AND REPLACE TRIGGER trigger\_za\_negativnu\_platu

BEFORE INSERT OR UPDATE OF plata

ON zaposlenici

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.plata < 0 THEN

Raise\_Application\_Error(-23010,'Plata ne smije biti negativna');

END\_IF;

END;

-- Provjera postojanja poslovnice

CREATE OR REPLACE TRIGGER ProvjeraPoslovnice  
BEFORE INSERT OR UPDATE OF poslovnica\_id ON odjel  
FOR EACH ROW   
DECLARE ID poslovnica.poslovnica\_id%TYPE;  
BEGIN  
SELECT poslovnica\_id  
INTO ID  
FROM poslovnica  
WHERE id=:new.poslovnica\_id   
EXCEPTION  
WHEN No\_Data\_Found THEN NULL;  
IF (ID IS null)  
THEN   
Raise\_Application\_Error (-20500, Odjel ne moze pripadati poslovnici koja ne postoji!');  
END IF;   
END;

-- Provjera negativne uplate za listic

CREATE AND REPLACE TRIGGER trigger\_za\_negativnu\_uplatu

BEFORE INSERT OR UPDATE OF uplata

ON listic

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.uplata < 0 THEN

Raise\_Application\_Error(-24012,'Uplata ne smije biti negativna');

END\_IF;

END;

-- Provjera negativnog dobitka

CREATE AND REPLACE TRIGGER trigger\_za\_negativan\_dobitak

BEFORE INSERT OR UPDATE OF dobitak

ON listic

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.dobitak < 0 THEN

Raise\_Application\_Error(-27012,'Dobitak ne smije biti negativan');

END\_IF;

END;

-- Provjera starosne dobi klijenta

CREATE AND REPLACE TRIGGER trigger\_za\_maloljetnike

BEFORE INSERT OR UPDATE OF starost

ON klijenti

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.starost < 18 THEN

Raise\_Application\_Error(-23017,'Klijent mora biti punoljetan!');

END\_IF;

END;

-- Provjera negativnog stanja racuna

CREATE AND REPLACE TRIGGER trigger\_za\_negativan\_racun

BEFORE INSERT OR UPDATE OF stanje

ON racun

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.stanje < 0 THEN

Raise\_Application\_Error(-23512,'Racun ne smije biti u minusu!');

END\_IF;

END;

-- Provjera starosne dobi zaposlenog

CREATE OR REPLACE TRIGGER ProvjeraDatuma  
BEFORE UPDATE OF datumRodjenja ON zaposleni  
FOR EACH ROW   
BEGIN  
IF (To\_Char(SYSDATE, 'yyyy') - To\_Char(:new.datum\_Rodjenja, 'yyyy')) < 18  
THEN   
Raise\_Application\_Error (-27700, 'Ne mozemo zaposliti maloljetnu osobu!');  
END IF;   
END;

-- Provjera negativnih parova

CREATE AND REPLACE TRIGGER trigger\_za\_parove

BEFORE INSERT OR UPDATE OF minParovi

ON parovi

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.minParovi < 0 THEN

Raise\_Application\_Error(-27777,'Morate odabrati minimalno jedan par!');

END\_IF;

END;

-- Provjera negativnog broja subjekata

CREATE AND REPLACE TRIGGER trigger\_za\_subjekte

BEFORE INSERT OR UPDATE OF broj\_subjekata

ON vrsta\_opklade

FOR EACH ROW

REFERENCING NEW AS NEW OLD AS OLD

BEGIN

IF :new.broj\_subjekata < 0 THEN

Raise\_Application\_Error(-27877,'Mora biti minimalno jedan subjekat!');

END\_IF;

END;

*PL/SQL SKRIPTA*

Moja skripta ce koristiti kursor da selektira 5 najvise placenih zaposlenika iz tabele zaposleni.

DECLARE

CURSOR kursor1

IS

SELECT ime,funkcija\_id,plata

FROM zaposleni

ORDER BY plata DESC;

Ime1 VARCHAR2(10);

Funkcija\_id1 NUMBER(4);

Plata1 DECIMAL(3);

BEGIN

OPEN kursor1;

FOR i IN 1..5 LOOP

FETCH kursor1 INTO ime1, funkcija\_id1,plata1;

EXIT WHEN kursor1%NOTFOUND;

INSERT INTO temp VALUES (plata1,funkcija\_id1,ime1);

COMMIT;

END LOOP;

CLOSE kursor1;

END;