Fakultet organizacionih nauka,

Univerzitet u Beogradu

**Projektni rad iz predmeta Baze podataka 2**

**Tema: Projektovanje IS nabavke privrednog društva Manet d.o.o.**

Sadržaj

[1.PMOV 3](#_Toc29734868)

[2.Relacioni model 8](#_Toc29734869)

[3.Denormalizacija 10](#_Toc29734870)

[4.Trigeri 16](#_Toc29734871)

[5.Korisnički definisani tipovi 21](#_Toc29734872)

[6. Indeksi 23](#_Toc29734873)

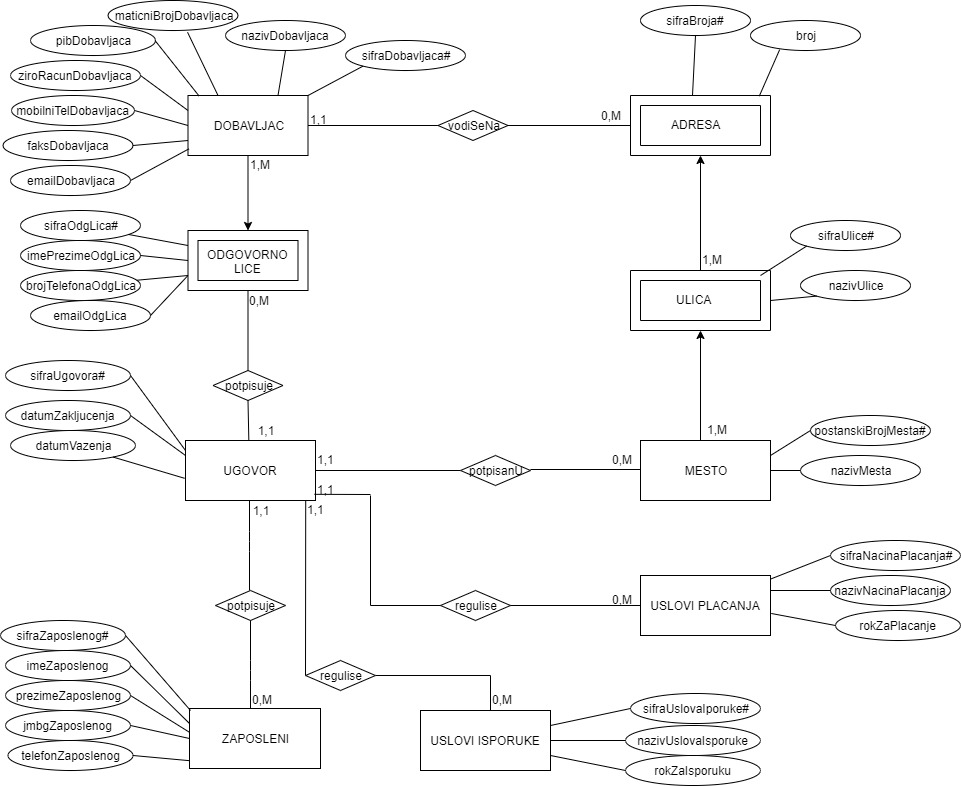
[7. Vertikalno particionisanje 25](#_Toc29734874)

[8. Horizontalno particionisanje 27](#_Toc29734875)

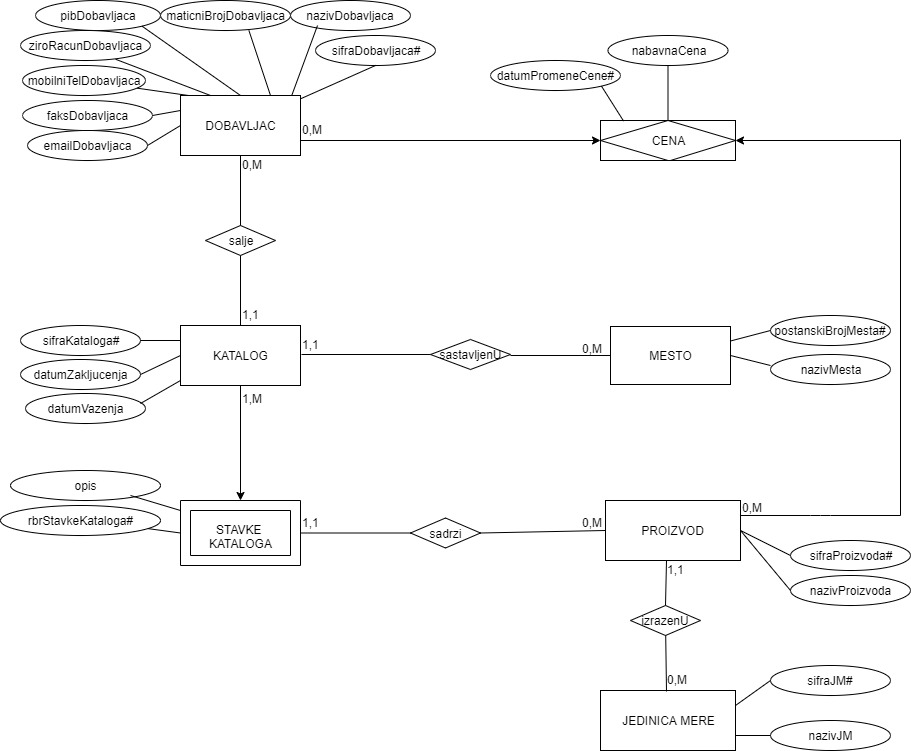
[9.Tehnike optimizacije sa pratećim uskladištenim procedurama 29](#_Toc29734876)

# 1.PMOV

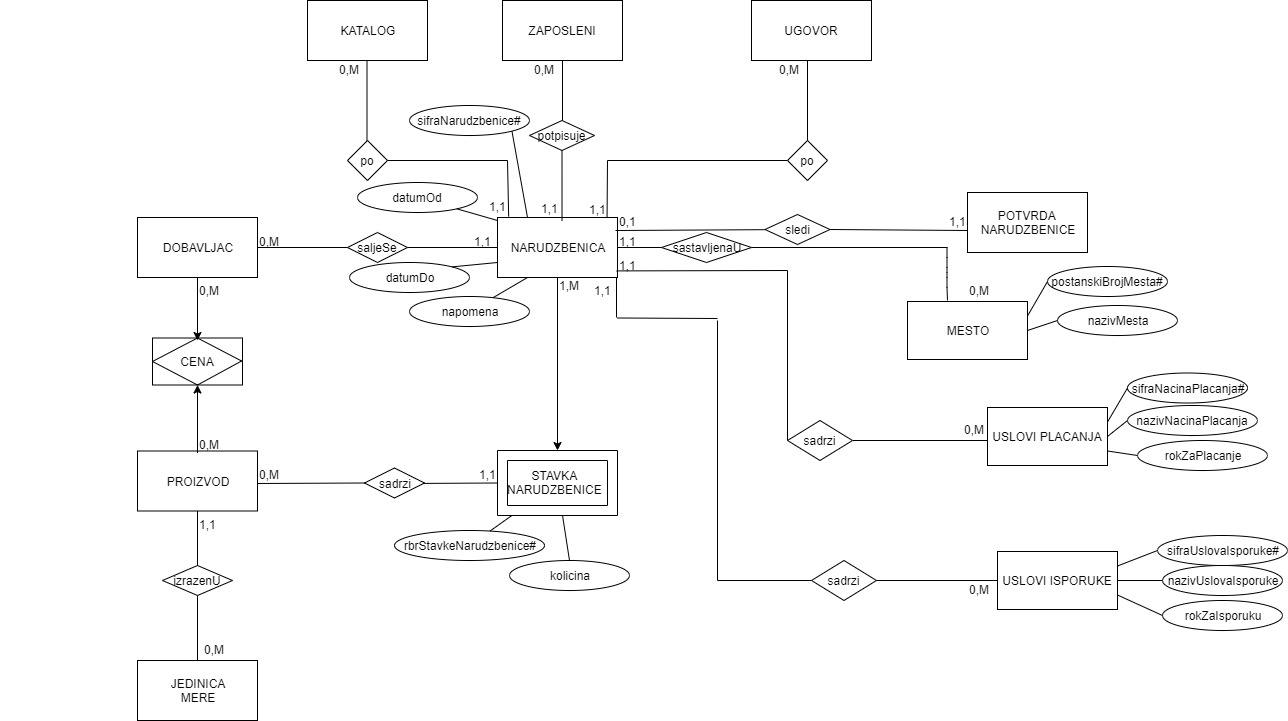
Ugovor



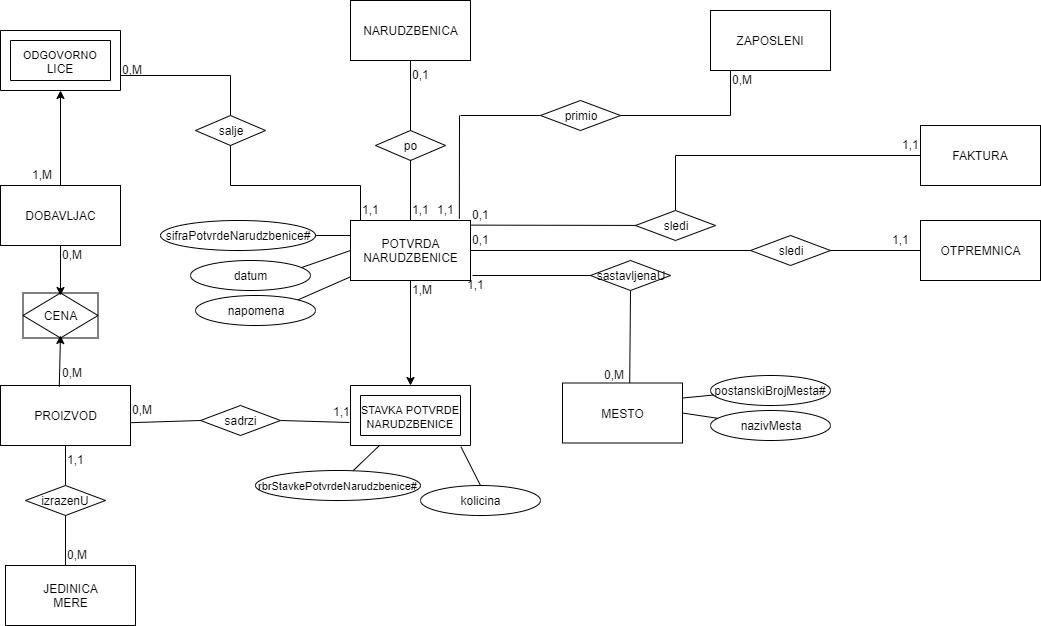
Katalog



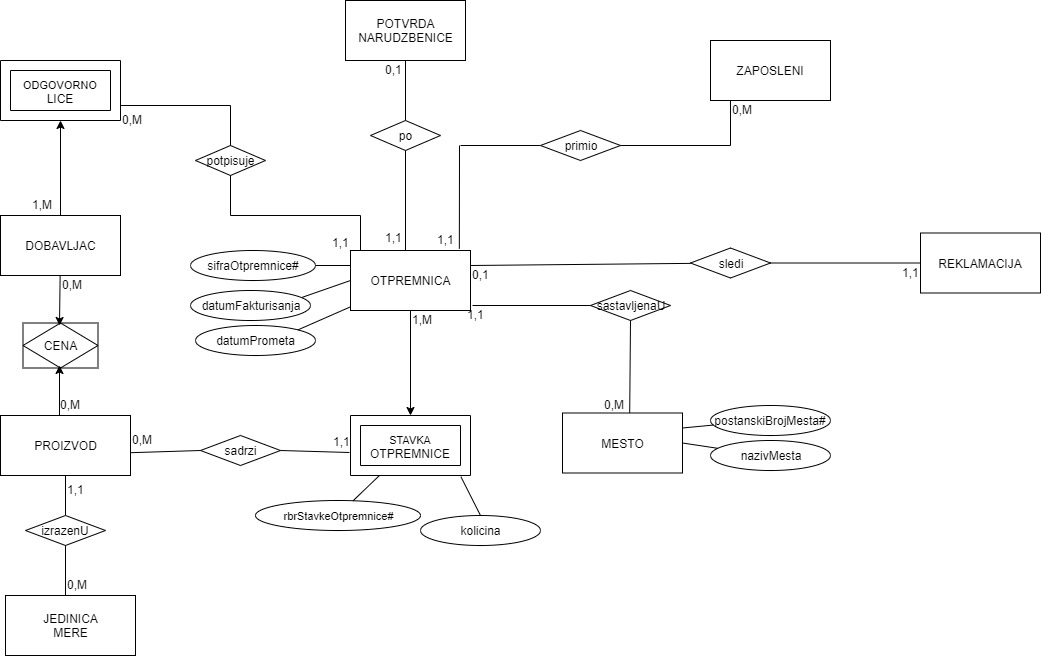
Narudžbenica



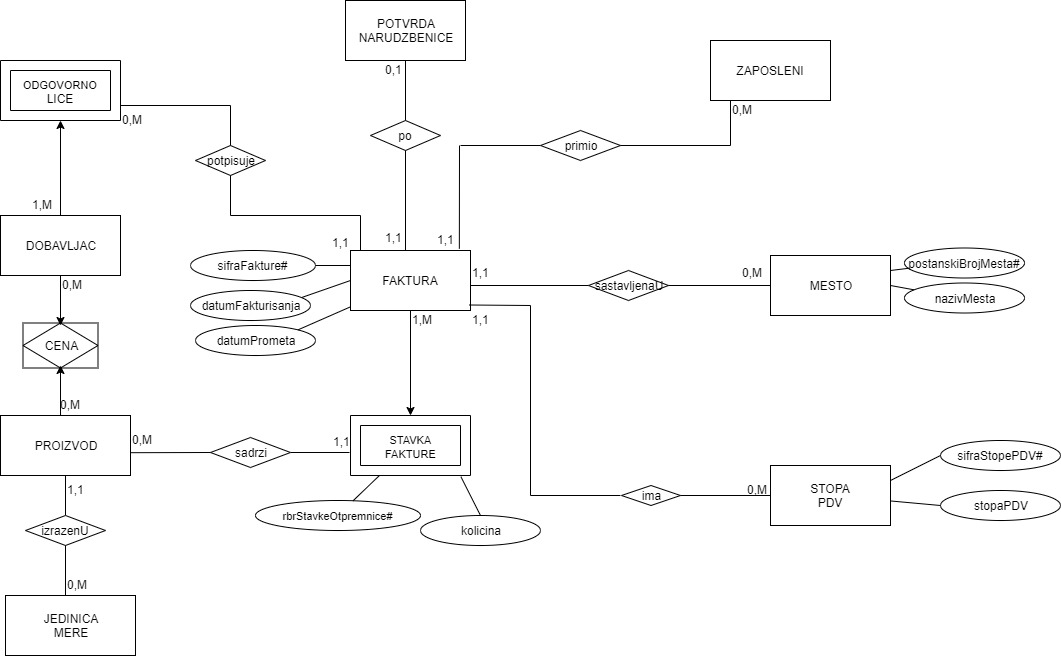
Potvrda narudzbenice



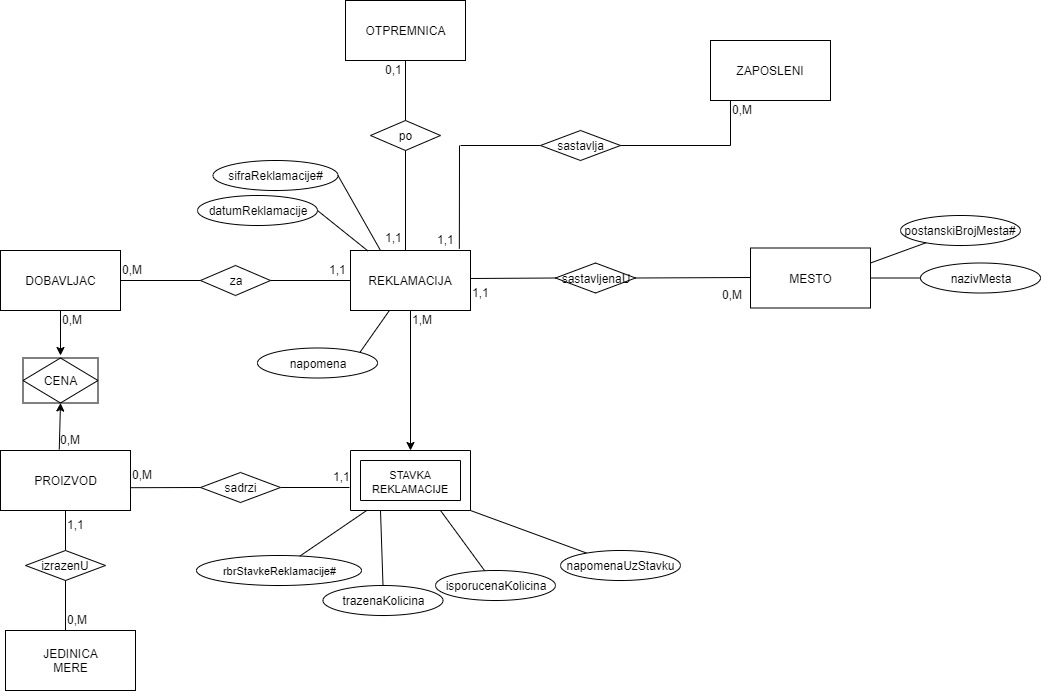
Otpremnica



Faktura



Reklamacija



# 2.Relacioni model

Mesto(postanskiBrojMesta, nazivMesta)

Ulica(sifraUlice, *postanskiBrojMesta*, nazivUlice)

Adresa(sifraBroja, *sifraUlice, postanskiBrojMesta*, broj)

Dobavljac(sifraDobavljaca, nazivDobavljaca, maticniBrojDobavljaca, pibDobavljaca, ziroRacunDobavljaca, mobilniTelDobavljaca, faksDobavljaca, emailDobavljaca, *sifraBroja*, *sifraUlice, postanskiBrojMesta*)

OdgovornoLice(sifraOdgLica, *sifraDobavljaca*, imePrezimeOdgLica, brojTelefonaOdgLica, emaliOdgLica)

Ugovor(sifraUgovora, datumZakljucenja, datumVazenja, *sifraOdgLica, sifraDobavljaca, sifraZaposlenog, postanskiBrojMesta, sifraUslovaPlacanja, sifraUslovaIsporuke*)

Zaposleni(sifraZaposlenog, imeZaposlenog, prezimeZaposlenog, jmbgZaposlenog, telefonZaposlenog)

UsloviPlacanja(sifraNacinaPlacanja, nazivNacinaPlacanja, rokZaPlacanje)

UslovIsporuke(sifraUslovaIsporuke, nazivUslovaIsporuke, rokZaIsporuku)

Katalog(sifraKataloga, datumZakljucenja, datumVazenja, *sifraDobavljaca, postanskiBrojMesta*)

StavkaKataloga(rbrStavkeKataloga, *sifraKataloga*, opis*, sifraProizvoda*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM)*

StopaPDV(sifraStopePDV, stopaPDV)

Cena(datumPromeneCene, *sifraDobavljaca, sifraProizvoda*, nabavnaCena)

JedinicaMere(sifraJM, nazivJM)

Narudzbenica(sifraNarudzbenice, datumOd, datumDo, napomena, *sifraDobavljaca, sifraKataloga, sifraZaposlenog, sifraUgovora, postanskiBrojMesta, sifraUslovaPlacanja, sifraUslovaIsporuke*)

ParitetIsporuke(sifraParitetaIsporuke, nazivParitetaIsporuke, napomena)

StavkaNarudzbenice(rbrStavkeNarudzbenice, *sifraNarudzbenice*, kolicina, *sifraProizvoda*)

PotvrdaNarudzbenice(sifraPotvrdeNarudzbenice, datum, napomena, *sifraOdgLica, sifraDobavljaca, sifraNarudzbenice, sifraZaposlenog, postanskiBrojMesta*)

StavkaPotvrdeNarudzbenice(rbrStavkePotvrdeNarudzbenice, *sifraPotvrdeNarudzbenice*, kolicina, *sifraProizvoda*)

Otpremnica(sifraOtpremnice, datumFakturisanja, datumPrometa, *sifraOdgLica, sifraDobavljaca, sifraZaposlenog, sifraPotvrdeNarudzbenice, postanskiBrojMesta*)

StavkaOtpremnice(rbrStavkeOtpremnice, *sifraOtpremnice*, kolicina, *sifraProizvoda*)

Faktura(sifraFakture, datumFakturisanja, datumPrometa, *sifraOdgLica, sifraDobavljaca, sifraZaposlenog, sifraPotvrdeNarudzbenice, postanskiBrojMesta, sifraStopePDV*)

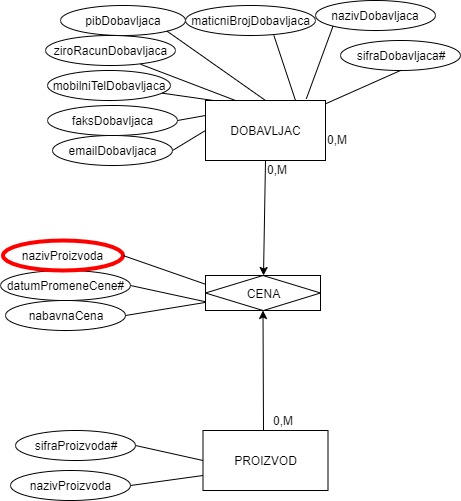
StavkaFakture(rbrStavkeFakture, *sifraFakture*, kolicina, *sifraProizvoda*)

Reklamacija(sifraReklamacije, datumReklamacije, *sifraOtpremnice, sifraDobavljaca, sifraZaposlenog, postanskiBrojMesta*)

StavkaReklamacije(rbrStavkeReklamacije, *sifraReklamacije*, trazenaKolicina, isporucenaKolicina, napomenaUzStavku, *sifraProizvoda*)

# 3.Denormalizacija

**Denormalizacija 2NF Pre joining**



Relacioni model pre izvršene denormalizacije:

Dobavljac(sifraDobavljaca, nazivDobavljaca, maticniBrojDobavljaca, pibDobavljaca, ziroRacunDobavljaca, mobilniTelDobavljaca, faksDobavljaca, emailDobavljaca, *sifraBroja*, *sifraUlice, postanskiBrojMesta*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM*)

Cena(datumPromeneCene, *sifraDobavljaca, sifraProizvoda*, nabavnaCena)

Relacioni model nakon izvršene denormalizacije:

Dobavljac(sifraDobavljaca, nazivDobavljaca, maticniBrojDobavljaca, pibDobavljaca, ziroRacunDobavljaca, mobilniTelDobavljaca, faksDobavljaca, emailDobavljaca, *sifraBroja*, *sifraUlice, postanskiBrojMesta*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM*)

Cena(datumPromeneCene, *sifraDobavljaca, sifraProizvoda*, nabavnaCena, **nazivProizvoda**)

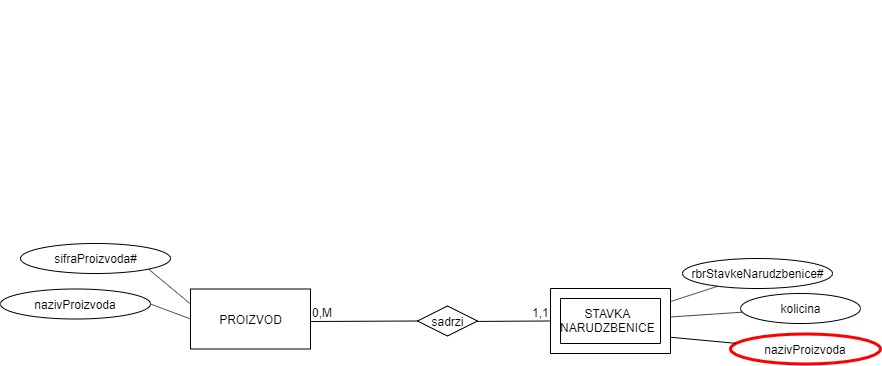
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tip trigera | Kolona | Potreban | Funkcija |
| Proizvod | insert |  | NE |  |
| update | nazivProizvoda | DA | Prilikom izmene vrednosti kolone nazivProizvoda u tabeli Proizvod pokreće se triger za ažuriranje vrednosti kolone nazivProizvoda u tabeli Cena |
| delete |  | NE |  |
| Cena | insert |  | DA | Kada se unese vrednost u kolonu sifraProizvoda automatski se popunjava kolona NazivProizvoda |
| update | nazivProizvoda | DA | Sprečava direktnu izmenu kolone nazivProizvoda |
| delete |  | NE |  |

**Funkcionalne zavisnosti:**

datumPromeneCene, sifraDobavljaca, sifraProizvoda -> nazivProizvoda

sifraProizvoda -> nazivProizvoda

**Denormalizacija 3NF Pre joining**



Relacioni model pre izvršene denormalizacije:

StavkaNarudzbenice(rbrStavkeNarudzbenice, *sifraNarudzbenice*, kolicina, *sifraProizvoda*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM*)

Relacioni model nakon izvršene denormalizacije:

StavkaNarudzbenice(rbrStavkeNarudzbenice, *sifraNarudzbenice*, kolicina, *sifraProizvoda,* **nazivProizvoda**)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM)*

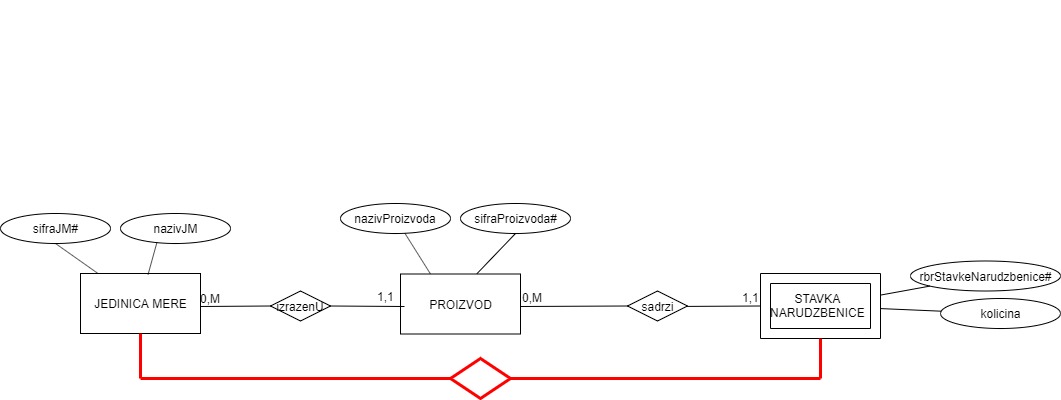
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tip trigera | Kolona | Potreban | Funkcija |
| Stavka narudzbenice | insert |  | DA | Na osnovu unete vrednosti u koloni sifraProizvoda unosi se vrednost u kolonu nazivProizvoda |
| update | nazivProizvoda | DA | Nije dozvoljeno ažuriranje ove kolone |
| sifraProizvoda | DA | Ažuriranjem vrednosti u koloni sifraProizvoda ažurira se i vrednost kolone nazivProizvoda |
| delete |  | NE |  |
| Proizvod | insert |  | NE |  |
| update | nazivProizvoda | DA | Kada se izmeni kolona nazivProizvoda ažurira se kolona nazivProizvoda u tabeli Stavka narudzbenice |
| delete |  | NE |  |
|  | | | | |

**Funkcionalne zavisnosti:**

rbrStavkeNarudzbenice, sifraNarudzbenice -> kolicina,sifraProizvoda, nazivProizvoda

sifraProizvoda -> nazivProizvoda

**Short - circuit tehnika 3NF**



Relacioni model pre izvršene denormalizacije:

StavkaNarudzbenice(rbrStavkeNarudzbenice, *sifraNarudzbenice*, kolicina, *sifraProizvoda*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM*)

JedinicaMere(sifraJM, nazivJM)

Relacioni model nakon izvršene denormalizacije:

StavkaNarudzbenice(rbrStavkeNarudzbenice, *sifraNarudzbenice*, kolicina, *sifraProizvoda, sifraJM*)

Proizvod(sifraProizvoda, nazivProizvoda, *sifraJM*)

JedinicaMere(sifraJM, nazivJM)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tip trigera | Kolona | Potreban | Funkcija |
| Proizvod | insert |  |  |  |
| update | sifraJM | DA | Ukoliko Proizvod postoji u nekoj Stavki Narudzbenice triger treba da omoguci azuriranje kolone sifraJM u tabeli Stavka Narudzbenice za sve stavke u kojima se nalazi taj proizvod |
| delete |  | NE |  |
| Stavka narudzbenice | insert |  | DA | Upisuje odgovarajucu vrednost u kolonu sifraJM na osnovu unite vrednosti za kolonu sifraProizvoda |
| update | sifraJM | DA | Sprecava direktnu izmenu |
| sifraProizvoda | DA | U kolonu sifraJM upisuje odgovarajucu vrednost na osnovu nove vrednosti za sifraProizvoda |
| delete |  | NE |  |

**Funkcionalne zavisnosti:**

rbrStavkeNarudzbenice, sifraNarudzbenice -> kolicina, sifraProizvoda, sifraJM

sifraProizvoda -> sifraJM

# 4.Trigeri

Triger za 2NF Pre - joining tehniku

*Tabela: Cena*

**Operacija: INSERT**

**Opis: Kada se unese vrednost u kolonu sifraProizvoda automatski se popunjava kolona nazivProizvoda**

create or replace trigger insert\_proizvod\_cena

before insert on cena

for each row

declare new\_nazivproizvoda varchar2(30);

begin

select nazivproizvoda into new\_nazivproizvoda

from proizvod

where proizvod.sifraproizvoda = :new.sifraproizvoda;

:new.nazivproizvoda := new\_nazivproizvoda;

end;

**Operacija: UPDATE**

**Opis: Sprečava se direktna izmena kolone sifraProizvoda**

create or replace TRIGGER update\_sifraproizvoda\_cena

before UPDATE OF sifraproizvoda ON cena

for each row

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

new\_sifraproizvoda NUMBER(20,0);

new\_sifradob NUMBER;

new\_datum date;

new\_cena decimal;

new\_naziv VARCHAR2(50);

BEGIN

new\_datum:=:new.datumpromenecene;

new\_sifradob:=:new.sifradobavljaca;

new\_cena:=:new.nabavnacena;

new\_naziv:=:new.nazivproizvoda;

select sifraproizvoda into new\_sifraproizvoda

from cena

where datumpromenecene = new\_datum and sifradobavljaca = new\_sifradob and nabavnacena = new\_cena and nazivproizvoda = new\_naziv;

:new.sifraproizvoda:= new\_sifraproizvoda;

commit;

END;

**Operacija: UPDATE**

**Opis: Sprečava se direktna izmena kolona nazivProizvoda**

create or replace TRIGGER update\_nazivproizvoda\_cena

before UPDATE OF nazivproizvoda ON cena

for each row

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

sifrapro NUMBER(20,0);

starinazivproizvoda varchar2(30);

BEGIN

sifrapro:=:new.sifraproizvoda;

select nazivproizvoda

into starinazivproizvoda

from proizvod

where sifraproizvoda=sifrapro;

:new.nazivproizvoda:= starinazivproizvoda;

commit;

END;

*Tabela: Proizvod*

**Operacija: UPDATE**

**Opis: Prilikom izmene vrednosti kolone nazivProizvoda u tabeli Proizvod pokreće se triger za ažuriranje vrednosti kolone nazivProizvoda u tabeli Cena**

create or replace TRIGGER

update\_nazivproizvoda\_proizvod AFTER UPDATE OF nazivproizvoda ON proizvod

FOR EACH ROW

DECLARE

PRAGMA AUTONOMOUS\_TRANSACTION;

new\_nazivprozivoda VARCHAR2(30);

sifrapro number(20,0);

BEGIN

execute immediate 'alter trigger update\_nazivproizvoda\_cena disable';

new\_nazivprozivoda:=:new.nazivproizvoda;

sifrapro:=:new.sifraproizvoda;

update cena

set nazivproizvoda=new\_nazivprozivoda

where sifraproizvoda=sifrapro;

commit;

begin

execute immediate 'alter trigger update\_nazivproizvoda\_cena enable';

END;

END;

Triger za Short-circuit tehniku 3NF

*Tabela: Stavka narudzbenice*

**Operacija: INSERT**

**Opis: Upisuje odgovarajuću vrednost u kolonu sifraJM na osnovu unete vrednosti za kolonu sifraProizvoda**

create or replace TRIGGER insert\_sifrajm\_in\_stavkanarudzbenice

BEFORE INSERT ON stavkanarudzbenice

FOR EACH ROW

declare new\_sifrajm number(20,0);

BEGIN

select sifrajm into new\_sifrajm

from proizvod

where sifraproizvoda=:new.sifraproizvoda;

:new.sifrajm:=new\_sifrajm;

END;

**Operacija: UPDATE**

**Opis: Sprecava direktnu izmenu kolone sifraJM u tabeli StavkaNarudzbenice**

create or replace TRIGGER stop\_update\_sifrajm\_stavkanarudzbenice

before UPDATE OF sifrajm ON stavkanarudzbenice

for each row

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

new\_sifraproizvoda NUMBER(20,0);

new\_sifrajm NUMBER(20,0);

BEGIN

new\_sifraproizvoda:=:new.sifraproizvoda;

select sifrajm into new\_sifrajm

from proizvod

where sifraproizvoda=new\_sifraproizvoda;

:new.sifrajm:= new\_sifrajm;

commit;

END;

**Operacija: UPDATE**

**Opis: U kolonu sifraJM upisuje odgovarajuću vrednost na osnovu nove vrednosti za sifraProizvoda**

create or replace TRIGGER update\_sifraproizvoda\_stavkanarudzbenice

BEFORE UPDATE OF sifraproizvoda ON stavkanarudzbenice

FOR EACH ROW

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

new\_nazivproizvoda VARCHAR2(20);

new\_sifrajm number(20,0);

BEGIN

select sifrajm into new\_sifrajm

from proizvod

where sifraproizvoda=:new.sifraproizvoda;

:new.sifrajm:=new\_sifrajm;

commit;

END;

*Tabela: Proizvod*

**Operacija: UPDATE**

**Opis: Ukoliko Proizvod postoji u nekoj Stavki narudzbenice omogućava ažuriranje kolone sifraJM u tabeli Stavka narudzbenice**

create or replace TRIGGER update\_sifrajm\_proizvod

after UPDATE OF sifrajm ON proizvod

FOR EACH ROW

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

new\_sifrajm NUMBER(20,0);

BEGIN

execute immediate 'alter trigger stop\_update\_sifrajm\_stavkanarudzbenice disable';

execute immediate 'alter trigger insert\_sifrajm\_in\_stavkanarudzbenice disable';

execute immediate 'alter trigger update\_sifrajm\_stavkanarudzbenice disable';

new\_sifrajm:=:new.sifrajm;

update stavkanarudzbenice

set sifrajm=new\_sifrajm

where sifraproizvoda=:new.sifraproizvoda;

commit;

begin

execute immediate 'alter trigger insert\_sifrajm\_in\_stavkanarudzbenice enable';

execute immediate 'alter trigger stop\_update\_sifrajm\_stavkanarudzbenice enable';

execute immediate 'alter trigger update\_sifrajm\_stavkanarudzbenice enable';

END;

END;

# 5.Korisnički definisani tipovi

**Distinct tip**

create or replace TYPE procenat AS object

(

tip NUMBER (13),

CONSTRUCTOR

FUNCTION procenat ( tip NUMBER )

RETURN SELF AS RESULT,

member function dajbroj return number

) instantiable FINAL;

**Body:**

create or replace TYPE BODY procenat AS

CONSTRUCTOR FUNCTION procenat

( tip NUMBER )

RETURN SELF AS RESULT AS tip\_beyond\_range EXCEPTION;

PRAGMA EXCEPTION\_INIT(tip\_beyond\_range, -20101);

BEGIN

IF tip BETWEEN 0

AND 100 THEN

self.tip := tip;

RETURN;

ELSE

raise\_application\_error(-20101, 'ERROR : Vrednost treba da bude izmedju 0 i 100');

END IF;

END ;

member function dajbroj return number as

begin

return self.tip;

end;

END;

create table stopapdv(sifrastopepdv number, procenattip procenat, sifraproizvoda number)

insert into stopapdv values (1, procenat(10));

**Strukturiran tip**

create or replace TYPE USLOVIPLACANJA AS OBJECT

(

nazivuslova VARCHAR2(30),

rok NUMBER (13),

CONSTRUCTOR

FUNCTION usloviplacanja ( rok NUMBER, nazivuslova VARCHAR2 )

RETURN SELF AS RESULT,

member function vratirok return number,

member function vratiFormiranString return VARCHAR2,

member function vratiNaziv return VARCHAR2

)INSTANTIABLE not final

*Body:*

create or replace TYPE BODY usloviplacanja AS

CONSTRUCTOR FUNCTION usloviplacanja

( rok NUMBER, nazivuslova VARCHAR2 )

RETURN SELF AS RESULT AS tip\_beyond\_range EXCEPTION;

PRAGMA EXCEPTION\_INIT(tip\_beyond\_range, -20101);

BEGIN

IF rok BETWEEN 1

AND 30 THEN

self.rok := rok;

self.nazivuslova := nazivuslova;

RETURN;

ELSE

raise\_application\_error(-20101, 'ERROR : Po ugovoru rok za placanje ne sme biti duzi od 30 dana');

END IF;

END ;

member function vratirok return number as

begin

return self.rok;

end;

member function vratiFormiranString return VARCHAR2 as

begin

return CONCAT(CONCAT('Placanje za ', to\_char(self.rok)), 'r dana.') ;

end;

member function vratiNaziv return VARCHAR2 as

begin

return self.nazivuslova;

end;

END;

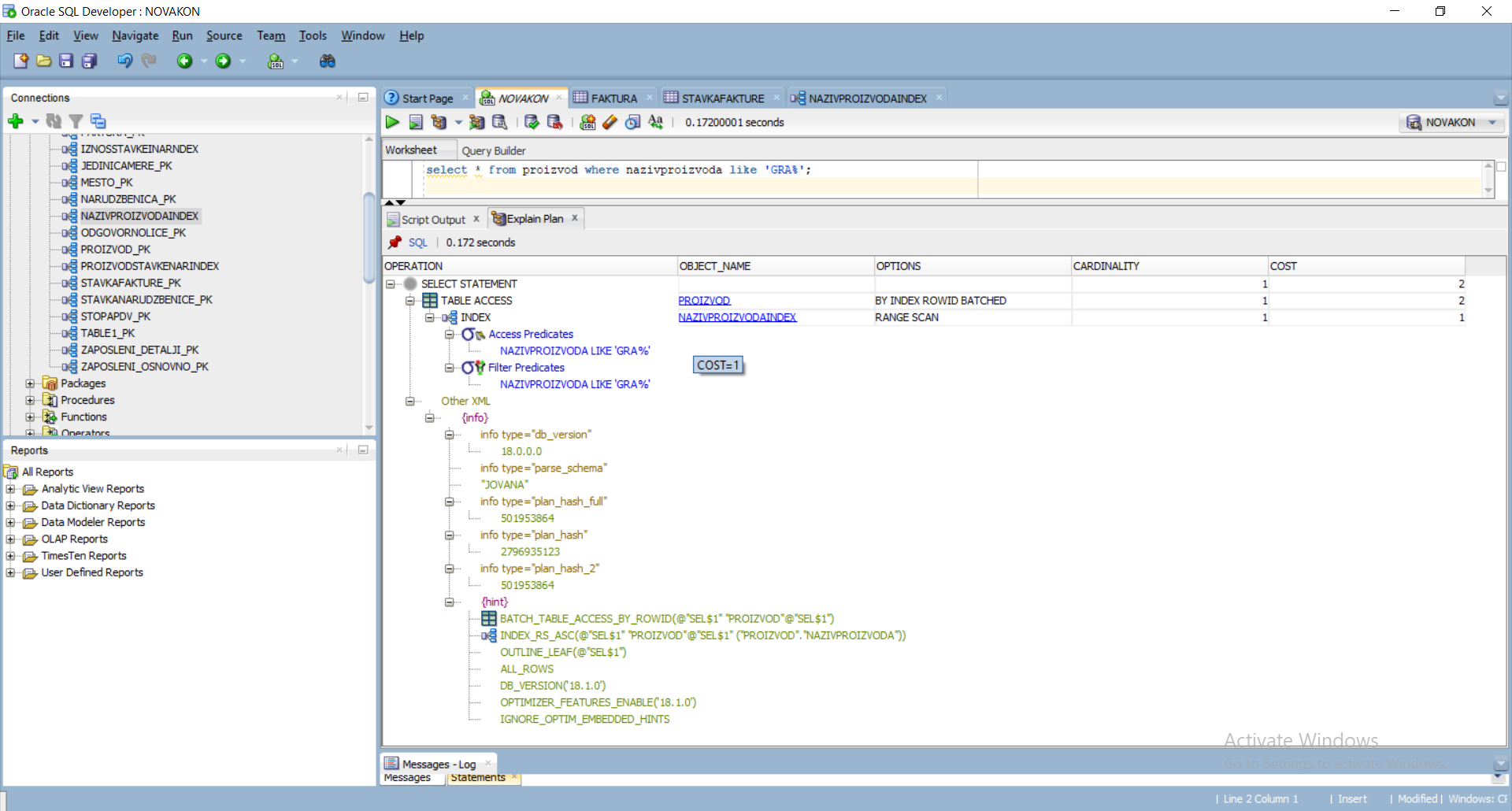
create table uslovi\_placanja(sifrauslova number, usloviplacanja\_tip usloviplacanja);

insert into uslovi\_placanja values(1,usloviplacanja(30,'gotovinsko'));

# 6. Indeksi

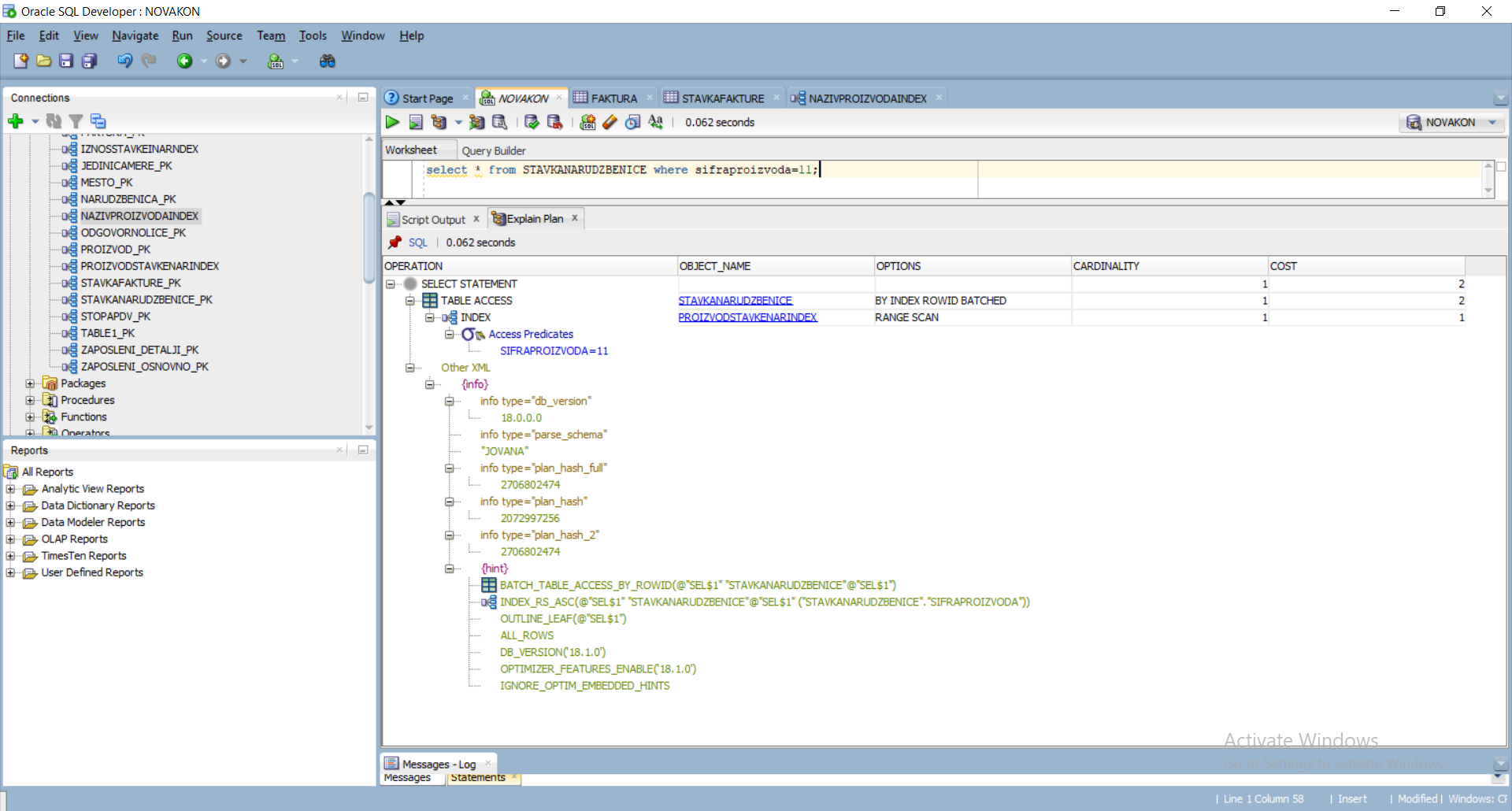
**Indeks nad tekstualnim poljem:**

CREATE INDEX NAZIVPROIZVODAINDEX ON PROIZVOD (NAZIVPROIZVODA);



**Indeks nad spoljnim kljucem:**

CREATE INDEX PROIZVODSTAVKENARINDEX ON STAVKANARUDZBENICE (SIFRAPROIZVODA)



# 7. Vertikalno particionisanje

Izvršeno je vertikalno particionisanje tabele Zaposleni

Zaposleni(sifraZaposlenog, imeZaposlenog, prezimeZaposlenog, jmbgZaposlenog, telefonZaposlenog)

Tabelu Zaposleni delimo na dva dela ZaposleniOsnovno i ZaposleniDetalji.

ZaposleniOsnovno(sifraZaposlenog, imeZaposlenog, prezimeZaposlenog)

CREATE TABLE ZAPOSLENI\_OSNOVNO(

SIFRAZAPOSLENOG NUMBER NOT NULL ENABLE,

IMEZAPOSLENOG VARCHAR2(50),

PREZIMEZAPOSLENOG VARCHAR(50),

CONSTRAINT ZAPOSLENI\_OSNOVNO\_PK PRIMARY KEY (SIFRAZAPOSLENOG) ENABLE

);

ZaposleniDetalji(sifraZaposlenog, jmbgZaposlenog, telefonZaposlenog)

CREATE TABLE ZAPOSLENI\_DETALJI(

SIFRAZAPOSLENOG NUMBER NOT NULL ENABLE,

JMBGZAPOSLENOG NUMBER,

TELEFONZAPOSLENOG VARCHAR2(50)

CONSTRAINT ZAPOSLENI\_DETALJI\_PK PRIMARY KEY (SIFRAZAPOSLENOG) ENABLE

)

View koji sadrži kolone obe tabele:

CREATE OR REPLACE VIEW ZAPOSLENI\_VIEW

AS

SELECT ZO.SIFRAZAPOSLENOG, ZO.IMEZAPOSLENOG, ZO.PREZIMEZAPOSLENOG, ZD.JMBGZAPOSLENOG, ZD.TELEFONZAPOSLENOG

FROM ZAPOSLENI\_OSNOVNO ZO, ZAPOSLENI\_DETALJI ZD

WHERE ZO.SIFRAZAPOSLENOG = ZD.SIFRAZAPOSLENOG;

Pregled sadržaja pogleda:

SELECT \* FROM ZAPOSLENI\_VIEW;

*Instead of triger za unos novog zaposlenog:*

CREATE OR REPLACE TRIGGER ZAPOSLENIINSERT

INSTEAD OF INSERT ON ZAPOSLENI\_VIEW

FOR EACH ROW

BEGIN

INSERT INTO ZAPOSLENI\_OSNOVNO(SIFRAZAPOSLENOG,IMEZAPOSLENOG,PREZIMEZAPOSLENOG)

VALUES

(:NEW.SIFRAZAPOSLENOG,:NEW.IMEZAPOSLENOG,:NEW.PREZIMEZAPOSLENOG);

INSERT INTO ZAPOSLENI\_DETALJI (SIFRAZAPOSLENOG,JMBGZAPOSLENOG, TELEFONZAPOSLENOG)

VALUES

(:NEW.SIFRAZAPOSLENOG,:NEW.JMBGZAPOSLENOG,:NEW.TELEFONZAPOSLENOG);

END;

*INSERT INTO ZAPOSLENI\_VIEW VALUES (1,'Jovana', 'Milic', 03079956247, '0564322121');*

*Instead of triger za ažuriranje zaposlenog:*

CREATE OR REPLACE TRIGGER ZAPOSLENIUPDATE

INSTEAD OF UPDATE ON ZAPOSLENI\_VIEW

FOR EACH ROW

BEGIN

UPDATE ZAPOSLENI\_OSNOVNO

SET SIFRAZAPOSLENOG= :NEW.SIFRAZAPOSLENOG, IMEZAPOSLENOG= :NEW.IMEZAPOSLENOG, PREZIMEZAPOSLENOG= :NEW.PREZIMEZAPOSLENOG

WHERE ZAPOSLENI\_OSNOVNO.SIFRAZAPOSLENOG= :NEW.SIFRAZAPOSLENOG;

UPDATE ZAPOSLENI\_DETALJI

SET SIFRAZAPOSLENOG= :NEW.SIFRAZAPOSLENOG,

JMBGZAPOSLENOG= :NEW.JMBGZAPOSLENOG,

TELEFONZAPOSLENOG= :NEW.TELEFONZAPOSLENOG

WHERE ZAPOSLENI\_DETALJI.SIFRAZAPOSLENOG= :NEW.SIFRAZAPOSLENOG;

END;

*UPDATE ZAPOSLENI\_OSNOVNO SET IMEZAPOSLENOG='ALeksandra'*

*WHERE SIFRAzaposlenog=1;*

*Instead od triger za birsanje zaposlenog:*

CREATE OR REPLACE TRIGGER ZAPOSLENIDELETE

INSTEAD OF DELETE ON ZAPOSLENI\_VIEW

FOR EACH ROW

BEGIN

DELETE FROM ZAPOSLENI\_OSNOVNO WHERE ZAPOSLENI\_OSNOVNO.SIFRAZAPOSLENOG = :OLD.SIFRAZAPOSLENOG;

DELETE FROM ZAPOSLENI\_DETALJI WHERE ZAPOSLENI\_DETALJI.SIFRAZAPOSLENOG= :OLD.SIFRAZAPOSLENOG;

END;

*DELETE FROM ZAPOSLENI\_VIEW WHERE SIFRAZAPOSLENOG=2;*

# 8. Horizontalno particionisanje

Kod horizontalnog particionisanja biće izvršeno grupisanje tabele Cena na osnovu vrednosti nabavne cene.

Cena(datumPromeneCene, *sifraDobavljaca, sifraProizvoda*, nabavnaCena)

CREATE TABLE CENA

(

SIFRADOBAVLJACA NUMBER NOT NULL

, SIFRAPROIZVODA NUMBER NOT NULL

, DATUMPROMENECENE DATE NOT NULL

, NABAVNACENA NUMBER(\*, 0)

, NAZIVPROIZVODA VARCHAR2(20 BYTE)

, CONSTRAINT TABLE1\_PK PRIMARY KEY

(

SIFRADOBAVLJACA

, SIFRAPROIZVODA

, DATUMPROMENECENE

)

ADD CONSTRAINT CENA\_FK1 FOREIGN KEY

(

SIFRADOBAVLJACA

)

REFERENCES DOBAVLJAC

(

SIFRADOBAVLJACA

)

ENABLE;

ALTER TABLE CENA

ADD CONSTRAINT CENA\_FK2 FOREIGN KEY

( SIFRAPROIZVODA )REFERENCES PROIZVOD

(SIFRAPROIZVODA )ENABLE;

ALTER TABLE CENA MODIFY

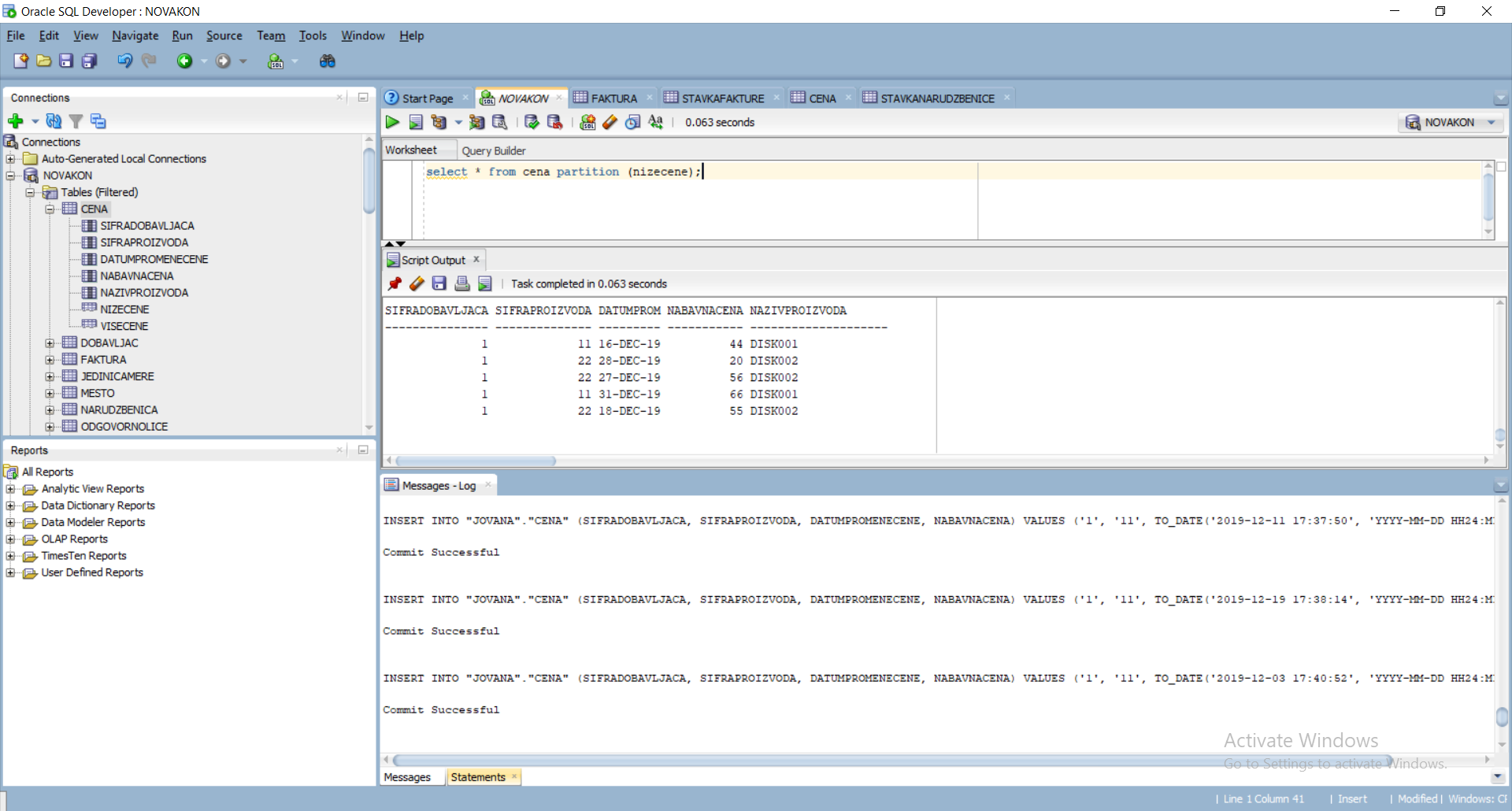
PARTITION BY RANGE (nabavnacena) INTERVAL (100)

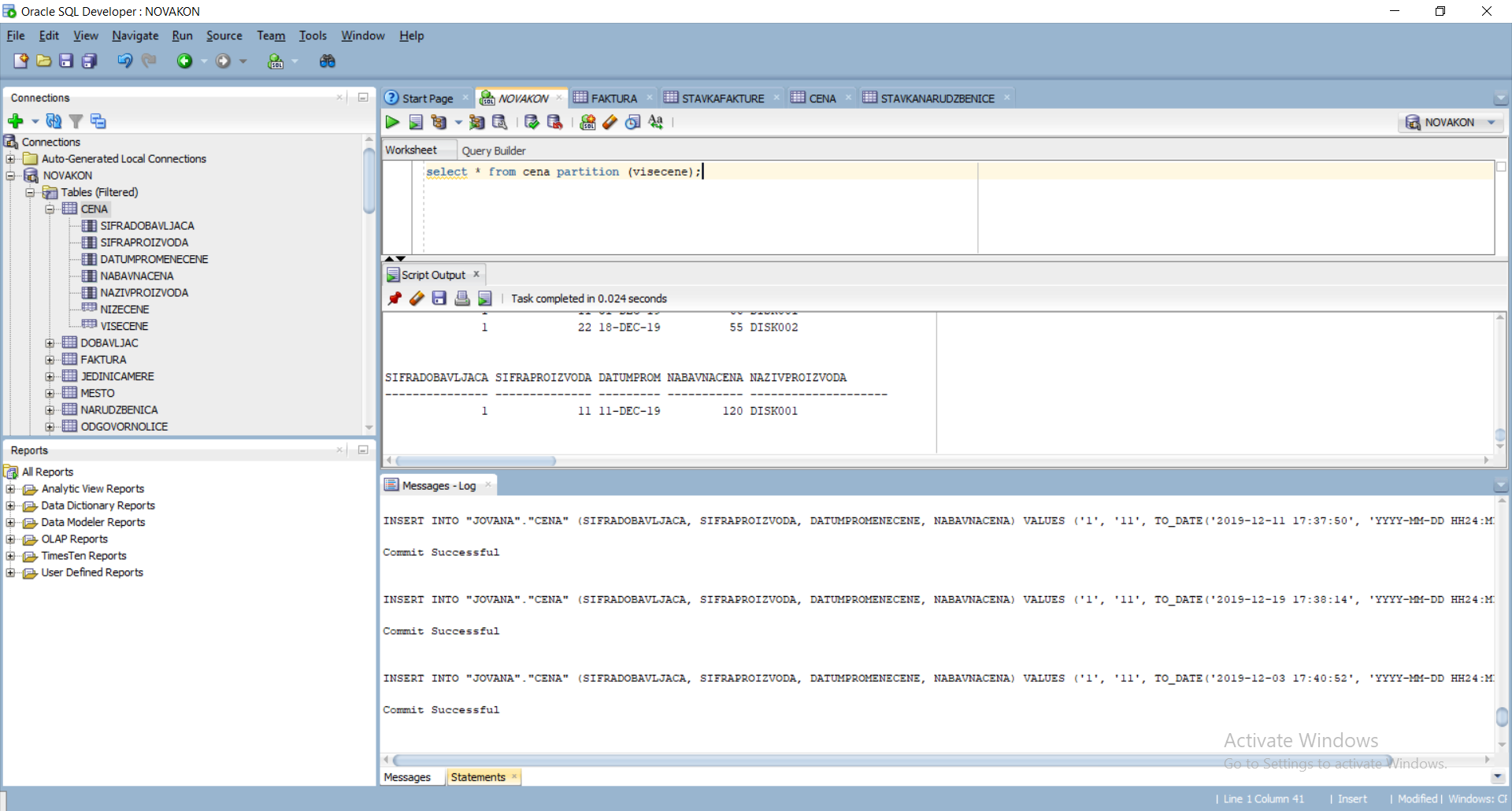
(PARTITION nizecene VALUES LESS THAN (100),

PARTITION visecene VALUES LESS THAN (200)

)ONLINE;

Pomoću select upita nad tabelom Cena možemo uvideti kako je izvršeno particionisanje.

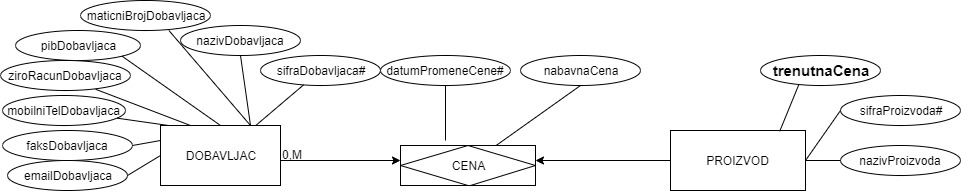




# 9.Tehnike optimizacije sa pratećim uskladištenim procedurama

**Repeating Single Detail with Master tehnika optimizacije**

Redundantan atribut **trenutnaCena** dodaje se tabeli Proizvod što je i prikazano na konceptualnom i relacionom modelu.



Cena*(sifraDobavljaca, sifraProizvoda*, datumPromeneCene, nabavnaCena)

Proizvod(sifraProizvoda, nazivProizvoda**, trenutnaCena**)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tip trigera | Kolona | Potreban | Šta treba da uradi? |
| Proizvod | insert |  | NE |  |
| update | trenutnaCena | DA | Zabrana ažuriranja polja trenutnaCena u tabeli Proizvod. |
| delete |  | NE |  |
| Cena | insert |  | DA | Prilikom unosa cene okida se triger koji poziva proceduru za ažuriranje polja trenutnaCena u tabeli Proizvod. |
| update | nabavnaCena | DA | Prilikom ažuriranja nabavne cene okida se triger koji poziva proceduru za ažuriranje polja trenutnaCena u tabeli Proizvod. |
| delete |  | DA | Prilikom brisanja cene okida se triger koji poziva proceduru za ažuriranje polja trenutnaCena u tabeli Proizvod. |

Package:

create or replace PACKAGE CENAPAKET AS

SIFRAPROIZVODA number;

END CENAPAKET;

Trigeri

create or replace TRIGGER PROIZVOD\_TRENUTNACENA

after INSERT OR UPDATE OF NABAVNACENA OR DELETE ON CENA

FOR EACH ROW

BEGIN

IF (INSERTING OR UPDATING)

THEN

BEGIN CENAPAKET.sifraproizvoda := :NEW.sifraproizvoda; END;

ELSE

BEGIN CENAPAKET.sifraproizvoda := :OLD.sifraproizvoda; END;

END IF;

END;

create or replace TRIGGER proizvod\_trenutnacena\_pozivprocedure

after INSERT OR UPDATE OF NABAVNACENA OR DELETE ON CENA

DECLARE

s NUMBER:=CENAPAKET.sifraproizvoda;

BEGIN

DISSABLE\_TRIGGER('PROIZVOD\_TRENUTNACENA\_STOP\_UPDATE',0);

TRENUTNACENA(s);

DISSABLE\_TRIGGER('PROIZVOD\_TRENUTNACENA\_STOP\_UPDATE',1);

END;

Procedura

create or replace PROCEDURE TRENUTNACENA(sifrapro IN NUMBER)

AS

trenCena PROIZVOD.TRENUTNACENA%type; brojac number := 0;

BEGIN

trenCena:=0;

select count(sifrapro) into brojac

from cena

where sifraproizvoda = sifrapro;

if(brojac >= 1)

then

SELECT nabavnacena INTO trenCena

FROM CENA

WHERE sifraproizvoda=sifrapro and datumpromenecene=

(select max(datumpromenecene)

from CENA

where sifraproizvoda=sifrapro and

datumpromenecene<=sysdate);

UPDATE PROIZVOD

SET TRENUTNACENA = trenCena

WHERE sifraproizvoda=sifrapro;

else

update proizvod

set trenutnacena = 0

where sifraproizvoda = sifrapro;

end if;

END;

Procedura za pokretanje i zaustavljanje trigera

create or replace PROCEDURE DISSABLE\_TRIGGER (imeTrigera varchar2, flag number) IS

PRAGMA AUTONOMOUS\_TRANSACTION;

BEGIN

if flag = 0 then begin

EXECUTE IMMEDIATE 'ALTER TRIGGER '||imeTrigera||' DISABLE';

end;

end if;

if flag = 1 then begin

EXECUTE IMMEDIATE 'ALTER TRIGGER '||imeTrigera||' ENABLE';

end;

end if;

commit;

END;

Triger koji onemogućava direktnu izmenu polja trenutnaCena u okviru tabele Proizvod

create or replace TRIGGER proizvod\_trenutnacena\_stop\_update

before UPDATE OF trenutnacena ON proizvod

for each row

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

novacena NUMBER;

staracena NUMBER;

sifrapro NUMBER;

BEGIN

novacena:=:new.trenutnacena;

sifrapro := :new.sifraproizvoda;

select trenutnacena into staracena

from proizvod

where sifraproizvoda=sifrapro;

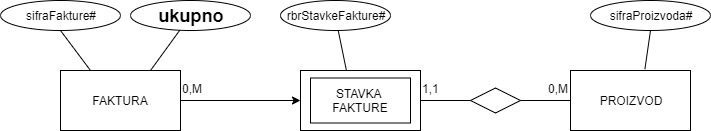
:new.trenutnacena:= staracena;

commit;

END;

**Storing Derivable Values tehnika optimizacije**

U relaciji Faktura dodaje se atribut ukupno u kojem će se čuvati ukupna vrednost fakture.



Faktura(sifraFakture, datumFakturisanja, datumPrometa, *sifraStopePdv*, *sifraZaposlenog, sifraDobavljaca, sifraOdgLica,* **Ukupno**)

StavkaFakture(rbrStavkeFakture, sifraFakture, kolicina, *sifraProizvoda*)

Proizvod(sifraProizvoda, nazivProizvoda)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tip trigera | Kolona | Potreban | Šta treba da uradi? |
| Faktura | insert |  | NE |  |
| update | Ukupno | DA | Zabrana ažuriranja polja Ukupno u tabeli Faktura. |
| delete |  | NE |  |
| Stavka Fakture | insert |  | DA | Prilikom unosa nove stavke fakture okida se triger koji poziva proceduru za ažuriranje polja Ukupno u tabeli Faktura. |
| update | kolicina | DA | Prilikom ažuriranja stavke fakture okida se triger koji poziva proceduru za ažuriranje polja Ukupno u tabeli Faktura. |
| delete |  | DA | Prilikom brisanja stavke fakture okida se triger koji poziva proceduru za ažuriranje polja Ukupno u tabeli Faktura. |

Package

create or replace PACKAGE FAKTURAPAKET AS

SIFRAFAKTURE number;

END FAKTURAPAKET;

Trigeri

create or replace TRIGGER FAKTURA\_UKUPNO

BEFORE INSERT OR UPDATE OR DELETE ON STAVKAFAKTURE

FOR EACH ROW

BEGIN

IF (INSERTING OR UPDATING)

THEN

BEGIN FAKTURAPAKET.SIFRAFAKTURE:=:NEW.SIFRAFAKTURE;END;

ELSE

BEGIN FAKTURAPAKET.SIFRAFAKTURE:=:OLD.SIFRAFAKTURE;END;

END IF;

END;

create or replace TRIGGER faktura\_ukupno\_pozivprocedure

AFTER INSERT OR UPDATE OR DELETE ON stavkafakture

DECLARE

sifra NUMBER:=FAKTURAPAKET.sifrafakture;

BEGIN

DISSABLE\_TRIGGER('faktura\_ukupno\_stop\_update',0);

UKUPNO(sifra);

DISSABLE\_TRIGGER('faktura\_ukupno\_stop\_update',1);

END;

Procedura

create or replace PROCEDURE UKUPNO

(sifrafak IN NUMBER) AS

suma NUMBER:=0; sifrastope NUMBER:=0;proc NUMBER;ukbezpdv DECIMAL:=0; ukpdv DECIMAL:=0;brojac number := 0;

BEGIN

select count(sifrafakture) into brojac

from stavkafakture

where sifrafakture = sifrafak;

if(brojac >= 1)

then

select sifrastopepdv into sifrastope

from faktura

where sifrafakture = sifrafak;

select procenat.dajbroj(PROCENATTIP) into proc

from stopapdv

where SIFRASTOPEPDV = sifrastope;

SELECT SUM(sf.kolicina\*p.TRENUTNACENA), SUM(sf.kolicina\*p.TRENUTNACENA\*proc/100)

INTO ukbezpdv, ukpdv

FROM proizvod p join stavkafakture sf on (p.sifraproizvoda=sf.sifraproizvoda)

WHERE sifrafakture = sifrafak;

UPDATE faktura

SET ukupno = ukbezpdv+ukpdv

WHERE sifrafakture = sifrafak;

else

update faktura

set ukupno = 0

where sifrafakture = sifrafak;

end if;

END;

Triger koji poziva proceduru pri ažuriranju polja kolicina

create or replace TRIGGER STAVKAFAKTURE\_UPDATE\_KOLICINA

AFTER UPDATE OF kolicina ON stavkafakture

DECLARE

sifra NUMBER := FAKTURAPAKET.sifrafakture;

BEGIN

DISSABLE\_TRIGGER('faktura\_ukupno\_stop\_update',0);

ukupno(sifra);

DISSABLE\_TRIGGER('faktura\_ukupno\_stop\_update',1);

END;

Triger koji onemogućava izmenu polja ukupno u tabeli Faktura

create or replace TRIGGER faktura\_ukupno\_stop\_update

before UPDATE OF ukupno ON faktura

for each row

declare

PRAGMA AUTONOMOUS\_TRANSACTION;

novavred NUMBER;

staravred NUMBER;

sifrafak NUMBER;

BEGIN

novavred:=:new.ukupno;

sifrafak := :new.sifrafakture;

select ukupno into staravred

from faktura

where sifrafakture=sifrafak;

:new.ukupno:= staravred;

commit;

END;