Univerzitet u Beogradu
Fakultet organizacionih nauka

Projektni rad iz predmeta “Baze podataka 2”

Trening centar

Mentor: Nenad Aničić
Student: Irena Karalić

Beograd, 2020.

Sadržaj

1. Opis podsistema

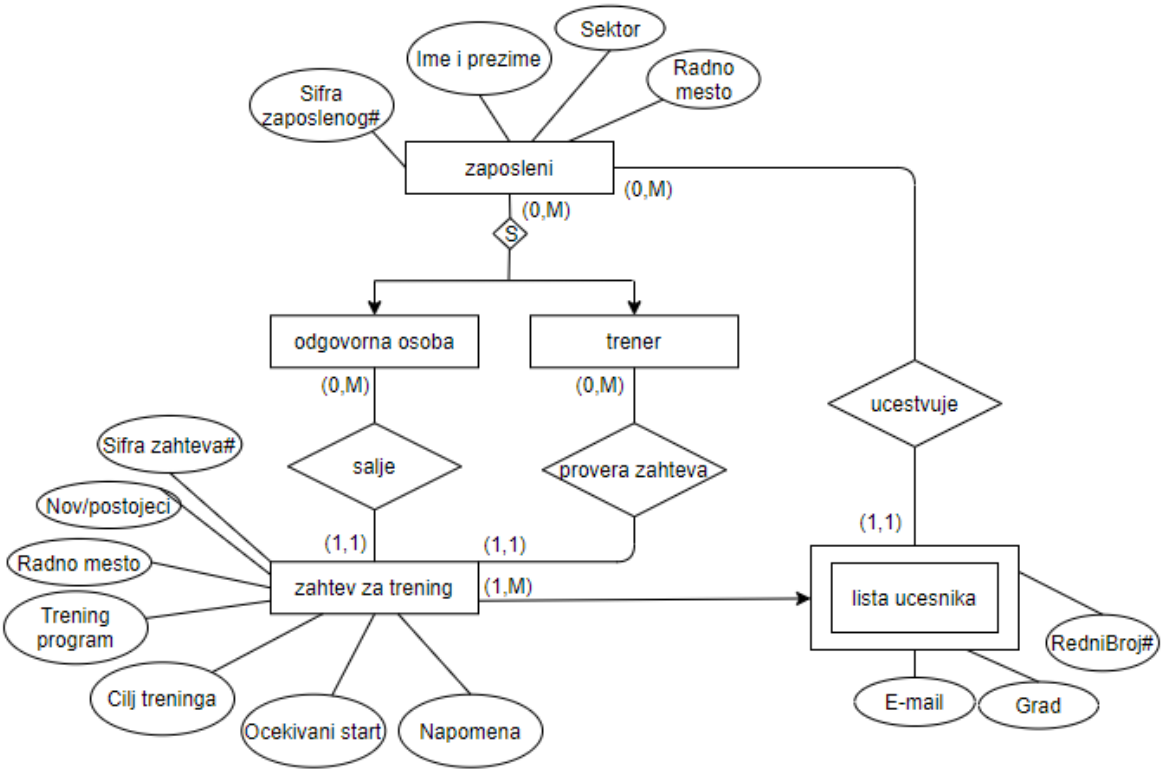
Trening proces započinje odgovorna osoba (direktor ili rukovodilac) koja šalje *zahtev za trening* na e-mail adresu Trening centra. U ime Trening centra trener proučava zahtev i ako je zahtev potpun i opravdan šalje *plan treninga* rukovodiocu trening centra i podnosiocu zahteva na usaglašavanje. Nakon što je plan usaglašen i odobren, trener organizuje trening, a podnosioc zahteva obaveštava i organizuje dolazak učesnika. Trener priprema trening materjial u skladu sa potrebama ciljne grupe, predmetnom temom i ciljem treninga. Osnovno trening sredstvo je *Prezentacija* (.ppt).

Na osnovu plana treninga i svih prethodno preduzetih aktivnosti u cilju pripreme, trener izvodi obuku za zaposlene. Na kraju treninga učesnici *test*, a povremeno radi provere znanja trener može dodatno testirati učesnike. Učesnici se nakon treninga potpisuju na *Prisustvo na treningu*i popunjavaju *Evaluacioni upitnik*. Nakon svakog treninga, trener priprema *Izveštaj o treningu* koji dostavlja rukovodiocu trening centra i podnosiocu zahteva. Izveštaj sadrži sve detalje o treningu, učesnicima, rezultatima testa, materijalima, zapažanjima i preporukama trenera.

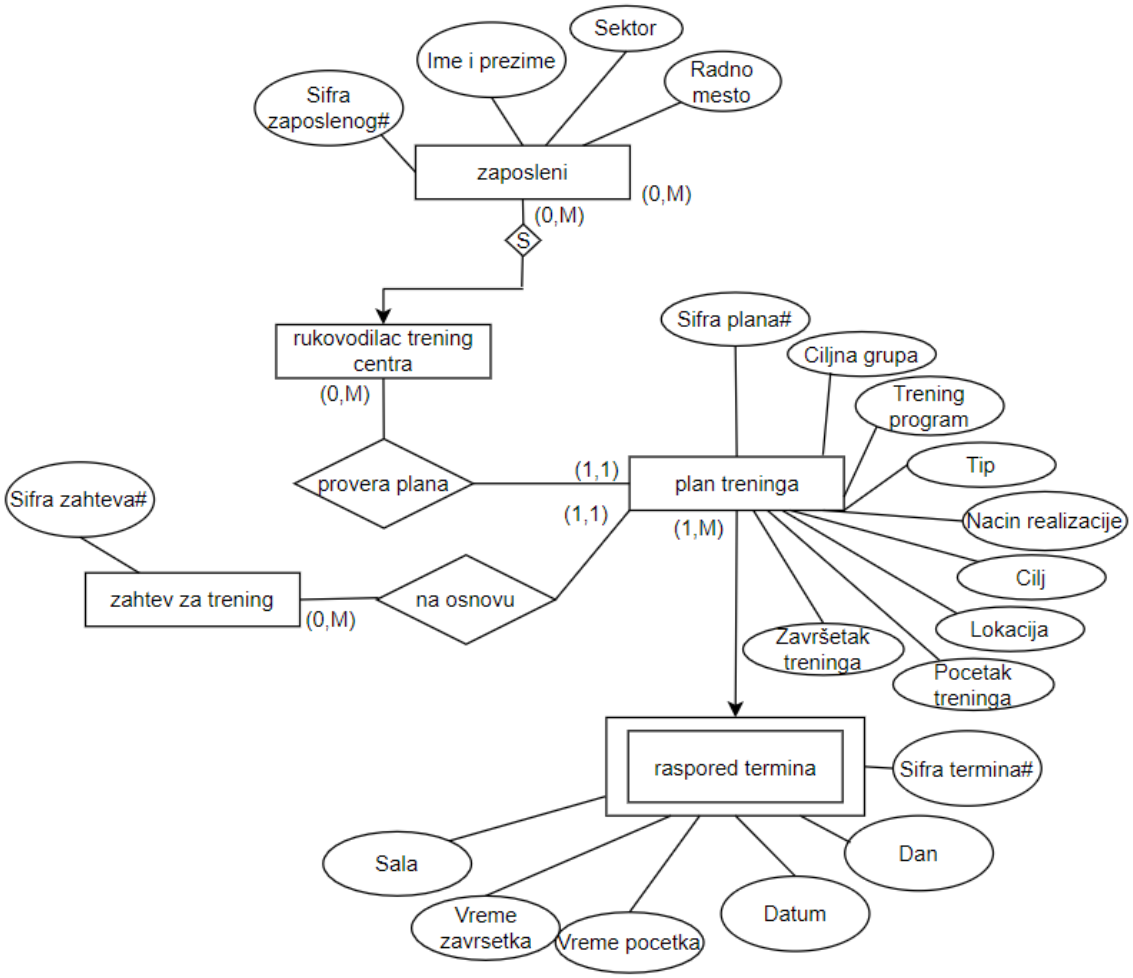
Poslovna dokumenta koja se koriste u procesu su: zahtev za trening, plan treninga, prezentacija, prisustvo na treningu, test, evaluacioni upitnik i izveštaj o treningu.

2. Model podataka - PMOV

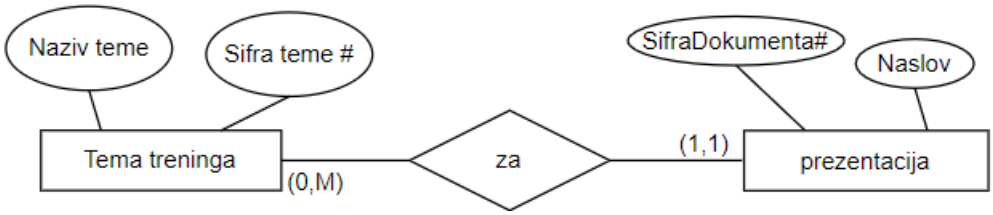
2.1. Podmodel ZahtevZaTrening



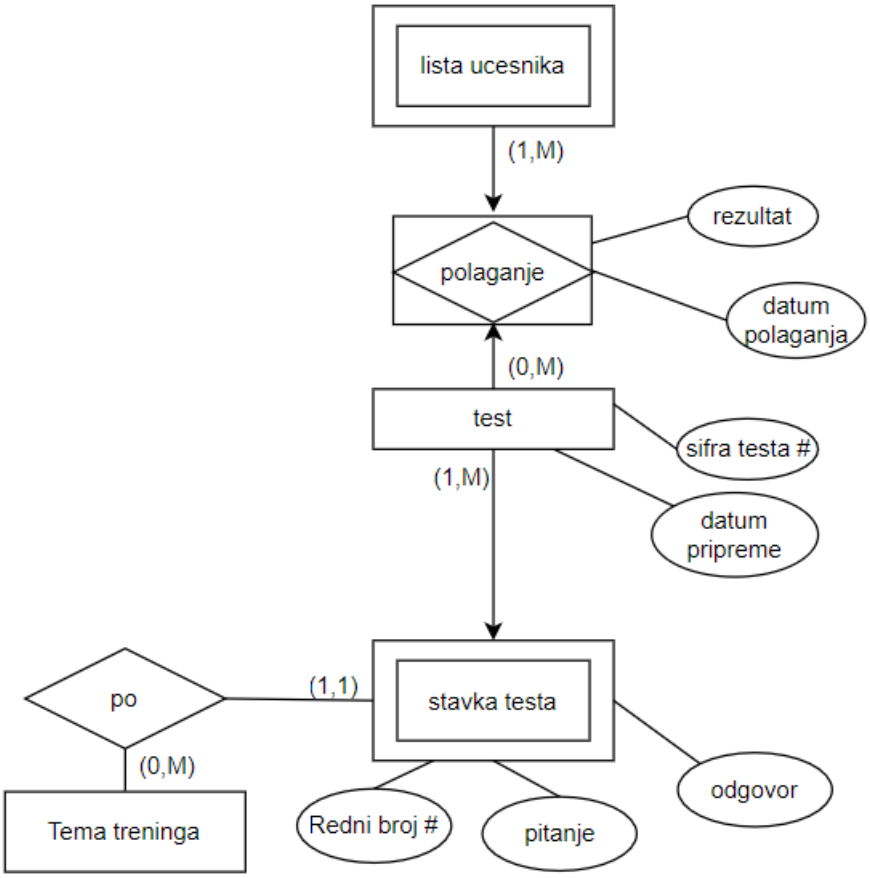
2.2. Podmodel PlanTreninga



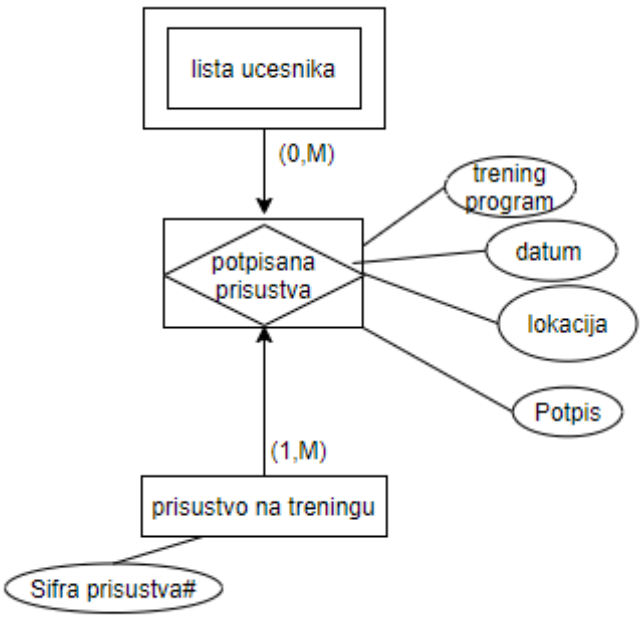
2.3. Podmodel Prezentacija



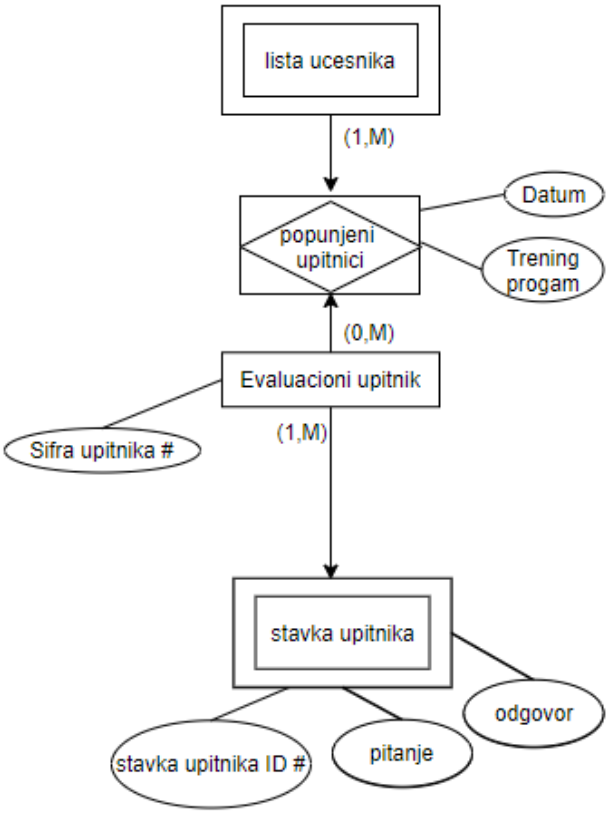
2.4. Podmodel Test



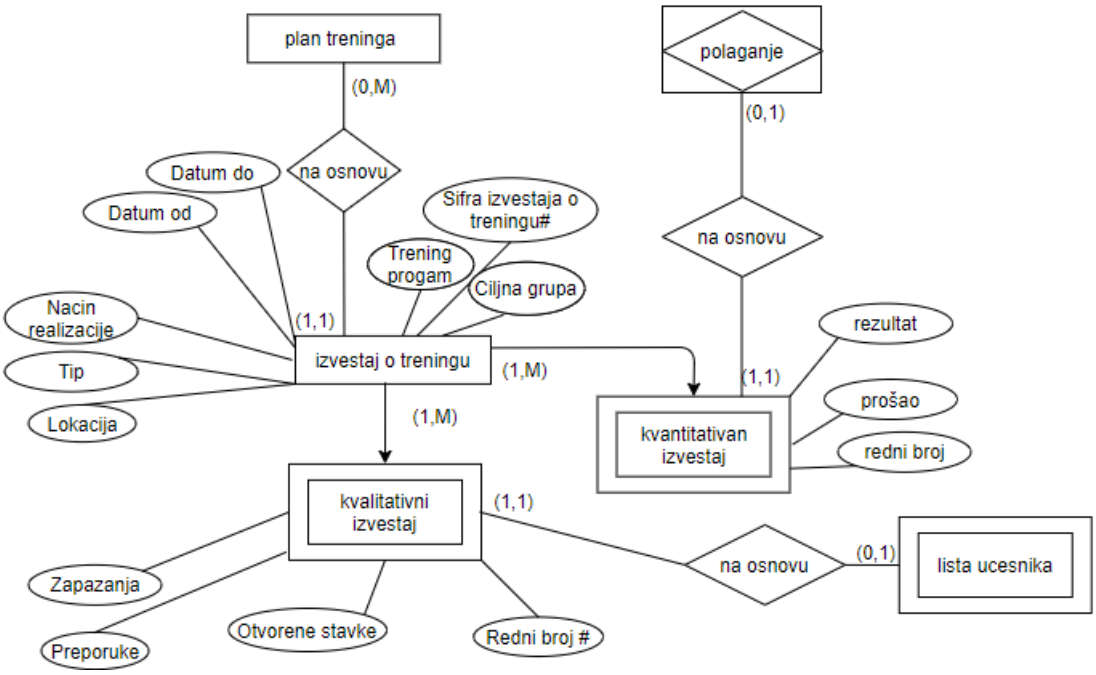
2.5. Podmodel PrisustvoNaTreningu



2.6. Podmodel EvaluacioniUpitnik



2.7. Podmodel IzvestajOTreningu



3. Relacioni model

Zaposleni (SifraZaposlenog, ImeiPrezime, Sektor, RadnoMesto)

OdgovornaOsoba (SifraZaposlenog)

Trener (SifraZaposlenog)

RukovodilacTreningCentra (SifraZaposlenog)

ZahtevZaTrening (SifraZahteva, NoviIliPostojeci, RadnoMesto, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ListaUcesnika (SifraZahteva, RedniBroj, Grad, Email, *SifraZaposlenog*)

PlanTreninga (SifraPlana, CiljnaGrupa, TreningProgram, Tip, NacinRealizacije, Cilj, Lokacija, PocetakTreninga, ZavrsetakTreninga, *SifraZahteva,RukovodilacTreningCentraID*)

RasporedTermina (SifraPlana, SifraTermina, Dan, Datum, VremePocetka, VremeZavrsetka, Sala, *SifraTeme*)

TemaTreninga (SifraTeme, NazivTeme)

Prezentacija (SifraDokumenta, Naslov,*SifraTeme*, *TrenerID*)

Test (SifraTesta, DatumPripreme)

StavkaTesta (SifraTesta, RedniBroj, Pitanje, Odgovor, *SifraTeme*)

Polaganje (SifraTesta, SifraZahteva, RedniBroj, DatumPolaganja, Rezultat)

PrisustvoNaTreningu (SifraPrisustva)

PotpisanaPrisustva (SifraPrisustva, SifraZahteva, RedniBroj, TreningProgram, Datum, Lokacija, Potpis)

EvaluacioniUpitnik (SifraUpitnika)

PopunjeniUpitnici (SifraUpitnika, SifraZahteva, RedniBroj, TreningProgram, Datum)

StavkaUpitnika (SifraUpitnika, StavkaUpitnikaID, Pitanje, Odgovor)

IzvestajOTreningu (SifraIzvestajaOTreningu, TreningProgram, CiljnaGrupa, NacinRealizacije, Tip, Lokacija, PocetakTreninga, ZavrsetakTreninga, *SifraPlana*)

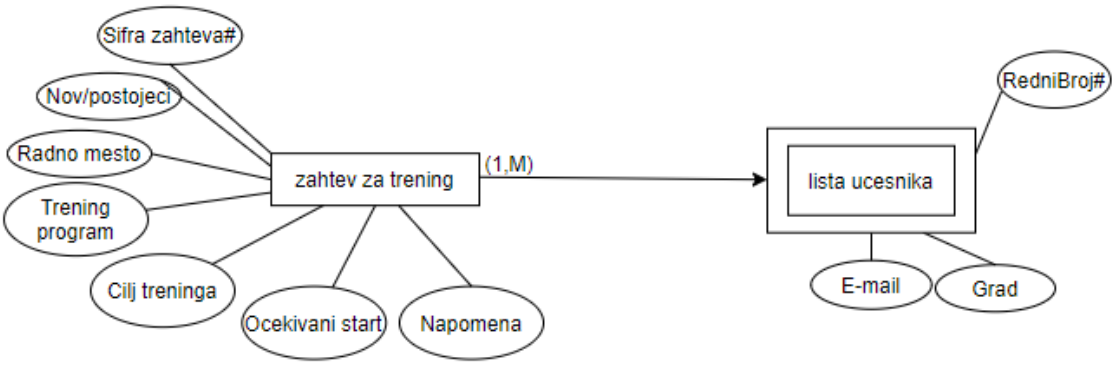
KvalitativniIzvestaj (SifraIzvestajaOTreningu, BrojID, Zapazanja, Preporuke, OtvoreneStavke, *SifraZahteva*, *RedniBroj*)

KvantitativniIzvestaj (SifraIzvestajaOTreningu, BrojID, Rezultat, Prosao, *SifraZahteva*, *RedniBroj*)

4. Denormalizacija relacija

4.1. Denormalizacija 2NF: Pre-joining tehnika

- Normalizovani konceptualni model



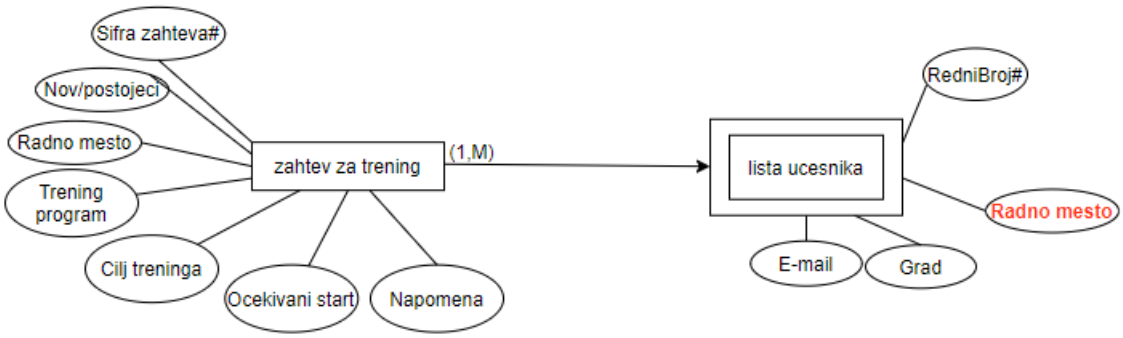
ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto , TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ListaUcesnika (SifraZahteva, RedniBroj, Grad, Email, *SifraZaposlenog*)

- Funkcionalni zahtev (listaUcesnika):

SifraZahteva, RedniBroj -> E-mail, Grad, SifraZaposlenog

- Denormalizovani konceptualni model



ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto , TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ListaUcesnika (SifraZahteva, RedniBroj, RadnoMesto, Grad, Email, *SifraZaposlenog*)

- Funkcionalnizahtev(lista ucesnika)

SifraZahteva, RedniBroj ->E-mail, Grad, RadnoMesto**,** *SifraZaposlenog*

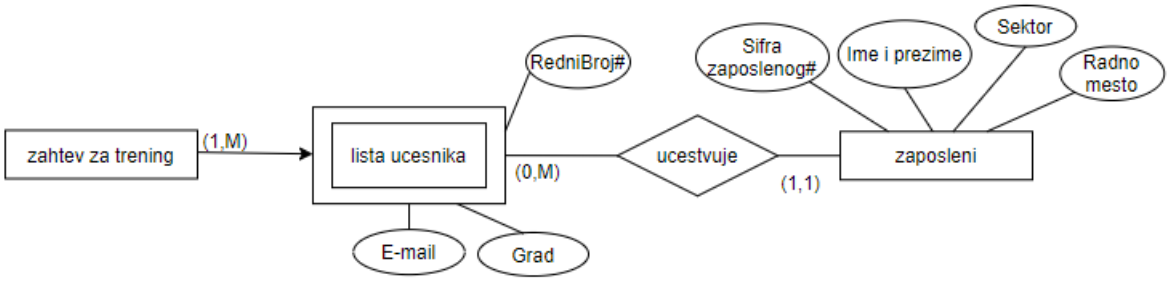
SifraZahteva ->RadnoMesto

Tabela specifikacije trigera za primer denormalizacije bi bila:

Tabela	Tip trigera	Kolona	Potreban	Šta treba da uradi?
ZahtevZaTrening	insert		NE	
	update	RadnoMesto	DA	Izmenjenu vrednost ažurira u tabeli ListaUcesnika.
	delete		NE	
ListaUcesnika	insert		DA	Ažurira vrednost kolone RadnoMesto na osnovu unete vrednosti kolone SifraZahteva.
	update	RadnoMesto	DA	Zabraniti direktno ažuriranje ove kolone.
		SifraZahteva	DA	Zabraniti direktno ažuriranje ove kolone.
	delete		NE	

4.2. Denormalizacija relacija 3NF: Pre-joining tehnika

- Normalizovani konceptualni model



Zaposleni (SifraZaposlenog, ImeiPrezime, Sekor, RadnoMesto)

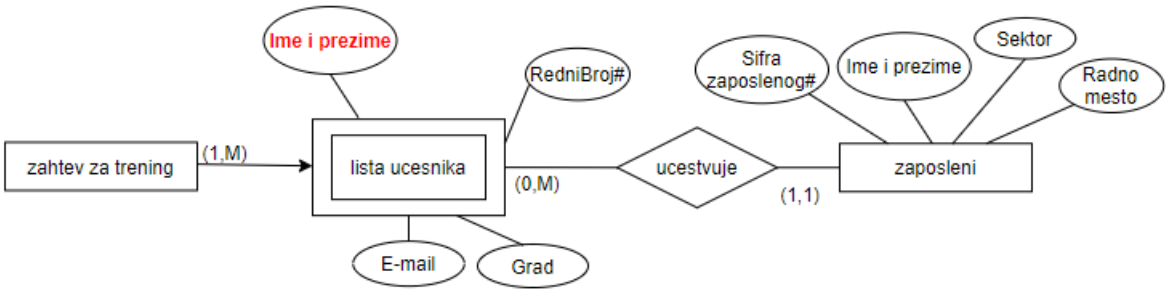
ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ListaUcesnika (SifraZahteva, RedniBroj, RadnoMesto, Grad, Email, *SifraZaposlenog*)

- Funkcionalni zahtev (ListaUcesnika)

SifraZahteva, RedniBroj -> E-mail, Grad, RadnoMesto, SifraZaposlenog

- Denormalizovani konceptualni model



Zaposleni (SifraZaposlenog, ImeiPrezime, Sekor, RadnoMesto)

ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto , TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ListaUcesnika (SifraZahteva, RedniBroj, ImeIPrezime, RadnoMesto, Grad, Email, *SifraZaposlenog*)

- Funkcionalni zahtev (ListaUcesnika)

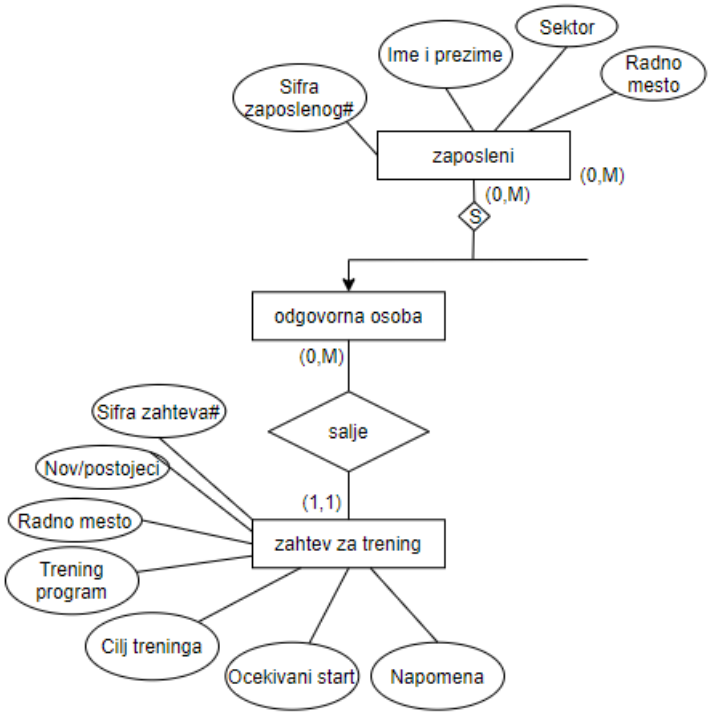
SifraZahteva, RedniBroj -->E-mail, Grad, RadnoMesto, ImeIPrezime, SifraZaposlenog

SifraZaposlenog -->ImeIPrezime

Tabela specifikacije trigera za primer denormalizacije bi bila:

Tabela	Tip trigera	Kolona	Potreban	Šta treba da uradi?
Zaposleni	insert		NE	
	update	ImeIPrezime	DA	Izmenjenu vrednost ažurira u tabeli ListaUcesnika.
	delete		NE	
ListaUcesnika	insert		DA	Ažurira vrednost kolone ImeIPrezime na osnovu unete vrednosti kolone SifraZaposlenog.
	update	SifraZaposlenog	DA	Ažurira vrednost kolone ImeIPrezime na osnovu izmenjene vrednosti SifraZaposlenog.
		ImeIPrezime	DA	Zabraniti direktno ažuriranje ove kolone.
	delete		NE	

- Normalizovani konceptualni model



Zaposleni (SifraZaposlenog, ImeiPrezime, Sekor, RadnoMesto)

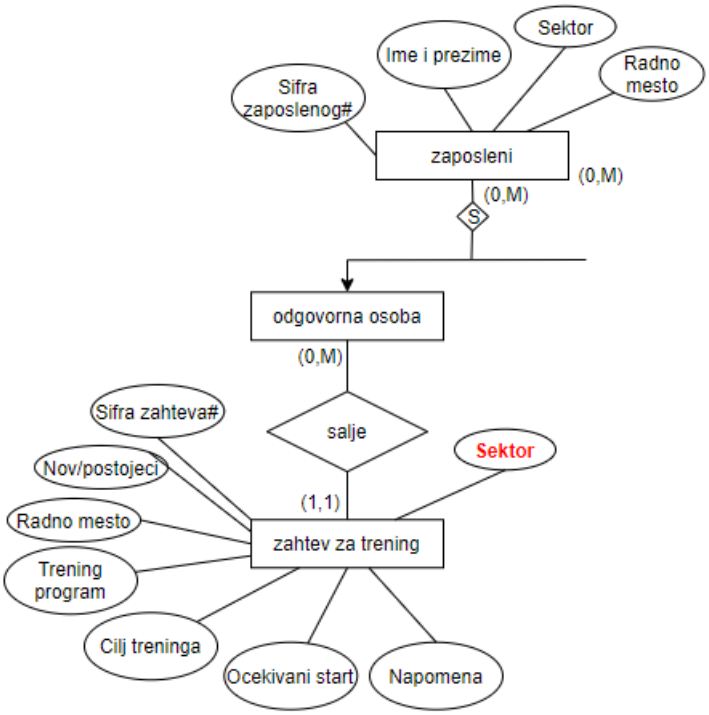
OdgovornaOsoba(SifraZaposlenog)

ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

- Funkcionalni zahtev (ZahtevZaTrening)

SifraZahteva ->Nov/postojeci, RadnoMesto , TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*

- Denormalizovani konceptualni model



Zaposleni (SifraZaposlenog, ImeiPrezime, Sekor, RadnoMesto)

OdgovornaOsoba(SifraZaposlenog)

ZahtevZaTrening (SifraZaheva, Nov/postojeci, RadnoMesto, Sektor, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

- Funkcionalni zahtev (ListaUcesnika)

SifraZahteva ->Nov/postojeci, RadnoMesto,Sektor, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, OdgovornaOsobaID, TrenerID

OdgovornaOsobaID ->Sektor

Tabela specifikacije trigera za primer denormalizacije bi bila:

Tabela	Tip trigera	Kolona	Potreban	Šta treba da uradi?
Zaposleni	insert		NE	
	update		NE	
	delete		NE	
ZahtevZaTrening	insert	Sektor	DA	Ažurira vrednost kolone Sektor na osnovu unete vrednosti kolone OdgovornaOsobaID.
	update	OdgovornaOsobaID	DA	Zabraniti direktno ažuriranje ove kolone.
		Sektor	DA	Zabraniti direktno ažuriranje ove kolone.
	delete		NE	

5. Korisnički definisani tipovi podataka

5.1. Objektni tip sa jednim atributom

```
CREATE OR REPLACE TYPE GradTip AS OBJECT (
grad varchar2(2 CHAR)
)
FINAL

/;


CREATE TABLE listaUcesnikaTreninga (
        SifraZahteva NUMBER NOT NULL,
        RedniBroj NUMBER,
        E-mail VARCHAR2(30),
        Grad GradTip,
        RadnoMesto VARCHAR2(30),
        ImeIPrezime VARCHAR2(30),
        SifraZaposlenog VARCHAR2(30),
constraint listaUcesnikaTreninga_pkprimary key (SifraZahteva),
constraint listaUcesnikaTreninga_fk foreign key (SifraZaposlenog) references
zaposleni(SifraZaposlenog));

INSERT INTO LISTAUCESNIKATRENINGA (SIFRAZAHTEVA,
REDNIBROJ,IMEIPREZIME,RADNOMESTO, GRAD, EMAIL,SIFRAZAPOSLENOG) VALUES
(1,2, '','',GRADTIP('BG'), '', 7);

SELECT U.SIFRAZAHTEVA, U.REDNIBROJ, U.IMEIPREZIME, U.RADNOMESTO,
U.GRAD.GRAD, U.EMAIL, U.SIFRAZAPOSLENOG
FROM LISTAUCESNIKATRENINGA U;
```

5.2. Objektni tip sa više atributa

```
CREATE OR REPLACE TYPE INFORMACIJEOTRENINGU AS OBJECT (
CILJNAGRUPA varchar2(30),
TRENINGPROGRAM varchar2(30),
TIP varchar2(30),
NACINREALIZACIJE varchar2(30),
LOKACIJA varchar2(30),
POCETAKTRENINGA DATE,
ZAVRSETAKTRENINGA DATE,
MEMBER FUNCTION GETCILJNAGRUPA return varchar2,
MEMBER FUNCTION GETTRENINGPROGRAM RETURN varchar2,
MEMBER FUNCTION GETTIP RETURN varchar2,
MEMBER FUNCTION GETNACINREALIZACIJE RETURN varchar2,
MEMBER FUNCTION GETLOKACIJA RETURN varchar2,
MEMBER FUNCTION GETPOCETAKTRENINGA RETURN DATE,
MEMBER FUNCTION GETZAVRSETAKTRENINGA RETURN DATE,
MEMBER FUNCTION dodeliTreningProgram RETURN varchar2

)

INSTANTIABLE NOT FINAL;

/

CREATE OR REPLACE TYPE BODY INFORMACIJEOTRENINGU AS
MEMBER FUNCTION GETCILJNAGRUPA RETURN varchar2 IS
BEGIN
RETURN SELF.CILJNAGRUPA;
END;
MEMBER FUNCTION GETTRENINGPROGRAM RETURN varchar2 IS
BEGIN
RETURN SELF.TRENINGPROGRAM;
END;
MEMBER FUNCTION GETTIP RETURN varchar2 IS
BEGIN
RETURN SELF.TIP;
END;
MEMBER FUNCTION GETNACINREALIZACIJE RETURN varchar2 IS
BEGIN
RETURN SELF.NACINREALIZACIJE;
END;
MEMBER FUNCTION GETLOKACIJA RETURN varchar2 IS
BEGIN
RETURN SELF.LOKACIJA;
END;
MEMBER FUNCTION GETPOCETAKTRENINGA RETURN DATE IS
BEGIN
RETURN SELF.POCETAKTRENINGA;
END;
MEMBER FUNCTION GETZAVRSETAKTRENINGA RETURN DATE IS
BEGIN
RETURN SELF.ZAVRSETAKTRENINGA;
END;
MEMBER FUNCTION dodeliTreningProgram RETURN varchar2 IS
BEGIN
```

```
CASE SELF.CILJNAGRUPA
    WHEN 'agent' THEN RETURN 'Uvodni trening';
   WHEN 'tim lider' THEN return 'Refreshment';
   WHEN 'rukovodilac' THEN return 'Komunikacija';
    ELSE  RETURN 'Novo';
END CASE;
END;
END;

CREATE TABLE NoviPlanTreninga (
SIFRAPLANA number NOT NULL,
TRENINGINFORMACIJE INFORMACIJEOTRENINGU,
CILJ VARCHAR2(30),
SIFRAZAHTEVA number NOT NULL,
SIFRARUKOVODIOCATRENINGCENTRA NUMBER NOT NULL,
constraint NoviPlanTreninga_pk primary key (SIFRAPLANA),
constraint NoviPlanTreninga_Fk1 foreign key (SIFRAZAHTEVA) references
ZAHTEVZATRENING(SIFRAZAHTEVA),
constraint NoviPlanTreninga_Fk2 foreign key (SIFRARUKOVODIOCATRENINGCENTRA)
references RUKOVODILACTRENINGCENTRA (SIFRAZAPOSLENOG)
);

INSERT INTO NOVIPLANTRENINGA
(sifraplana,treninginformacije,cilj,sifrazahteva,sifrarukovodiocatreningcentra)
VALUES (1, informacijeotreningu('agent', 'uvodni trening', 'teorijski', 'live', 'Beograd',
'22-JAN-2021','15-FEB-2021'),
'savladati osnove', 1, 15);

select P.SIFRAPLANA,
P.TRENINGINFORMACIJE.GETCILJNAGRUPA(),P.TRENINGINFORMACIJE.DODELITRENIN
GPROGRAM(), P.TRENINGINFORMACIJE.GETTIP(),
P.TRENINGINFORMACIJE.GETNACINREALIZACIJE(),
P.TRENINGINFORMACIJE.GETLOKACIJA(),
P.TRENINGINFORMACIJE.GETPOCETAKTRENINGA(),P.TRENINGINFORMACIJE.GETZAV
RSETAKTRENINGA(),P.SIFRAZAHTEVA, P.SIFRARUKOVODIOCATRENINGCENTRA
FROM NOVIPLANTRENINGA P;
```

6. Trigeri

6.1. Trigeri – denormalizacija 2NF

```
CREATE OR REPLACE TRIGGER promeniRadnoMesto
AFTER UPDATE OF radnoMesto
ON zahtevZaTreningDetalji
FOR EACH ROW
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
EXECUTE IMMEDIATE 'ALTER TRIGGER ZABRANAIZMENERADNOGMESTA DISABLE';
UPDATE listaUcesnikaTreninga
SET radnoMesto = :NEW.radnoMesto
WHERE sifraZahteva = :OLD.sifraZahteva;
COMMIT;
BEGIN
EXECUTE IMMEDIATE 'ALTER TRIGGER ZABRANAIZMENERADNOGMESTA ENABLE';
END;
END;

CREATE OR REPLACE TRIGGER ucesnici
BEFORE INSERT
ON listaUcesnikaTreninga
FOR EACH ROW
DECLARE
v_radnoMesto varchar2(30);
BEGIN
SELECT radnoMesto INTO v_radnoMesto
FROM zahtevZaTreningDetalji
WHERE sifraZahteva = :NEW.sifraZahteva;
:NEW.radnoMesto:= v_radnoMesto;
END;

CREATE OR REPLACE TRIGGER zabranaIzmeneRadnogMesta
BEFORE UPDATE OF radnoMesto
ON listaUcesnikaTreninga
FOR EACH ROW
BEGIN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena radnog mesta u
okviru tabele lista učesnika.');
END;

CREATE OR REPLACE TRIGGER zabranaIzmeneSifreZahteva
BEFORE UPDATE OF sifraZahteva
ON listaUcesnikaTreninga
FOR EACH ROW
BEGIN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena šifre zahteva u okviru
tabele lista učesnika.');
END;
```

6.2. Trigeri – denormalizacija 3NF

```
CREATE OR REPLACE TRIGGER promeniImeIPrezime
AFTER UPDATE OF imeIPrezime
ON zaposleni
FOR EACH ROW
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
EXECUTE IMMEDIATE 'ALTER TRIGGER ZABRANAIZMENEIMENAIPREZIMENA DISABLE';
UPDATE listaUcesnikaTreninga
SET imeIPrezime = :NEW.imeIPrezime
WHERE sifraZaposlenog = :OLD.sifraZaposlenog;
COMMIT;
BEGIN
EXECUTE IMMEDIATE 'ALTER TRIGGER ZABRANAIZMENEIMENAIPREZIMENA ENABLE';
END;
END;

CREATE OR REPLACE TRIGGER unosUcesnika
BEFORE INSERT
ON listaUcesnikaTreninga
FOR EACH ROW
DECLARE
v_imeIPrezime varchar2(30);
v_radnomesto varchar2(30);
BEGIN
SELECT imeIPrezime,radnomesto INTO v_imeIPrezime,v_radnomesto
FROM zaposleni
WHERE sifraZaposlenog= :NEW.sifraZaposlenog;
:NEW.imeIPrezime:= v_imeIPrezime;
:NEW.radnomesto:= v_radnomesto ;
END;

CREATE OR REPLACE TRIGGER promenaSifreZaposlenog
BEFORE UPDATE OF sifraZaposlenog
ON listaUcesnika
FOR EACH ROW
DECLARE
v_imeIPrezime varchar2(30);
BEGIN
SELECT imeIPrezimeINTO v_imeIPrezime
FROM zaposleni
WHERE sifraZaposlenog= :NEW.sifraZaposlenog;
:NEW.imeIPrezime:= v_imeIPrezime;
END;

CREATE OR REPLACE TRIGGER zabranaIzmeneImenaIPrezimena
BEFORE UPDATE OF imeIPrezime
ON listaUcesnika
FOR EACH ROW
BEGIN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena imena i prezimena u
okviru tabele lista učesnika.');
END;
```

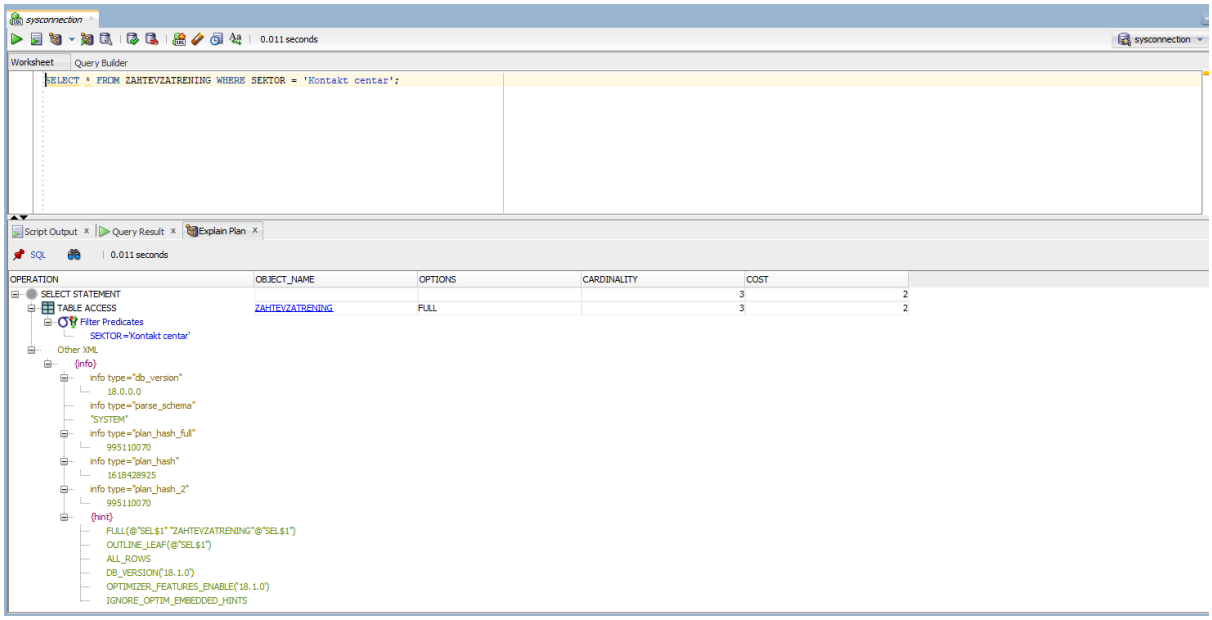
```
CREATE or REPLACE TRIGGER unosZahtevaZaTreningOsnovno
BEFORE INSERT
ON zahtevZaTreningOsnovno
FOR EACH ROW
DECLARE
v_sektor varchar2(30);
BEGIN
SELECT sektor INTO v_sektor
FROM zaposleni
WHERE sifraZaposlenog = :NEW.odgovornaOsobaid;
:NEW.sektor:= v_sektor;
END;

CREATE OR REPLACE TRIGGER zabranaIzmeneOdgovorneOsobe
BEFORE UPDATE OF odgovornaosobaid
ON zahtevzatreningosnovno
FOR EACH ROW
BEGIN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena odgovorne osobe u
okviru tabele zahtev za trening.');
END;

CREATE OR REPLACE TRIGGER zabranaIzmeneSektora
BEFORE UPDATE OF sektor
ON zahtevZaTreningOsnovno
FOR EACH ROW
BEGIN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena sektora u okviru
tabele zahtevZaTrening.');
END;
```

7. Optimizacija baze podataka

7.1. Indeksi

Indeksi nad tekstualnim poljem

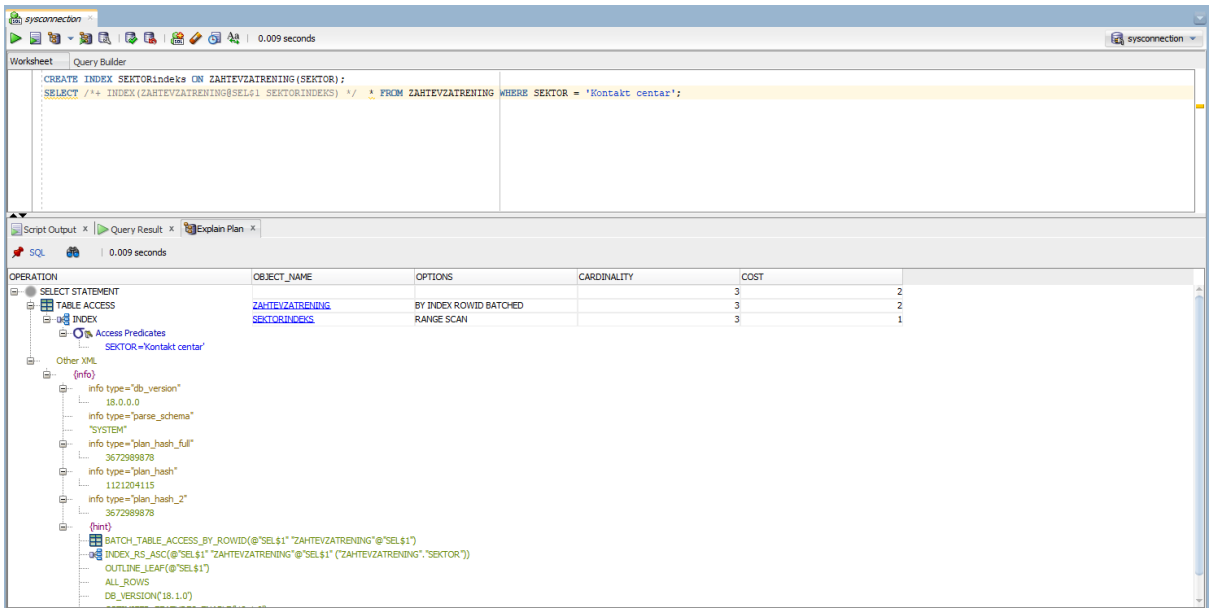
SELECT * FROM ZAHTEVZATRENING WHERE SEKTOR = 'Kontakt centar';



Iz Explain Plan prozora uočavamo da je vreme izvršavanja upita 0,011 sekundi, da je trošak 2, dok je metod pretrage full.

Sada cemo kreirati index nad tekstualnim poljem SEKTOR u tabeli ZAHTEVZATRENING:

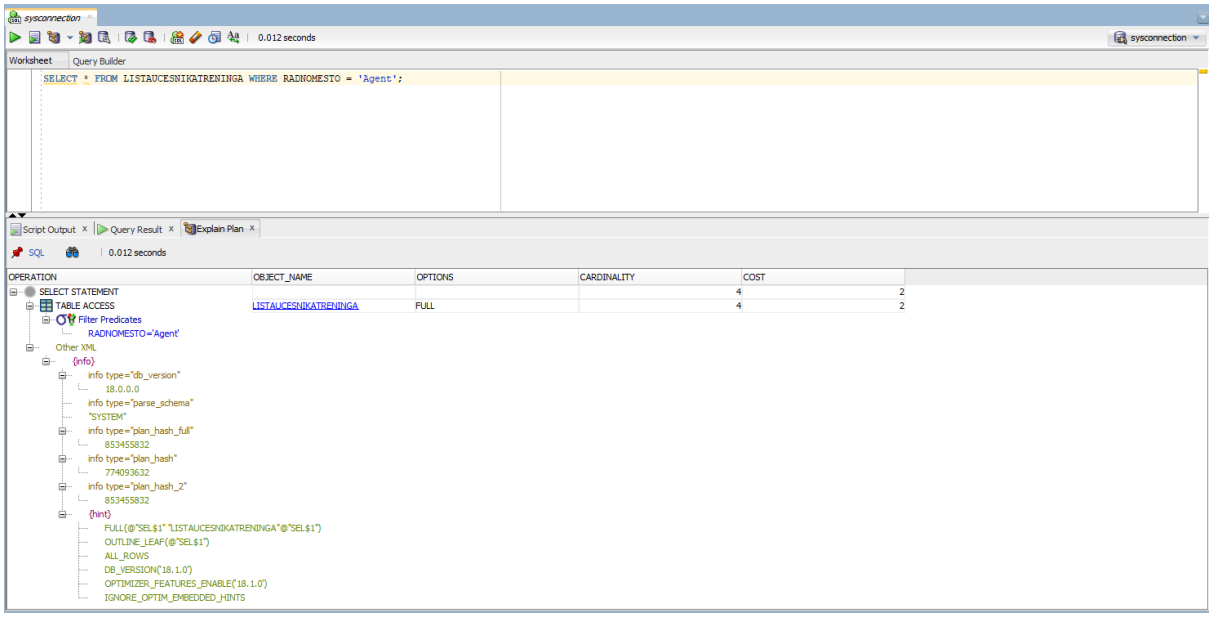
CREATE INDEX SEKTORindeks ON ZAHTEVZATRENINGOSNOVNO(SEKTOR);
SELECT * FROM ZAHTEVZATRENING WHERE SEKTOR = 'Kontakt centar';



Iz Explain Plan prozora uočavamo da se vreme izvršavanja upita smanjilo na 0,009 sekundi, a trošak na 1. Metod pretrage je range scan, a možemo videti i da je korišćen kreirani index.

Iz navedenog zaključujemo da indexi doprinose smanjenju vremena izvršavanja čak i na malom skupu podataka kao što je ovde slučaj, dok bi na većem skupu podataka mogli imati veoma značajnu uštedu vremena.

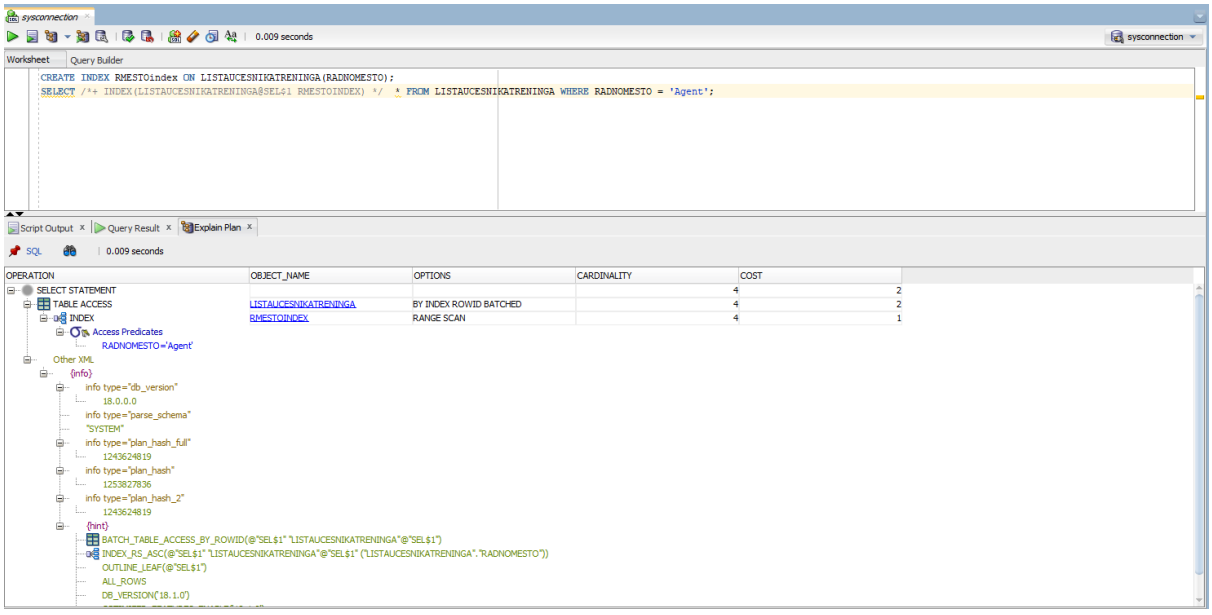
SELECT * FROM LISTAUCESNIKATRENINGA WHERE RADNOMESTO = 'Agent';



Iz Explain Plan prozora uočavamo da je vreme izvršavanja upita 0,012 sekundi, da je trošak 2, dok je metod pretrage full.

Sada cemo kreirati index nad tekstualnim poljem SEKTOR u tabeli LISTAUCESNIKATRENINGA:

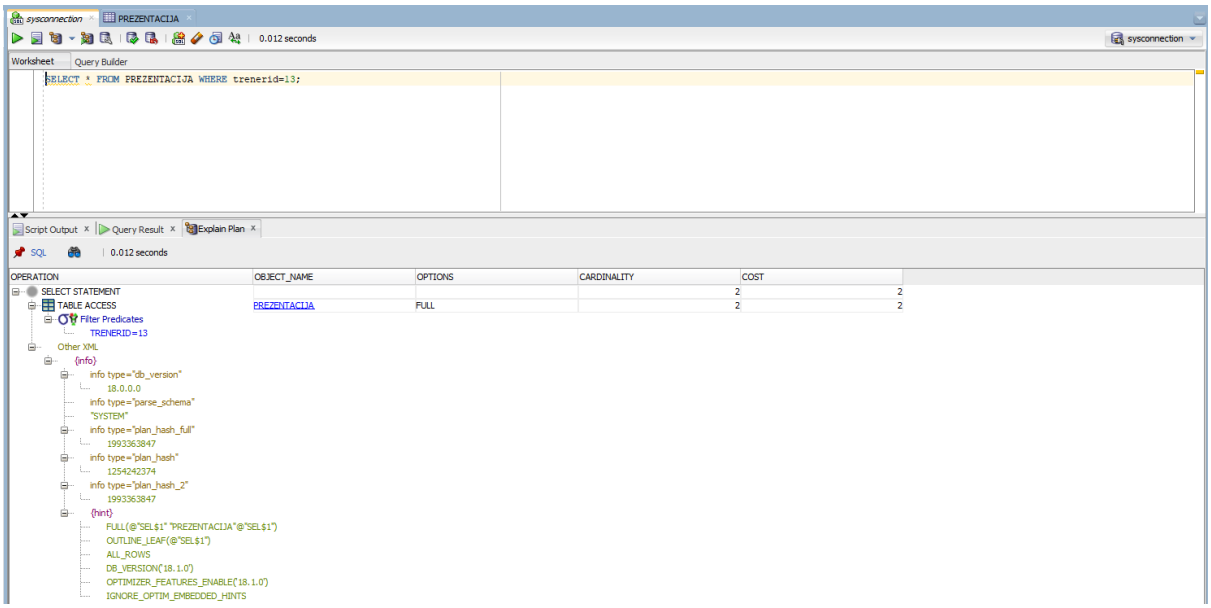
CREATE INDEX RMESTOindex ON LISTAUCESNIKATRENINGA(RADNOMESTO);
SELECT * FROM LISTAUCESNIKATRENINGA WHERE RADNOMESTO = 'Agent';



Iz Explain Plan prozora uočavamo da se vreme izvršavanja upita smanjilo na 0,009 sekundi, a trošak na 1. Metod pretrage je range scan, a možemo videti i da je korišćen kreirani index.
Iz navedenog zaključujemo da indexi doprinose smanjenju vremena izvršavanja čak i na malom skupu podataka kao što je ovde slučaj, dok bi na većem skupu podataka mogli imati veoma značajnu uštedu vremena.

Indeks nad spoljnim ključem

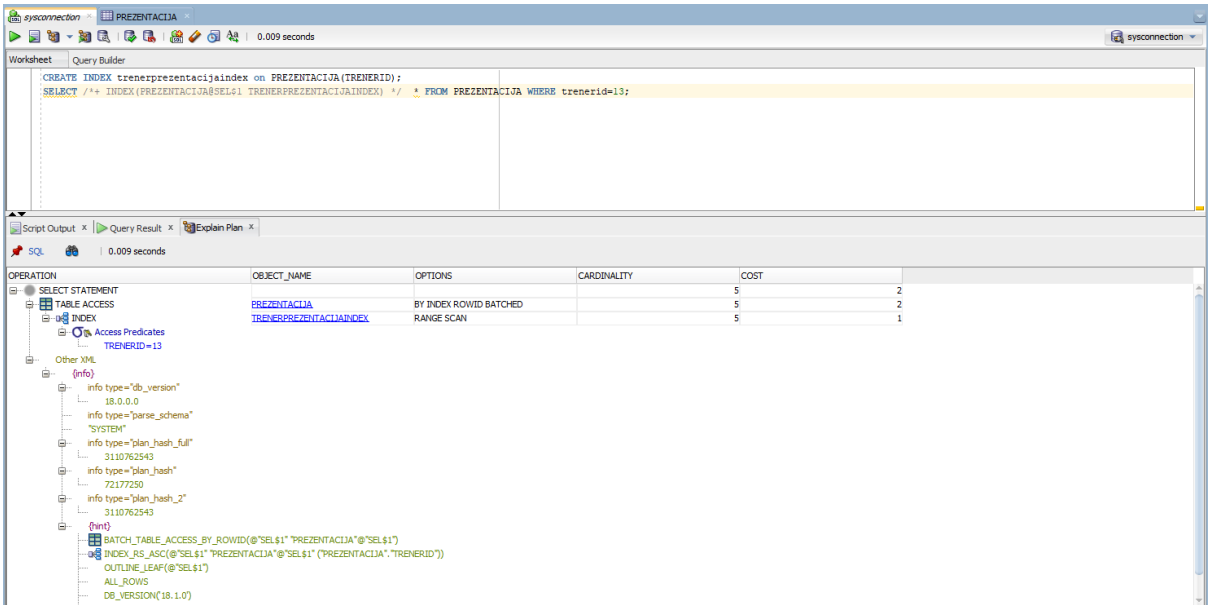
SELECT * FROM PREZENTACIJA WHERE trenerid=13;



Iz Explain Plan prozora uočavamo da je vreme izvršavanja upita 0,012 sekundi, da je trošak 2, dok je metod pretrage full.

Sada cemo kreirati index nad tekstualnim poljem TRENERID u tabeli PREZENTACIJA:

CREATE INDEX trenerprezentacijaindex on PREZENTACIJA(TRENERID);
SELECT * FROM PREZENTACIJA WHERE trenerid=13;



Iz Explain Plan prozora uočavamo da se vreme izvršavanja upita smanjilo na 0,009 sekundi, a trošak na 1. Metod pretrage je range scan, a možemo videti i da je korišćen kreirani index.
Iz navedenog zaključujemo da indexi doprinose smanjenju vremena izvršavanja čak i na malom skupu podataka kao što je ovde slučaj, dok bi na većem skupu podataka mogli imati veoma značajnu uštedu vremena.

7.2. Horizontalno particionisanje

```
CREATE TABLE "SYSTEM"."TEMATRENINGA_PARTICIJE"

("SIFRATEME" NUMBER NOT NULL,
"NAZIVTEME" VARCHAR2(30 BYTE) NOT NULL ENABLE,
CONSTRAINT "TEMATRENINGAPART_PK" PRIMARY KEY ("SIFRATEME")ENABLE
)

PARTITION BY LIST (NAZIVTEME)
( PARTITION p_tema1 VALUES ('Uvod'),
PARTITION p_tema2 VALUES ('Usluge'),
PARTITION p_tema3 VALUES ('Aplikacije'),
PARTITION p_tema4 VALUES ('Oprema'),
PARTITION p_tema5 VALUES ('SAP')
);

CREATE TABLE "SYSTEM"."PREZENTACIJA_PARTICIJE"
( "SIFRADOKUMENTA" NUMBER NOT NULL ENABLE,
"NASLOV" VARCHAR2(30 BYTE),
"SIFRATEME" NUMBER NOT NULL ENABLE ,
"TRENERID" NUMBER NOT NULL ENABLE,
CONSTRAINT "PREZENTACIJAPART_PK" PRIMARY KEY ("SIFRADOKUMENTA") ENABLE,
CONSTRAINT "PREZENTACIJAPART_FK1" FOREIGN KEY ("SIFRATEME")
REFERENCES "SYSTEM"."TEMATRENINGA_PARTICIJE" ("SIFRATEME") ENABLE,
CONSTRAINT "PREZENTACIJAPART_FK2" FOREIGN KEY ("TRENERID")
REFERENCES "SYSTEM"."TRENER" ("SIFRAZAPOSLENOG") ENABLE
)

  PARTITION BY REFERENCE (PREZENTACIJAPART_FK1);
```

7.3. Vertikalno particionisanje

ZahtevZaTrening (SifraZahteva, NoviIliPostojeci, RadnoMesto, Sektor, TreningProgram, CiljTreninga, OcekivaniStart, Napomena, *OdgovornaOsobaID*, *TrenerID*)

ZahtevZaTreningOsnovno (SifraZahteva, NoviIliPostojeci, Sektor, TreningProgram, OcekivaniStart, *TrenerID*)

ZahtevZaTreningDetalji (SifraZaheva, RadnoMesto , CiljTreninga, Napomena, *OdgovornaOsobaID*)

```
CREATE TABLE "SYSTEM"."ZAHTEVZATRENINGOSNOVNO"
  (     "SIFRAZAHTEVA" NUMBER NOT NULL ENABLE,
        "NOVILIPOSTOJECI" VARCHAR2(30 BYTE),
        "SEKTOR" VARCHAR2(30 BYTE),
        "TRENINGPROGRAM" VARCHAR2(30 BYTE),
        "OCEKIVANISTART" DATE,
        "TRENERID" NUMBER,
```

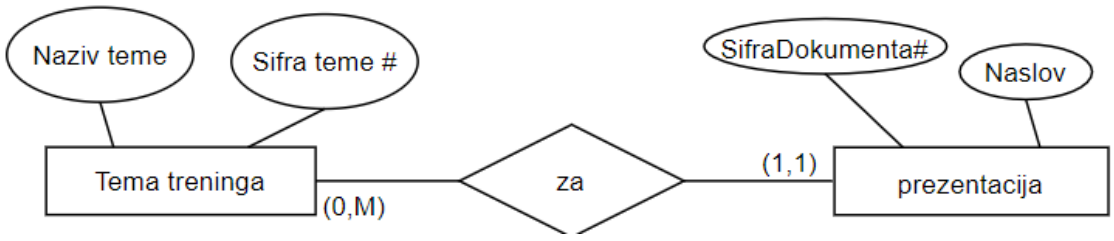
```
        CONSTRAINT "ZAHTEVZATRENINGOSNOVNO_PK" PRIMARY KEY
("SIFRAZAHTEVA")ENABLE,
        CONSTRAINT "ZAHTEVZATRENINGOSNOVNO_FK2" FOREIGN KEY ("TRENERID")
         REFERENCES "SYSTEM"."TRENER" ("SIFRAZAPOSLENOG") ENABLE
  );

CREATE TABLE "SYSTEM"."ZAHTEVZATRENINGDETALJI"
  (     "SIFRAZAHTEVA" NUMBER NOT NULL ENABLE,
        "RADNOMESTO" VARCHAR2(30 BYTE),
        "CILJTRENINGA" VARCHAR2(30 BYTE),
        "NAPOMENA" VARCHAR2(30 BYTE),
        "ODGOVORNAOSOBAID" NUMBER,
         CONSTRAINT "ZAHTEVZATRENINGDETALJI_PK" PRIMARY KEY
("SIFRAZAHTEVA") ENABLE,
         CONSTRAINT "ZAHTEVZATRENINGDETALJI_FK" FOREIGN KEY
("ODGOVORNAOSOBAID")
          REFERENCES "SYSTEM"."ODGOVORNAOSOBA" ("SIFRAZAPOSLENOG")
ENABLE
         );

CREATE OR REPLACE VIEW ZAHTEVZATRENINGPOGLED AS
SELECT ZO.SIFRAZAHTEVA, zo.novilipostojeci, ZO.SEKTOR, ZO.TRENINGPROGRAM,
ZO.OCEKIVANISTART, ZO.TRENERID,
ZD.RADNOMESTO, ZD.CILJTRENINGA, ZD.NAPOMENA, ZD.ODGOVORNAOSOBAID
FROM zahtevzatreningosnovno ZO, zahtevzatreningdetalji ZD
WHERE zo.sifrazahteva = zd.sifrazahteva;

CREATE OR REPLACE TRIGGER ZAHTEVZATRENINGUNOS
INSTEAD OF INSERT ON ZAHTEVZATRENINGPOGLED
REFERENCING NEW AS novi
FOR EACH ROW
BEGIN
INSERT INTO zahtevzatreningosnovno (SIFRAZAHTEVA, NOVIILIPOSTOJECI,SEKTOR,
TRENINGPROGRAM,OCEKIVANISTART,TRENERID,ODGOVORNAOSOBAID)
VALUES (:novi.SIFRAZAHTEVA, :novi.NOVIILIPOSTOJECI,:novi.SEKTOR,
:novi.TRENINGPROGRAM,:novi.OCEKIVANISTART,:novi.TRENERID,:novi.ODGOVORNAOS
OBAID);
INSERT INTO zahtevzatreningdetalji (SIFRAZAHTEVA, RADNOMESTO,CILJTRENINGA,
NAPOMENA)
VALUES (:novi.SIFRAZAHTEVA, :novi.RADNOMESTO,:novi.CILJTRENINGA,
:novi.NAPOMENA);
END;

INSERT INTO zahtevzatreningpogled VALUES (10, 'novi','kontakt centar','uvodni trening', '27-
FEB-21', 13, 'agent','savladati osnove','',1);

CREATE OR REPLACE TRIGGER ZAHTEVZATRENINGIZMENA
INSTEAD OF UPDATE ON zahtevzatreningpogled
REFERENCING NEW AS novi
```

```
FOR EACH ROW
BEGIN
UPDATE zahtevzatreningosnovno SET NOVIILIPOSTOJECI =
:novi.NOVIILIPOSTOJECI,SEKTOR = :novi.SEKTOR, TRENINGPROGRAM =
:novi.TRENINGPROGRAM,OCEKIVANISTART = :novi.OCEKIVANISTART, TRENERID =
:novi.TRENERID, ODGOVORNAOSOBAID = :novi.ODGOVORNAOSOBAID
WHERE SIFRAZAHTEVA = :novi.SIFRAZAHTEVA;
UPDATE zahtevzatreningdetalji SET RADNOMESTO = :novi.RADNOMESTO,CILJTRENINGA
= :novi.CILJTRENINGA, NAPOMENA = :novi.NAPOMENA
WHERE SIFRAZAHTEVA = :novi.SIFRAZAHTEVA;
END;

UPDATE zahtevzatreningpogled
SET ocekivanistart = '01-MAR-21', napomena = 'fokus na pakete'
WHERE sifrazahteva = 10;

CREATE OR REPLACE TRIGGER ZAHTEVZATRENINGBRISANJE
INSTEAD OF DELETE ON zahtevzatreningpogled
BEGIN
DELETE FROM zahtevzatreningosnovno WHERE sifrazahteva = :OLD.sifrazahteva;
DELETE FROM zahtevzatreningdetalji WHERE sifrazahteva = :OLD.sifrazahteva;
END;

DELETE FROM zahtevzatreningpogled
WHERE sifrazahteva=10;
```

7.4. Primena drugih optimizacionih tehnika

Hard – Coded Value

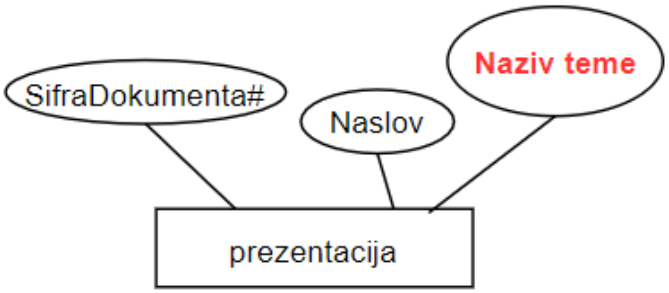
- Početni konceptualni model



TemaTreninga (<u>SifraTeme</u>, NazivTeme)

Prezentacija (<u>SifraDokumenta</u>, Naslov, *SifraTeme, TrenerID*)

- Konceptualni model posle optimizacije

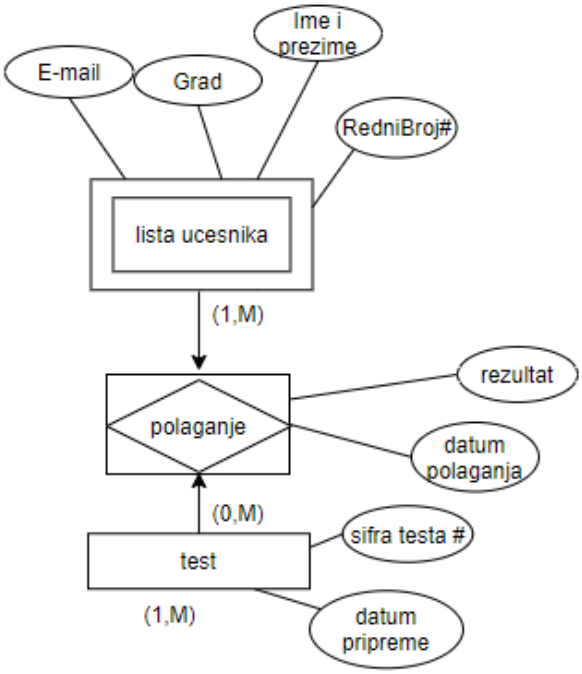


Prezentacija (SifraDokumenta, Naslov, NazivTeme*, TrenerID*)

ALTER TABLE PREZENTACIJA ADD CONSTRAINT checkOgrPrezentacija CHECK (NazivTeme IN ('Uvod', 'Usluge',));

Storing Derivable Value

- Početni konceptualni model

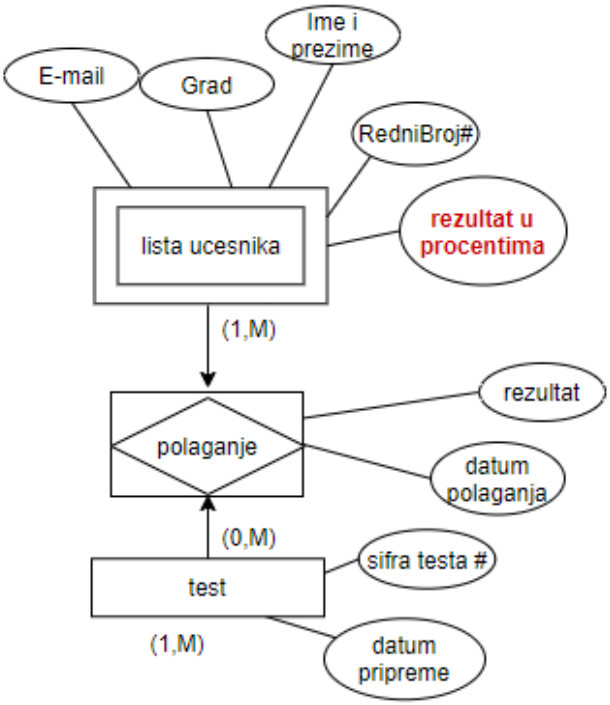


ListaUcesnika (SifraZahteva, RedniBroj, ImeiPrezime, RadnoMesto, Grad, Email, *SifraZaposlenog*)

Polaganje (SifraZahteva, RedniBroj, SifraTesta, DatumPolaganja, Rezultat)

Test (SifraTesta, DatumPripreme)

- Konceptualni model posle optimizacije



ListaUcesnikaTreninga (SifraZahteva, RedniBroj, ImeiPrezime, RadnoMesto, Grad, Email, RezultatUProcentima, *SifraZaposlenog*)

Polaganje (SifraZahteva, RedniBroj, SifraTesta, DatumPolaganja, Rezultat)

Test (SifraTesta, DatumPripreme)

Tabela specifikacije trigera

Tabela	Tip trigera	Kolona	Potreban	Sta treba da uradi?
ListaUcesnikaTreninga	INSERT	RezultatUProcentima	DA	Sprečava direktan unos
	UPDATE	RezultatUProcentima	DA	Sprečava direktnu izmenu
	DELETE		NE	
Polaganje	INSERT	RezultatUProcentima	DA	Poziva proceduru da ažurira vrednost kolone RezultatUProcentima u tabeli ListaUčesnika.
	UPDATE		NE	

	DELETE		NE	
Test	INSERT		NE	
	UPDATE		NE	
	DELETE		NE	

```
CREATE OR REPLACE TRIGGER zabranaUnosaRezultataUProcentima
BEFORE INSERT ON ListaUcesnikaTreninga
FOR EACH ROW
BEGIN
IF :NEW.rezultatUProcentima !=0 THEN
RAISE_APPLICATION_ERROR(-20000,'Nije dozvoljen direktan unos rezultata u procentima u
tabeli lista ucesnika.');
END IF;
END;

CREATE OR REPLACE PACKAGE paket AS
pomocna number:=0;
sifraTesta varchar2(30);
sifraZahteva varchar2(30);
redniBrojUcesnika number;
END paket;

CREATE OR REPLACE TRIGGER zabranaIzmeneRezultataUProcentima
BEFORE UPDATE OF rezultatUProcentima
ON listaUcesnikaTreninga
FOR EACH ROW
BEGIN
IF paket.pomocna = 0 THEN
RAISE_APPLICATION_ERROR(-20000, 'Nije dozvoljena direktna izmena rezultata u
procentima u tabeli lista ucesnika.');
END IF;
END;

CREATE OR REPLACE TRIGGER sacuvajSifre
BEFORE INSERT OR DELETE OR UPDATE ON polaganje
FOR EACH ROW
BEGIN
IF(INSERTING) THEN
BEGIN paket.sifraTesta := :NEW.sifraTesta;
paket.sifraZahteva := :NEW.sifraZahteva;
paket.redniBrojUcesnika := :NEW.redniBroj;
END;
ELSE
BEGIN paket.sifraTesta := :OLD.sifraTesta;
paket.sifraZahteva := :OLD.sifraZahteva;
paket.redniBrojUcesnika := :OLD.redniBroj;
END;
END IF;
END;
```

```
CREATE OR REPLACE TRIGGER procenat
AFTER INSERT OR UPDATE OR DELETE ON polaganje
DECLARE
sifraTesta varchar2(30):= paket.sifraTesta;
sifraZahteva varchar2(30):= paket.sifraZahteva;
redniBroj number := paket.redniBrojUcesnika;
BEGIN
paket.pomocna := 1;
izracunajProcenat(sifraTesta, sifraZahteva, redniBroj);
paket.pomocna := 0;
END;

CREATE OR REPLACE PROCEDURE izracunajProcenat (sifTesta IN varchar2, sifZahteva IN
varchar2, rbUcesnika IN number) AS
testUradjen boolean:= true;
rezultat number:=0;
procenat number:=0;
BEGIN
SELECT polaganje.rezultat INTO rezultat
FROM polaganje
WHERE SIFRATESTA = sifTesta AND SIFRAZAHTEVA = sifZahteva AND
REDNIBROJUCESNIKA = rbUcesnika;
IF(rezultat IS null) THEN
testUradjen := false;
ELSE
SELECT (rezultat/20) INTO procenat
FROM polaganje
WHERE SIFRATESTA = sifTesta AND SIFRAZAHTEVA = sifZahteva AND
REDNIBROJUCESNIKA = rbUcesnika;
END IF;
UPDATE ListaUcesnikaTreninga
SET rezultatUProcentima = procenat
WHERE SIFRAZAHTEVA = sifZahteva AND REDNIBROJUCESNIKA = rbUcesnika;
END;
```

8. TEHNOLOGIJE ZA IMPLEMENTACIJU PROJEKTA

8.1. SUBP korišćen za implementaciju baze podatka

Oracle baza podataka je multimodelski sistem za upravljanje bazama podataka koji proizvodi I prodaje kompanija Oracle Corporation. Ona se obično koristi za izvršavanje mrežne obrade transakcija, skladištenje podataka I mešovitih radnih opterećenja baza podataka.

Ova baza je vrlo složena i potrebno je tehničko predznanje za rad sa Oracle bazom podataka. Oracle baza podataka je korisna samo u slučaju baratanja ogromnom količinom podataka.

To je baza podataka koja se obično koristi za izvršavanje obrade transakcija na mreži (OLTP), skladištenje podataka (DB) I mešovitih radnih opterećenja (OLTP i DB).

Usluge su dostupne lokalno (on-prem) i u oblaku (on-cloud).

Postoji besplatna verzija koja je svima dostupna (pošto ovde govorimo o besplatnih verzijama) pod nazivom Oracle Database 18c Express Edition (XE). To je moćna Oracle baza podataka i nju koriste kompanije širom sveta. Može se koristiti u bilo kom okruženju, a postoji mogućnost ugradnje i ponovne distribucije.

8.2. Programsko okruženje za razvoj korisničkog interfejsa

Dizajn i struktura aplikacije urađeni su u razvojnom okruženju Netbeans, koje je jedno od najkvalitetnijih razvojnih okruženja otvorenog koda (IDE) na tržištu i omogućava razvoj aplikacija na mnogim programskim jezicima, ali je prvenstveno namenjeno razvoju Java aplikacija. NetBeans je razvijen korišćenjem programskog jezika Java i njegova dostupnost je omogućena na svim platformama. Nastao je iz studentskog projekta. Nakon stavljanja proizvoda na tržište postigao je veliki uspeh, a 1996. godine ga je kupila već poznata kompanija Sun Microsistems, a nakon godinu dana NetBeans je pušten pod licencom otvorenog kodaPrema mogućnostima i karakteristikama NetBeans IDE, podudara se sa ostalim komercijalnim alatima, a zbog svoje jednostavnosti izabran je kao alat za kreiranje java desktop aplikacije za ovaj projekat.