**BAZE PODATAKA**

Projekat

**Projektovanje baze podataka za podršku organizatorima mačevalačkih turnira**

Predmetni nastavnik: Studenti:

# **Dr Milosav Majstorović** Adam Milovanovic 355/18

# Aleksandar Ristić 556/19

# Datum Predaje\_\_\_\_\_\_\_\_\_\_\_\_

**Beograd**

**Jun, 2020**

# Sadržaj

1. Rezime.......................................................................................................................................3

2. Poslovni domen - OPIS......... ..................................................................................................4

3. Podmodeli podataka............... ...................................................................................................5

4. Integrisani model podataka (PMOV).............. ...........................................................................7

5. Prevodjenje PMOV u relacioni model ....... ................................................................................8

6. SQL naredbe za krerianje tabela................... ................................. ............................................10

7. SQL naredbe za umetanje podataka ........... ...............................................................................14

8. Tabelarni prikaz sadržaja tabela .............. ...................................................................................19

9. Prikaz baznih pogleda, procedura i funkcija.... ............................................................................22

9.1 Podsistem Registracija ...... ........................................................................................................22

9.1.1 Poslovna Transakcija (Opis)..... .................................................................................22

9.1.2 Bazna procedura za registrovanje učesnka......... ........................................................23

9.1.3 Pogled naplate dodataka..............................................................................................27

9.1.4 Pogled naplate disciplina.............................................................................................29

9.1.5 Pogled objedinjene naplate turnira..............................................................................31

9.2 Podsistem Turnir.........................................................................................................................32

9.2.1 Procedura za unos takmičara i sudija .........................................................................32

9.3 Integrisani sistem.....................................................................................................................35

9.3.1 Bazna funkcija koja vraća račun naplate turnira prijavljenom učesniku.......35

9.2.2 Poslovna Transakcija (Opis)...... ................................................................................37

9.2.3 Procedura za unos mečeva ....... .................................................................................38

9.2.4 Procedura za unos statistike ...................................................................................... 40

9.2.5 Pogled za pregled rezultata mečeva.. .........................................................................43

9.2.6 Bazna funkcija za prva tri osvojena mesta u disciplini...... ........................................45

9.2.7 Bazna funkcija koja vraća statistiku odabranog učesnika tokom celog turnira .........46

10 Literatura....................................... ...............................................................................................47

# 1. Rezime

U ovom projektu je izvršena analiza poslovnog domena organizacije eliminiacionog mačevalačkog turnir za potrebe mačevalačkih klubova, organizatora takvih dodgadjaja. Kreiran je model podataka za ovaj poslovni domen i izvršena je njegova implementacija kroz kreiranje šeme relacione baze podataka.Kroz SQL insert naredbe izvršseno je instanciranje ove šeme relacione baze podataka. U projektu je dat i tabelarni prikaz sadrzaja tabela baze podataka. Za svaki podsistem definisane su poslovne transakcije, dati pogledi i bazne procedure pomoću kojih se realizuju poslovne transakcije. Za svaki podsistem kreirane su i bazne funkcije cijim izvrsenjem su pokazane realizacije odgovarajucih poslovnih transakcija.

# 2. Poslovni domen – OPIS

Za potrebe organizatora turnira Istorijskog Evropskog Mačevanja, potrebno je isprojektovati bazu podataka sa neophodnim podacima. Poslovni domen sistema je dekomponovan na dva podsistema:

Registracija I Turnir. U daljem nastavku se daje kratak opis ovih podsistema.

***I Podsistem –Registracija (Adam Milovanović )***

U turnir mogu da se registruju samo clanovi Mačevalackih klubova. Mačevalacki klubovi pripadaju razlicitim Državama, podrazumevajuci da se u jednoj drzavi moze nalaziti vise klubova.

Za države pratimo naziv drzave I šifru drzave (‘SRB’, ‘USA’ I slicno…). Država moze imati 0 ili vise klubova. Klub mora pripadati nekoj državi. Klubovi imaju naziv I šifru. Učesnici turnira mogu da pripadaju samo jednom Klubu (u okviru ovog turnira), dok jedan Klub moze da ima vise učesnika u samom turniru. Za Učesnika se evidentira ime, prezime, šifraUcesnika. Turnir svojim ucesnicima organizuje opcionalne Dodatke, koji ukljucuju doručak, ručak I noćenje. Dodatci su organizovani u šifarniku Dodataka pri cemu se za svaki dodatak evidentira cena, naziv I šifra dodatka. (rucak, 500, ‘RUC’) Jedan ucesnik moze da odabere onoliko dodataka (brojDodataka) koliko organizatora turnira predvide da će biti neophodno za trajanje samog turnira (turnir u trajanju od tri dana, zahteva tri noćenja, recimo).Dakle svaki učesnik ima maksimalan broj odabranih dodataka, koji je u naše svrhe 3, za ovaj projekat podrazumevajuci da se turnir organizuje za tri različite discipline, održavajuci jednu disciplinu dnevno kroz tri dana. Disciplina je šifarnik za ogranicen I unapred definisani broj disciplina na turniru ( za nase svrhe su discipline Rapir, Longsword I Sidesword). Jedan učesnik moze bude registrovan za jednu ili maksimalno sve tri discipline. Svaka odabrana disciplina sa sobom nosi osnovnu cenu naplate. (Takmicenje se naplaćuje kotizacijom od 1000din po disciplini)

***Ucesnik po naplati registracije za svaku disciplinu postaje registrovani Učesnik***  koji dobija svoj redni broj. Jedan učesnik moze da se registruje maksimalno 1put za svaku od tri discipline. (dakle učesnik koji se takmiči na sve tri discipline će da bude registrovan 1put za svaku disciplinu u kojoj se takmiči, kako se discipline u turniru odžavaju nezavisno jedne od drugih.)

***II Podsistem – Turnir (Aleksandar Ristić)***

Registrovani učesnik mora da bude ili sudija na odabranoj disciplini ili takmicar u istoj. Mecevi Turnira se odvijaju po razlicitim fazama. (eliminaciona, polufinalna, finalna, meč za bronzu). Svaki meč ima dve sudije I dva takmicara, takodje jedan meč moze da se odvije samo u jednoj disciplini I jednoj fazi. Jedna disciplina naravno moze da ima vise mečeva kao što I svaka faza moze da ima više mečeva. Za svaki meč se vodi statistika ostvarenih rezultata svakog takmicara, kao I ishod meča. Statistika vodi evidencija o broju bodova koji nose 1, 2, 3 I 4 poena u zavisnosti od vrednosti ostvarenog udarca ili boda u okviru meča. Uz to statistika vodi racuna o ukupnom broju ostvarenih poena za svakog takmičara, ishod meča I broj izazova svakog takmičara.

# 3.Podmodeli podataka

U nastavku su prikazani podmodeli podataka, goreopisanih celina, kroz podmodele PMOV.



Slika 1. Podmodel PMOV – Registracija

**Registrovani-Učesnik** treba da predstavi sve moguće instance **Učesnika** na samom turniru, pri cemu se jedan učesnik pojavljuje maksimalno jednom po disciplini. (Nova registracija učesnika za svaku novu disciplinu koju prijavi)

Agregacijom **Učesnik-U-Disciplini** smo ostvarili uslov da svaki **Registrovani učesnik** mora sa sobom da bude vezan za jednu I samo jednu disciplinu po registraciji. Ovaj uslov je realizovan preko referencijalnog integriteta spoljnih ključeva, komponenti kompozitnog primarnog ključa agregacije **Ucesnik-U-Disciplini**, koji sprečava postojanje dva identicna reda kombinacije **Učesnik**a I **Discipline**.

Agregacijom **Odabrani-Dodatak**, smo vezali **Učesnike** za svaki njihov izabrani opcionalni dodatak u okviru dodataka samog turnira. Broj dodataka se definiše u samoj agregaciji za svaki izabrani dodatak, bez potrebe da se isti dodatak više puta vezuje za jednog **Učesnika**.



Slika 2. Podmodel PMOV Turnir

Jednom **Registrovan-Učesnik** na disciplini moze da učestvuje u turniru ili kao **Sudija** ili kao **Takmičar** te discipline, sto predstavlja posebnu pojavu istog entiteta, zbog cega su **Sudija** I **Takmicar** specijalizacije generalnog nad tipa **Registrovan-Ucesnik**. Podrazumeva se za svakog ucesnika da ce biti deo turnira u jednom od ta dva kapaciteta nakon registracije .

Meč moze biti realizovan kao agregacija, kako su pojave svih njegovih atributa(spoljnihKljuceva), garantovano NOT NULL polja usled restrikcije referencijalnog integriteta spoljnih ključeva. Medjutim, za potrebe projekta **Mec** realizujemo kao jak Entitet radi lakšeg referenciranja **Meča** u okviru pogleda I procedura samog projekta, dodelom jedinstvenog primarnog ključa #RedniBrojMeca.

U eventualnosti da se jedan ili oba takmicara ne pojave za predvidjeni meč, oni se I dalje zavode u **Meč** kao ucesnici, medjutim se usled odsustva jednog ili oba takmičara kao rezultat(ishodMeca) u njihovoj statistici za sam meč navodi “Poraz”.

Rezulat turnira, prvo, drugo I treće mesto kao I ostala relevantna statistika je predvidjen da se realizuje preko pogleda I procedura koje uključuju relevantne tabele za tu svrhu.

# 4. Integrisani model podataka (PMOV)



Slika 3. Integrisani PMOV

Neophodno je napomenuti kako podmodel-Registracija i podmodel-Mec koriste isti Entitet “Disciplina” pri cemu podmodel-Registracija vezuje Ucesnike sa njihovim odabranim disciplinama, dok podmodel-Mec vezuje meceve po disciplinama. Dakle oba podmodela zahtevaju pristup tabeli Disciplina. Podmodelu Registracija je neophodan za registrovanje ucesnika, dok je podmodelu Turnir neophodan radi praćenja statistike takmičara I mečeva za svaku disciplinu.

# 5. Prevodjenje PMOV u relacioni model

**Drzava**(DrzavaID#, NazivDrzave)

**Dodatak**(DodatakID#, NazivDodatka, CenaDodatka)

**Disciplina**(DisciplinaID#, NazivDiscip, CenaDiscip)

**FazaTakmicenja**(FazaID#, NazivFaze)

**Klub**(KlubID#, NazivKluba, DrzavaID#fk)

Klub(DrzavaID#fk) referencira Drzava(DrzavaID#)

**Ucesnik**(UcesnikID#, Ime, Prezime, KlubID#fk)

Ucesnik(KlubID#fk) referencira Klub(KlubID#)

**OdabraniDodatak**(UcesnikID#fk, DodatakID#fk, BrojDodataka)

OdabraniDodatak(UcesnikID#fk) referencira Ucesnik(UcesnikID#)

OdabraniDodatak(DodatakID#) referencira Dodatak(DodatakID#)

**RegistrovaniUcesnik**(UcesnikID#fk, DisciplinaID#fk)

RegistrovaniUcesnik(UcesnikID#fk) referencira Ucesnik(UcesnikID#)

RegistrovaniUcesnik(DisciplinaID#fk) referencira Disciplina(DiscipllinaID#)

**Sudija**(SudijaID#fk,DisciplinaID#fk)

Sudija(SudijaID#fk, DisciplinaID#fk)

referencira RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

**Takmicar**(TakmicarID#fk, DisciplinaID#fk)

Takmicar(TakmicarID#fk, DisciplinaID#fk)

referencira RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

**Mec**(RedniBrojMeca#, DisciplinaID#fk, PlaviBoracID#fk, PlaviDisciplinaID#fk, CrveniBoracID#fk, CrveniDisciplinaID#fk, Sudija1ID#fk, Sudija1DisciplinaID#fk, Sudija2ID#fk, Sudija2DisciplinaID#fk, FazaID#fk)

MEC(DisciplinaID#fk) referencira Disciplina(DisciplinaID#)

MEC(PlaviBoracID#fk, PlaviDisciplinaID#fk)

referencira Takmicar(TakmicarID#fk, DisciplinaID#fk)

MEC(CrveniBoracID#fk, CrveniDisciplinaID#fk)

referencira Takmicar(TakmicarID#fk, DisciplinaID#fk)

Mec(Sudija1ID#fk, Sudija1DisciplinaID#fk)

referencira Sudija(SudijaID#fk, DisciplinaID#fk)

Mec(Sudija2ID#fk, Sudija2DisciplinaID#fk)

referencira Sudija(SudijaID#fk, DisciplinaID#fk)

Mec(FazaID#fk) referencira Faza(FazaID#)

Ogranicenje( Sudija1DisciplinaID#fk = Sudija2DisciplinaID#fk = PlaviDisciplinaID#fk = CrveniDisciplinaID#fk = DisciplinaID#fk)

**Statistika**(RedniBrMeca#fk,TakmicarID#fk, DisciplinaID#fk, 1Poen, 2Poena, 3Poena, 4Poena, brChallenge, ishodMeca)

Statistika(RedniBrMeca#fk) referencira Mec(redniBrMeca#)

Statistika(TakmicarID#fk, DisciplinaID#fk)

referencira Takmicar(TakmicarID#fk, DisciplinaID#fk)

# 6. SQL naredbe za kreiranje tabela

USE master;

GO

--Ako postoji neka stara baza podataka sa imenom Swordplay, unistimo je(drop)

--I kreiramo novu bazu sa tim imenom

IF EXISTS(SELECT name from master.dbo.sysdatabases WHERE name = N'Swordplay')

DROP DATABASE Swordplay;

GO

CREATE DATABASE Swordplay;

GO

--Korisnik upotrebljava kreiranu bazu

USE Swordplay

GO

CREATE TABLE Drzava(

DrzavaID# CHAR(3) NOT NULL,

NazivDrzave NVARCHAR(30) NOT NULL,

PRIMARY KEY(DrzavaID#)

);

CREATE TABLE Dodatak(

DodatakID# CHAR(3) NOT NULL, --'dor''ruc''noc'

NazivDodatka VARCHAR(30) NOT NULL,

CenaDodatka DECIMAL(19,4) NOT NULL, --Koristimo decimal umesto MONEY zbog racunanja u procedurama

PRIMARY KEY(DodatakID#)

);

CREATE TABLE Disciplina(

DisciplinaID# CHAR(3) NOT NULL, --'RAP' 'LON' 'SID'

NazivDiscip VARCHAR(30) NOT NULL,

CenaDiscip DECIMAL(19,4) NOT NULL,

PRIMARY KEY(DisciplinaID#)

);

CREATE TABLE FazaTakmicenja(

FazaID# CHAR(6) NOT NULL, --'poluF', 'final', 'cetFin', 'bronza'

NazivFaze VARCHAR(30),

PRIMARY KEY(FazaID#)

);

CREATE TABLE Klub(

KlubID# CHAR(2) NOT NULL, --00, 01, 02,...,99

NazivKluba NVARCHAR(30) NOT NULL,

DrzavaID#fk CHAR(3) NOT NULL,

PRIMARY KEY(KlubID#),

FOREIGN KEY(DrzavaID#fk) REFERENCES Drzava(DrzavaID#)

);

CREATE TABLE Ucesnik(

UcesnikID# CHAR(3) NOT NULL, --001, 002,003,...,999

ime NVARCHAR(30) NOT NULL,

Prezime NVARCHAR(30) NOT NULL,

KlubID#fk CHAR(2) NOT NULL,

PRIMARY KEY(UcesnikID#),

FOREIGN KEY(KlubID#fk) REFERENCES KLUB(KlubID#)

);

CREATE TABLE OdabraniDodatak(

UcesnikID#fk CHAR(3) NOT NULL,

DodatakID#fk CHAR(3) NOT NULL,

BrojDodataka INT NOT NULL

PRIMARY KEY(UcesnikID#fk, DodatakID#fk),

FOREIGN KEY(UcesnikID#fk) REFERENCES Ucesnik(UcesnikID#),

FOREIGN KEY(DodatakID#fk) REFERENCES Dodatak(DodatakID#)

);

CREATE TABLE RegistrovaniUcesnik(

UcesnikID#fk CHAR(3) NOT NULL,

DisciplinaID#fk CHAR(3) NOT NULL,

PRIMARY KEY(UcesnikID#fk, DisciplinaID#fk),

FOREIGN KEY(UcesnikID#fk) REFERENCES Ucesnik(UcesnikID#),

FOREIGN KEY(DisciplinaID#fk) REFERENCES Disciplina(DisciplinaID#)

);

-- Ispod Sudija i Takmicar tabele kao specijalizacije tabele Registrovani Ucesnik

CREATE TABLE Sudija(

SudijaID#fk CHAR(3) NOT NULL,

DisciplinaID#fk CHAR(3) NOT NULL,

PRIMARY KEY(SudijaID#fk, DisciplinaID#fk),

FOREIGN KEY(SudijaID#fk, DisciplinaID#fk) REFERENCES RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

);

CREATE TABLE Takmicar(

TakmicarID#fk CHAR(3) NOT NULL,

DisciplinaID#fk CHAR(3) NOT NULL,

PRIMARY KEY(TakmicarID#fk, DisciplinaID#fk),

FOREIGN KEY(TakmicarID#fk, DisciplinaID#fk)

REFERENCES RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

);

CREATE TABLE Mec(

RedniBrojMeca# INT NOT NULL,

DisciplinaID#fk CHAR(3) NOT NULL,

--Plavi borac

PlaviBoracID#fk CHAR(3) NOT NULL,

PlaviDisciplinaID#fk CHAR(3) NOT NULL,

--

--Crveniborac

CrveniBoracID#fk CHAR(3) NOT NULL,

CrveniDisciplinaID#fk CHAR(3) NOT NULL,

--

--Sudija1

Sudija1ID#fk CHAR(3) NOT NULL,

Sudija1DisciplinaID#fk CHAR(3) NOT NULL,

--

--Sudija2

Sudija2ID#fk CHAR(3) NOT NULL,

Sudija2DsciplinaID#fk CHAR(3) NOT NULL,

--

FazaID#fk CHAR(6) NOT NULL,

PRIMARY KEY(RedniBrojMeca#),

FOREIGN KEY(DisciplinaID#fk) REFERENCES Disciplina(DisciplinaID#),

--

--Plavi Borac foreign key

FOREIGN KEY(PlaviBoracID#fk, PlaviDisciplinaID#fk)

REFERENCES TAKMICAR(TakmicarID#fk, DisciplinaID#fk),

--

--Crveni Borac Foreign Key

FOREIGN KEY(CrveniBoracID#fk, CrveniDisciplinaID#fk)

REFERENCES TAKMICAR(TakmicarID#fk, DisciplinaID#fk),

--

--Sudija1 Foreign key

FOREIGN KEY(Sudija1ID#fk, Sudija1DisciplinaID#fk)

REFERENCES Sudija(SudijaID#fk, DisciplinaID#fk),

--

--Sudija2 Foreign key

FOREIGN KEY(Sudija2ID#fk, Sudija2DsciplinaID#fk)

REFERENCES Sudija(SudijaID#fk, DisciplinaID#fk),

--

FOREIGN KEY(FazaID#fk) REFERENCES FazaTakmicenja(FazaID#),

-- Dole kreiramo granicenje cime obezbedjujemo da moze samo da se kreira mec za jednu disciplinu ako su sve sudije i borci registrovani na istu disciplinu

CONSTRAINT IstaDisciplina CHECK(DisciplinaID#fk = PlaviDisciplinaID#fk

AND PlaviDisciplinaID#fk = CrveniDisciplinaID#fk

AND Sudija1DisciplinaID#fk = DisciplinaID#fk

AND Sudija2DsciplinaID#fk = Sudija1DisciplinaID#fk)

);

CREATE TABLE Statistika(

--Komponenta Primarnog kljuca prema mecu

RedniBrojMeca#fk INT NOT NULL,

--

--Komponenta primarnog kljuca prema TAKMICARU

TakmicarID#fk CHAR(3) NOT NULL,

DisciplinaID#fk CHAR(3) NOT NULL,

--

JedanPoen INT NOT NULL,

DvaPoena INT NOT NULL,

TriPoena INT NOT NULL, --> Unosi se vrednost 0 ako nije ostvaren nijedan poen

CetiriPoena INT NOT NULL,

UkupnoPoena INT NOT NULL,

BrojIzazova INT NOT NULL, --> Ako se izazov uvazi, takmicar ima prava na sledeci izazov (teoretski takmicari ako su uvek u pravu nemaju ogranicen broj izazova na raspolaganju)

IshodMeca CHAR(6) NOT NULL --> 'pobeda' 'poraz'

CONSTRAINT Ishod CHECK(IshodMeca IN('pobeda', 'poraz')),

PRIMARY KEY(RedniBrojMeca#fk, TakmicarID#fk, DisciplinaID#fk),

FOREIGN KEY(RedniBrojMeca#fk) REFERENCES MEC(RedniBrojMeca#),

FOREIGN KEY(TakmicarID#fk, DisciplinaID#fk)

REFERENCES Takmicar(TakmicarID#fk, DisciplinaID#fk)

);

# 7. SQL naredbe za umetanje podataka

--Sledecim INSERT naredbama kreiramo instancu jednog mini-turnira za disciplinu "Longsword".

-- DRZAVA(drzavaID,nazivDrzave)

INSERT INTO DRZAVA VALUES ('SRB','Srbija');

INSERT INTO DRZAVA VALUES ('BUG','Bugarska');

INSERT INTO Drzava VALUES('HUN', 'Madjarska');

--dodatak(dodatakID,nazivDodatka,CenaDodatka)

INSERT INTO DODATAK VALUES('dor','dorucak',800.00);

INSERT INTO DODATAK VALUES('ruc','rucak',1000.00);

INSERT INTO DODATAK VALUES('noc','nocenje',3000.00);

--disciplina(disciplinaID,nazivDiscip,CenaDiscip)

INSERT INTO DISCIPLINA VALUES('RAP','rapir',1000.00);

INSERT INTO DISCIPLINA VALUES('LON','longsword',1000.00);

INSERT INTO DISCIPLINA VALUES('SID','sidesword',1000.00);

--faza takmicenja(fazaID,nazivfaze)

INSERT INTO FazaTakmicenja VALUES('poluf','polufinale');

INSERT INTO FazaTakmicenja VALUES('final','finale');

INSERT INTO FazaTakmicenja VALUES('cetFi','cetvrtfinale');

INSERT INTO FazaTakmicenja VALUES('bronza','bronza mec'); /\*Mec za bronzu\*/

INSERT INTO FazaTakmicenja VALUES('grup','grupna faza');

INSERT INTO FazaTakmicenja VALUES('elim','eliminacije');

--KLUB(klubID(01,02..),ime kluba,drzavaID)

INSERT INTO KLUB VALUES('01','Terca','SRB');

INSERT INTO KLUB VALUES('02','MOTUS','BUG');

INSERT INTO Klub VALUES ('03', 'ARS ENSIS', 'HUN');

--za mini turnir inicijalne instance Baze su nam su dovoljna dva kluba inace bi bilo i jos njih

--ucesnik(ucesnikID(001,002...),ime,prezime,klubID)

--Ispod ucesnici u buduce registrovani kao Takmicari

INSERT INTO UCESNIK VALUES('001','Angel','Chernaev','02');

INSERT INTO UCESNIK VALUES('002','Stefan','Milenkovic','01');

INSERT INTO UCESNIK VALUES('003','Borislav','Krustev','02');

INSERT INTO UCESNIK VALUES('004','Adam','Milovanovic','01');

--Ispod Ucesnici registrovani za sudije

INSERT INTO UCESNIK VALUES('005','Aleksandar','Ristic','01');

INSERT INTO UCESNIK VALUES('006','Jelena','Arsenijevic','01'); --da imamo dvoje za sudiju

--odabrani dodatak(ucesnikID(char3),dodatakID(char3),brojdodatka(int)

INSERT INTO OdabraniDodatak VALUES('001','dor',1);

INSERT INTO OdabraniDodatak VALUES('002','ruc',1);

INSERT INTO OdabraniDodatak VALUES('003','noc',1);

INSERT INTO OdabraniDodatak VALUES('001','noc',1);

INSERT INTO OdabraniDodatak VALUES('004','ruc',1);

--registrovaniucesnik(ucesnikID(char3),disciplinaID(char3)

INSERT INTO RegistrovaniUcesnik VALUES('001','lon');

INSERT INTO RegistrovaniUcesnik VALUES('002','lon');

INSERT INTO RegistrovaniUcesnik VALUES('003','lon');

INSERT INTO RegistrovaniUcesnik VALUES('004','lon');

INSERT INTO RegistrovaniUcesnik VALUES('005','lon');

INSERT INTO RegistrovaniUcesnik VALUES('006','lon');

-- Ovde primer registrovanih ucesnika za jednu disciplinu (Turnir za Longsword)

--sudijaID=ucesnikID,disciplinaID

INSERT INTO Sudija VALUES('005','lon');

INSERT INTO Sudija VALUES('006','lon');

--sad samo dva kao minimum

--takmicar(takmicarID=ucesnikID,disciplinaID)

INSERT INTO Takmicar VALUES('001','lon');

INSERT INTO Takmicar VALUES('002','lon');

INSERT INTO Takmicar VALUES('003','lon');

INSERT INTO Takmicar VALUES('004','lon');

--Mec(redniBrmeca(int),discID(char3),plaviborac=ucesnikID(char3 00x),discID(char3),

--u istom maniru dodati za crvenog borca i za obe sudije odgovarajuca polja)

INSERT INTO Mec VALUES(001,'lon','001','lon','002','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(002,'lon','001','lon','003','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(003,'lon','001','lon','004','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(004,'lon','002','lon','003','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(005,'lon','001','lon','004','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(006,'lon','003','lon','004','lon','005','lon','006','lon','grup');

INSERT INTO Mec VALUES(007,'lon','001','lon','002','lon','005','lon','006','lon','poluf');

INSERT INTO Mec VALUES(008,'lon','003','lon','004','lon','005','lon','006','lon','poluf');

INSERT INTO Mec VALUES(009,'lon','001','lon','002','lon','005','lon','006','lon','final');

--Limitiramo svako sa svakim za grupnu fazu,onda dve za poluf,i jedna za kraj

--statistikaID(redniBrMeca,takmicarID=ucesnikID,discID, jp,dp,tp,cp,up,bi-int,ishod-char6(pobeda/poraz)

INSERT INTO STATISTIKA VALUES(001,'001','lon',1,1,1,1,10,0,'pobeda');

INSERT INTO STATISTIKA VALUES(001,'002','lon',1,0,0,1,5,0,'poraz');

INSERT INTO STATISTIKA VALUES(002,'001','lon',1,1,1,1,10,0,'pobeda');

INSERT INTO STATISTIKA VALUES(002,'003','lon',0,0,0,1,4,0,'poraz');

INSERT INTO STATISTIKA VALUES(003,'001','lon',1,1,1,1,10,0,'pobeda');

INSERT INTO STATISTIKA VALUES(003,'004','lon',1,0,0,0,1,0,'poraz');

INSERT INTO STATISTIKA VALUES(004,'002','lon',1,1,1,2,14,0,'pobeda');

INSERT INTO STATISTIKA VALUES(004,'003','lon',1,0,0,1,5,0,'poraz');

INSERT INTO STATISTIKA VALUES(005,'002','lon',3,1,1,2,16,0,'pobeda');

INSERT INTO STATISTIKA VALUES(005,'004','lon',1,0,0,1,5,0,'poraz');

INSERT INTO STATISTIKA VALUES(006,'003','lon',0,0,0,4,16,0,'poraz');

INSERT INTO STATISTIKA VALUES(006,'004','lon',1,1,2,2,5,17,'pobeda');

INSERT INTO STATISTIKA VALUES(007,'001','lon',1,1,1,1,10,0,'pobeda');

INSERT INTO STATISTIKA VALUES(007,'003','lon',1,3,0,1,7,0,'poraz');

INSERT INTO STATISTIKA VALUES(008,'002','lon',1,1,1,5,26,0,'pobeda');

INSERT INTO STATISTIKA VALUES(008,'004','lon',1,1,1,4,20,0,'poraz');

INSERT INTO STATISTIKA VALUES(009,'001','lon',1,1,1,1,10,0,'poraz');

INSERT INTO STATISTIKA VALUES(009,'002','lon',1,1,2,2,16,0,'pobeda');

--Registracija za Turnir Rapir

--Pocetak Insert naredbi za rapir turnir

INSERT INTO RegistrovaniUcesnik VALUES('001','rap');

INSERT INTO RegistrovaniUcesnik VALUES('002','rap');

INSERT INTO RegistrovaniUcesnik VALUES('003','rap');

INSERT INTO RegistrovaniUcesnik VALUES('004','rap');

INSERT INTO RegistrovaniUcesnik VALUES('005','rap');

INSERT INTO RegistrovaniUcesnik VALUES('006','rap');

--Gorepomenuti ucesnici su registrovani na dva turnira(Longsword I ovde Rapir)

--sudijaID=ucesnikID,disciplinaID

INSERT INTO Sudija VALUES('001','rap');

INSERT INTO Sudija VALUES('002','rap');

--sad samo dva kao minimum

--takmicar(takmicarID=ucesnikID,disciplinaID)

INSERT INTO Takmicar VALUES('003','rap');

INSERT INTO Takmicar VALUES('004','rap');

INSERT INTO Takmicar VALUES('005','rap');

INSERT INTO Takmicar VALUES('006','rap');

--Mec(redniBrmeca(int),discID(char3),plaviborac=ucesnikID(char3 00x),discID(char3),

--u istom maniru dodati za crvenog borca i za obe sudije odgovarajuca polja)

--Grupna faza

INSERT INTO Mec VALUES(010,'rap','003','rap','004','rap','001','rap','002','rap','grup');

INSERT INTO Mec VALUES(011,'rap','003','rap','005','rap','001','rap','002','rap','grup');

INSERT INTO Mec VALUES(012,'rap','003','rap','006','rap','001','rap','002','rap','grup');

INSERT INTO Mec VALUES(013,'rap','004','rap','005','rap','001','rap','002','rap','grup');

INSERT INTO Mec VALUES(014,'rap','005','rap','006','rap','001','rap','002','rap','grup');

INSERT INTO Mec VALUES(015,'rap','006','rap','004','rap','001','rap','002','rap','grup');

--Limitiramo svako sa svakim za grupnu fazu,onda dve za poluf,i jedna za kraj

--Poluf

INSERT INTO Mec VALUES(016,'rap','003','rap','005','rap','001','rap','002','rap','poluf');

INSERT INTO Mec VALUES(017,'rap','005','rap','006','rap','001','rap','002','rap','poluf');

--Finale

INSERT INTO Mec VALUES(018,'rap','003','rap','005','rap','001','rap','002','rap','final');

--statistikaID(redniBrMeca,takmicarID=ucesnikID,discID, jp,dp,tp,cp,up,bi-int,ishod-char6(pobeda/poraz)

INSERT INTO STATISTIKA VALUES(010,'003','rap',0,1,1,1,9,0,'pobeda');

INSERT INTO STATISTIKA VALUES(010,'004','rap',1,0,0,0,1,0,'poraz');

INSERT INTO STATISTIKA VALUES(011,'003','rap',0,0,0,3,12,0,'pobeda');

INSERT INTO STATISTIKA VALUES(011,'005','rap',0,0,0,1,4,0,'poraz');

INSERT INTO STATISTIKA VALUES(012,'003','rap',2,0,0,2,10,0,'pobeda');

INSERT INTO STATISTIKA VALUES(012,'006','rap',2,0,0,0,2,0,'poraz');

INSERT INTO STATISTIKA VALUES(013,'004','rap',2,0,0,3,14,0,'pobeda');

INSERT INTO STATISTIKA VALUES(013,'005','rap',0,0,2,0,6,0,'poraz');

INSERT INTO STATISTIKA VALUES(014,'005','rap',0,0,0,2,8,0,'pobeda');

INSERT INTO STATISTIKA VALUES(014,'006','rap',0,0,0,0,0,2,'poraz');

INSERT INTO STATISTIKA VALUES(015,'006','rap',1,2,1,0,8,1,'pobeda');

INSERT INTO STATISTIKA VALUES(015,'004','rap',0,0,0,0,0,0,'poraz');

INSERT INTO STATISTIKA VALUES(016,'003','rap',1,1,1,2,14,0,'pobeda');

INSERT INTO STATISTIKA VALUES(016,'005','rap',2,3,0,0,8,0,'poraz');

INSERT INTO STATISTIKA VALUES(017,'005','rap',0,0,5,0,15,2,'pobeda');

INSERT INTO STATISTIKA VALUES(017,'006','rap',2,1,1,0,7,0,'poraz');

INSERT INTO STATISTIKA VALUES(018,'003','rap',0,1,1,1,9,0,'poraz');

INSERT INTO STATISTIKA VALUES(018,'005','rap',2,1,0,3,20,0,'pobeda');

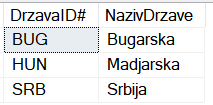
# 8. Tabelarni prikaz sadržaja tabela



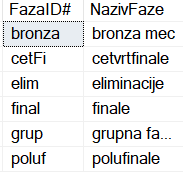
Slika 1. Tabela – Angazovanje



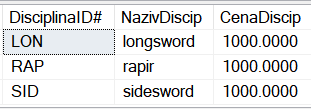
Slika 2. Tabela – Dodatak



Slika 3. Tabela – Drzava



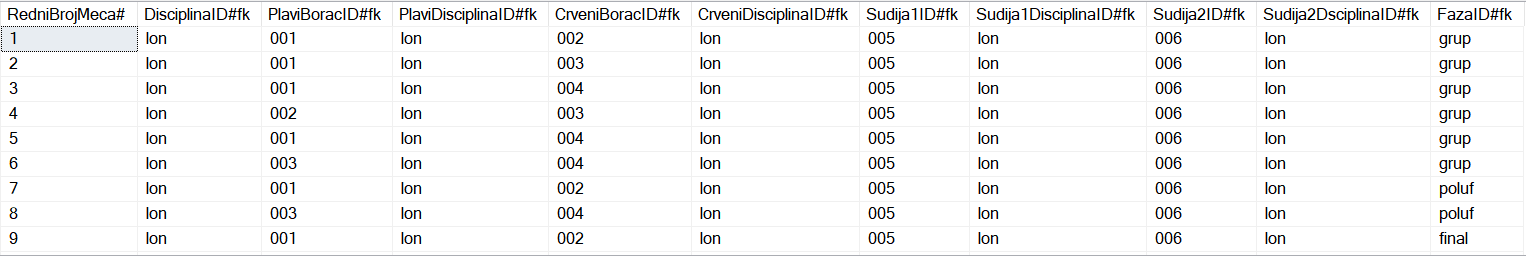
Slika 4. Tabela – FazaTakmicenja



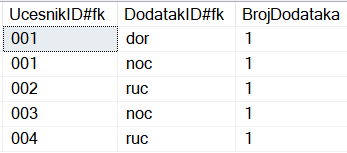
Slika 5. Tabela – Disciplina



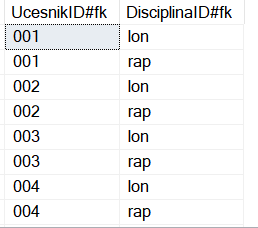
Slika 6. Tabela – Ucesnik



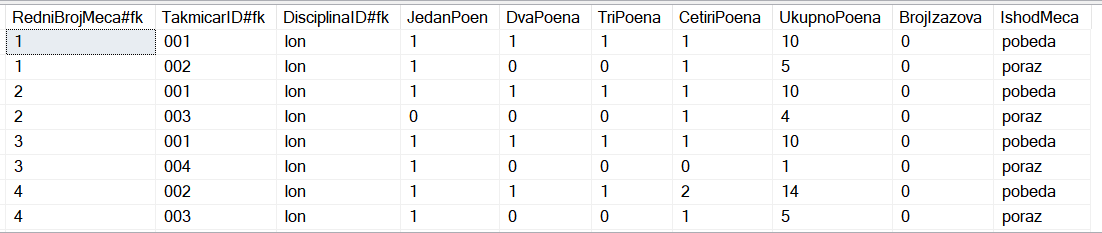
Slika 7. Tabela – Mec



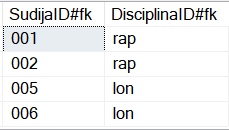
Slika 8. Tabela - OdabraniDodatak



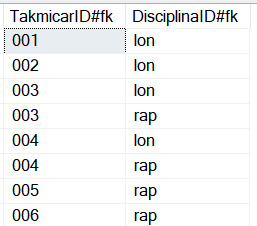
Slika 9. Tabela – RegistrovaniUcesnik



Slika 10. Tabela – Statistika



Slika 11. Tabela – Sudija



Slika 12. Tabela – Takmicar

# 9. Prikaz baznih pogleda, procedura i funkcija

U ovom poglavlju, po podsistemima se daju prikazi pogleda, baznih procedura i funkcija.

## 9.1 Podsistem Registracija

## 9.1.1 Poslovna transakcija I bazna procedura

### 9.1.1.1 *Definicija poslovne transakcije*

--Pretpostavka je da će izvestan broj dana pre datuma održavanja turnira klubovi da prijave svoje prisustvo na turniru kao posrednici eventualnim članovima svog kluba koji su izrazili želju da učestvuju. Najkasnije dve nedelje pred dogadjaj. U bazu se vrši direktan insert klubova ako vec ne postoje u bazi. Dve nedelje pred dogadjaj Učesnici klubova imaju pristup formi za on-line registraciju za koju je neophodno kreirati proceduru ‘kdpRegistrovanje\_Novog\_Ucesnika’ koja bi ažurirala registrovane učesnike u tabelama baze.

On-line registracija učesnika je transakcija neophodna za naš sistem, preko koje ostvarujemo mogućnost takmicarima da se registruju imenom I prezimenom odaberu dodatke na raspolaganju. (Dorucak I rucak u restoranu) kao I nocenje u hotelu I mogućnost da se opredele za jednu ili više disciplina u samom turniru.

--Pretpostavka je da ucesnici biraju na svojoj on-line formi unapred ponudjene vrednosti za disciplinu I dodatke, koji bi prosledjivali bazi parametre za sifru discipline I dodatka u samoj proceduri preko koje se registruju.

Po završenoj on-line registraciji, učesnicima se mail’om prosledjuje njihov jedinstveni UcesnikID# kreiran u našoj bazi, sa napomenom da se njome služe kao šifrom njihove registracije sa kojom se prijavljuju pri ličnom dolaženju na sam turnir).

Učesnici koji su evidentirani u bazi preko on-line registracije, vrše plaćanje na licu mesta potvrdom svog prisustva turniru, koji se vrši potpisom I prilaganjem šifre registracije na prijemnom šalteru. Volonteri na šalteru preko funkcije ‘kdfPrikaz\_Racuna(@SifraUcesnika CHAR(3))’ za priloženu šifru svakog učesnika (koji se učesnicima prosledjuje putem mail’a po završenoj on-line registraciji) dobijaju informacije o iznosu naplata koji su dužni da traže od učesnika prilikom prijave na sam turnir.

**(U kontekstu prikaza rada I funkcionalnosti procedura specifičnih sistemu, od ovog momenta, radimo sa novom instancom baze koja u sebi ne sadrzi prethodno umetnute učesnike, takmičare, mečeve Ili statistike, kako je sistem predvidjen da prethodno pomenute vrednosti ažurira putem procedura kreiranih u cilju praćenja dogadjaja samog turnira, pa je instanca baze prilikom svog rada u poslovnom domenu Mačevalačkog Turnira predvidjena da započne rad sa praznim tabelama; ‘Ucesnik’, ‘RegistrovaniUcesnik’, ‘OdabraniDodaci’, ‘Statistika’, ‘Meč’, ‘Sudja’ i ‘Takmičar’ koje bi tek nakon održavanja samog turnira bile popunjene za definisane discipline).**

Kod za krerianje procedure *kdpRegistrovanje\_Novog\_Ucesnika* sledi u daljem tekstu

### 9.1.2 Bazna procedura

Napraviti baznu proceduru kdpRegistrovanje\_Novog\_Ucesnika kojom se za svakog ucesnika unose opcionalni parametri, Disciplina1, Disciplina2, Disciplina3, kao I opcionalni parametri za broj dodataka (Broj dorucaka, rucaka, nocenja). Za svakog ucesnika je u proceduri neophodno uneti Sifru ucesnika, Ime, Prezime, Sifru Kluba. Učesnik ne može da se registruje ako njegov klub nije već umetnut u bazi, takodje učesnik mora u proceduri da unese makar jednu disciplinu da bi registracija bila uspešna.

USE Swordplay

GO

CREATE PROCEDURE kdpRegistrovanje\_Novog\_Ucesnika

--Potrebni Parametri

@Ime NVARCHAR(30),

@Prezime NVARCHAR (30),

@SifraUcesnika CHAR(3),

@SifraKluba CHAR(2),

@Disciplina1 CHAR(3) = NULL,

@Disciplina2 CHAR(3) = NULL,

@Disciplina3 CHAR(3) = NULL, --Dodelom inicijalne NULL vrednosti

@BrojDorucaka INT = NULL, --Obezbedjujemo da su svi parametri opcionalni

@BrojRucaka INT = NULL, --MEdjutim ako su sve tri discipline null

@BrojNocenja INT = NULL --Sama procedura vraca gresku da je obavezna bar jedna disciplina

AS

DECLARE

--Lokalne varijable

@MaxDodataka AS INT

BEGIN TRY

SET XACT\_ABORT ON -- automatsko sprecavanje daljeg izvrsenja naredbi ukoliko dodje do greske

SET @MaxDodataka = 3;

/\*Ako korisnik iz nekog razloga unese nulu kao vrednost, polje ostaje null\*/

IF @BrojDorucaka = 0

BEGIN

SET @BrojDorucaka = NULL;

END

IF @BrojRucaka = 0

BEGIN

SET @BrojRucaka = NULL;

END

IF @BrojNocenja = 0

BEGIN

SET @BrojNocenja = Null;

END

/\*Proveravamo za moguce duplikat unose \*/

IF @SifraUcesnika IN (SELECT UcesnikID# FROM Ucesnik)

BEGIN

PRINT 'Sifra ucesnika vec postoji u bazi';

END

IF @SifraKluba NOT IN (SELECT KlubID# FROM Klub)

BEGIN

PRINT 'Navedeni klub nije registrovan za ovaj turnir';

END

/\*Logicka greska bi bila voditi evidenciju o ucesniku koji ne ucestvuje ni na jednom turniru\*/

IF @Disciplina1 IS NULL AND @Disciplina2 IS NULL AND @Disciplina3 IS NULL

BEGIN

PRINT 'Ucesnik mora da izabere bar jednu disciplinu';

END

/\*Proveravamo da li je unos broja dodataka veci od raspolozivog maksimuma po ucesniku\*/

IF @BrojDorucaka > @MaxDodataka OR @BrojRucaka > @MaxDodataka OR @BrojNocenja > @MaxDodataka

BEGIN

PRINT 'Unesena vrednost prevazilazi max\_broj dodataka';

END

/\*Proveravamo da li je u proceduri unesena neka disciplina koja ne postoji u bazi\*/

IF ((@Disciplina1 NOT IN(SELECT DisciplinaID# FROM Disciplina))

OR (@Disciplina2 NOT IN(SELECT DisciplinaID# FROM Disciplina))

OR (@Disciplina3 NOT IN(SELECT DisciplinaID# FROM Disciplina)))

BEGIN

PRINT 'Ne postoji uneta discpline, proverite parametre';

END

/\*Proveravamo da li je korisnik uneo vise puta istu disciplinu \*/

IF(@Disciplina1 = @Disciplina2 OR @Disciplina2 = @Disciplina3 OR @Disciplina3 = @Disciplina1)

BEGIN

PRINT 'Visestruk unos iste discipline nije dozvoljen';

END

/\*Zavrsavamo sa proverama greski za unete parametre

--zapocinjemo sa transakcijom koja azurira neophodne tabele za registraciju novogUcesnika

--Neophodno je za svaki dodatak i za svaku disciplinu proveriti ako je unet parametar

--I za one discpline i dodatke za koje parametar nije NULL vrednost izvrsiti odgovarajuce insert naredbe \*/

BEGIN --Zapocinjemo niz naredbi

BEGIN TRANSACTION --Unutar niza naredbi zapocinjemo eksplicitnu transakciju

--Azuriramo tabelu ucesnik, sa korisnikovim unetim informacijama

INSERT INTO UCESNIK(UcesnikID#, Ime, Prezime, KlubID#fk)

VALUES(@SifraUcesnika, @Ime, @Prezime, @SifraKluba)

PRINT 'Ubacen je novi ucesnik'

IF (@Disciplina1 IS NOT NULL)

BEGIN

INSERT INTO RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

VALUES(@SifraUcesnika, @Disciplina1)

PRINT 'Ucesnik je registrovan na discplinu: ' + CAST(@Disciplina1 AS VARCHAR(10));

END

IF (@Disciplina2 IS NOT NULL)

BEGIN

INSERT INTO RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

VALUES(@SifraUcesnika, @Disciplina2)

PRINT 'Ucesnik je registrovan na discplinu: ' + CAST(@Disciplina2 AS VARCHAR(10));

END

IF (@Disciplina3 IS NOT NULL)

BEGIN

INSERT INTO RegistrovaniUcesnik(UcesnikID#fk, DisciplinaID#fk)

VALUES(@SifraUcesnika, @Disciplina3)

PRINT 'Ucesnik je registrovan na discplinu: ' + CAST(@Disciplina3 AS VARCHAR(10));

END

IF (@BrojDorucaka IS NOT NULL)

BEGIN

INSERT INTO OdabraniDodatak(UcesnikID#fk, DodatakID#fk, brojDodataka)

VALUES(@SifraUcesnika, 'dor', @BrojDorucaka)

PRINT 'Odabran je Dorucak \* ' + CAST(@BrojDorucaka AS VARCHAR(10));

END

IF(@BrojRucaka IS NOT NULL)

BEGIN

INSERT INTO OdabraniDodatak(UcesnikID#fk, DodatakID#fk, BrojDodataka)

VALUES(@SifraUcesnika, 'ruc', @BrojRucaka)

PRINT 'Odabran je Rucak \* ' + CAST(@BrojRucaka AS VARCHAR(10));

END

IF(@BrojNocenja IS NOT NULL)

BEGIN

INSERT INTO OdabraniDodatak(UcesnikID#fk, DodatakID#fk, BrojDodataka)

VALUES(@SifraUcesnika, 'noc', @BrojNocenja)

PRINT'Odabrano je nocenje \* ' + CAST(@BrojNocenja AS VARCHAR(10));

END

/\*Potvrdjujemo azuriranje tabela gorenapisanim insert narebama\*/

COMMIT TRANSACTION

END /\* Kraj Niza naredbi \*/

END TRY

BEGIN CATCH

PRINT 'Doslo je do pojave greske!';

PRINT'-----Proverite ulazne parametre -------';

PRINT ERROR\_MESSAGE();

/\*Ponistavamo sve promene insert naredbi do kojih je doslo pre greske\*/

ROLLBACK TRANSACTION

PRINT 'Procedura je ponistila promene';

END CATCH;

GO

/\*Poziv procedure za unos I registraciju svih takmicara za turnir Rapir discipline

Ovom procedurom smo zamenili sve prvobitne insert naredbe za podmodel Registracija

koje su se odnosile na unos ucesnika, registrovanje na disciplinu I odabir dodataka.\*/

EXEC kdpRegistrovanje\_Novog\_Ucesnika

@SifraUcesnika = '001', @Ime = 'Angel', @Prezime = 'Chernaev', @SifraKluba = '02',

@BrojDorucaka = 3, @BrojRucaka = 2, @BrojNocenja = 2, @Disciplina1 = 'rap', @Disciplina2 = 'lon', @Disciplina3 = 'sid'

EXEC kdpRegistrovanje\_Novog\_Ucesnika

@SifraUcesnika = '002', @Ime = 'Stefan', @Prezime = 'Milenkovic', @SifraKluba = '01',

@BrojDorucaka = 1, @BrojRucaka = 1, @Disciplina1 = 'rap', @Disciplina2 = 'lon', @Disciplina3 = 'sid'

EXEC kdpRegistrovanje\_Novog\_Ucesnika

@SifraUcesnika = '003', @Ime = 'Borislav', @Prezime = 'Krustev', @SifraKluba = '02',

@BrojDorucaka = 3, @BrojRucaka = 3, @BrojNocenja = 3, @Disciplina1 = 'rap', @Disciplina2 = 'lon', @Disciplina3 = 'sid'

EXEC kdpRegistrovanje\_Novog\_Ucesnika

@SifraUcesnika = '004', @Ime = 'Adam', @Prezime = 'Milovanovic', @SifraKluba = '01',

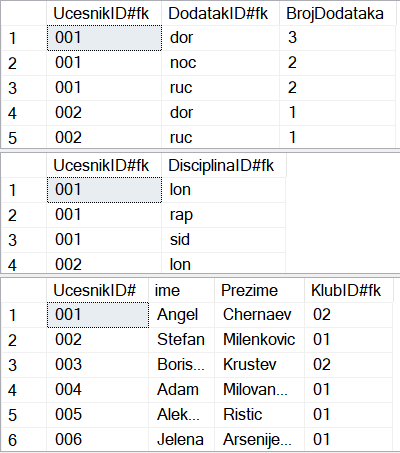
@BrojDorucaka = 0, @BrojRucaka = 3, @BrojNocenja = 0, @Disciplina1 = 'rap', @Disciplina2 = 'lon', @Disciplina3 = 'sid'

EXEC kdpRegistrovanje\_Novog\_Ucesnika

@SifraUcesnika = '005', @Ime = 'Aleksandar', @Prezime = 'Ristic', @SifraKluba = '01',

@BrojDorucaka = 0, @BrojRucaka = 0, @BrojNocenja = 0, @Disciplina1 = 'rap', @Disciplina2 = 'lon', @Disciplina3 = 'sid'

...



Rezultat Azuriranja tabela procedurom kdpRegistrovanje\_Novog\_Ucesnika

### 9.1.3 Pogled

Za Podmodel Registracija je neophodno takodje kreirati pogled **pUkupna\_Naplata\_Dodataka** koji evidentira broj svakog dodataka za koji se opredeljuju pojedini ucesnici, istovremeno racunajuci ukupan iznos za naplatu Dodataka u samom pogledu.

CREATE VIEW pUkupna\_Naplata\_Dodataka

AS

/\*PODUPITI WITH KLAUZULE\*/

WITH

/\*Podupit vraca odabrani broj dodataka za dodatak Dorucak \*/

DodatakDorucak AS(Select OdabraniDodatak.BrojDodataka AS Dorucak,

Dodatak.CenaDodatka AS CenaDorucka,

OdabraniDodatak.UcesnikID#fk,

OdabraniDodatak.DodatakID#fk

FROM OdabraniDodatak JOIN Dodatak

ON(OdabraniDodatak.DodatakID#fk = Dodatak.DodatakID#)

WHERE DodatakID#fk = 'dor'),

/\*Podupit vraca broj odabranih dodataka za dodatak rucak\*/

DodatakRucak AS (Select OdabraniDodatak.BrojDodataka AS Rucak,

Dodatak.CenaDodatka AS CenaRucka,

OdabraniDodatak.UcesnikID#fk,

OdabraniDodatak.DodatakID#fk

FROM OdabraniDodatak JOIN Dodatak ON(OdabraniDodatak.DodatakID#fk = Dodatak.DodatakID#)

WHERE DodatakID#fk = 'ruc'),

/\*Podupit vraca broj odabranih dodataka za dodatakk nocenje \*/

DodatakNocenje AS (Select OdabraniDodatak.BrojDodataka AS Nocenje,

OdabraniDodatak.UcesnikID#fk,

Dodatak.CenaDodatka AS CenaNocenja, /\*Pored svakog broja Odabranih dodataka zelimo da u tabeli moze da se vidi cena dodatka \*/

OdabraniDodatak.DodatakID#fk

FROM OdabraniDodatak JOIN Dodatak

ON(OdabraniDodatak.DodatakID#fk = Dodatak.DodatakID#) /\* Vezujemo tabelu Dodatak sa tabelom Odabrani Dodatak Da bi pristupili ceni dodataka\*/

WHERE DodatakID#fk = 'noc')

--> GLAVNI UPIT

SELECT DISTINCT Ucesnik.UcesnikID#, Ucesnik.Ime, Ucesnik.Prezime,

ISNULL(DodatakDorucak.Dorucak,0) AS Dorucak, /\* Polje predstavlja broj Dorucaka\*/

(ISNULL(DodatakDorucak.CenaDorucka,0) \* ISNULL(Dorucak,0)) AS NaplataDorucka,

ISNULL(DodatakRucak.Rucak,0)AS Rucak, /\* Broj Rucaka \*/

(ISNULL(DodatakRucak.CenaRucka,0) \* ISNULL(Rucak,0)) AS NaplataRucka,/\* Oreginalnu cenu dodatka(NaplataNocenja) mnozimo sa brojem dodataka kolone(Rucak/Dorucak/Nocenje)\*/

ISNULL(DodatakNocenje.Nocenje,0)AS Nocenje,

/\*Vrsimo sumu svih cena svih dodataka mnozene odabranim brojem dodataka za svaki dodatak\*/

(ISNULL(DodatakNocenje.CenaNocenja,0) \* ISNULL(Nocenje,0)) AS NaplataNocenja,

(ISNULL(DodatakDorucak.CenaDorucka,0) \* ISNULL(Dorucak,0) + ISNULL(DodatakRucak.CenaRucka,0) \* ISNULL(Rucak,0) + ISNULL(DodatakNocenje.CenaNocenja,0) \* ISNULL(Nocenje,0)) AS UkupnaNaplata

FROM Ucesnik LEFT JOIN DodatakDorucak

ON (Ucesnik.UcesnikID# = DodatakDorucak.UcesnikID#fk)

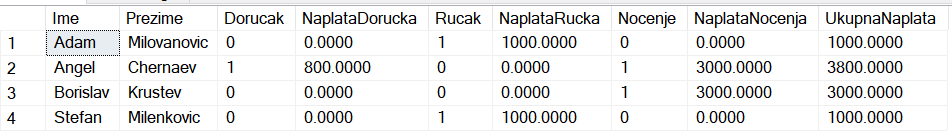
LEFT JOIN DodatakRucak ON(Ucesnik.UcesnikID# = DodatakRucak.UcesnikID#fk)

LEFT JOIN DodatakNocenje ON(Ucesnik.UcesnikID# = DodatakNocenje.UCesnikID#fk)

GO

--> Upit nad pogledom

SELECT \* FROM pUkupna\_Naplata\_Dodataka



Rezultat upita nad Pogledom – “pUkupna\_Naplata\_Dodataka”

## 9.2.4 Pogled

Za Podmodel ‘Registracija’ neophodno je kreirati pogled **pUukupna\_Naplata\_Disciplina** koji prikazuje ime I prezime ucesnika, kao I njihovo ostvareno dugovanje za svaku disciplinu u kojoj se registruju

Ukoliko su ucesnici u ulozi sudije a ne takmicara za datu disciplinu u kojoj se nalaze registrovani, kotizacija te discipline se ne naplacuje I ne uracunava u ukupnu naplatu.

Ovde je neophodno vršiti preklapanje podsistema ‘registracija’ sa tabelom iz podsistema ‘turnir’, kako je prilikom izvršenja naplate neophodno gledati da li su učesnici zavedeni kao takmcari ili sudije za date discipline (kako registrovane sudije ne plaćaju discipline na kojima sude).

USE Swordplay

GO

CREATE VIEW pUkupna\_Naplata\_Disciplina AS

/\*WITH PODUPITI\*/

WITH

DisciplinaRapir AS

(

SELECT Disciplina.CenaDiscip AS Rapir, RegistrovaniUcesnik.UcesnikID#fk

FROM Disciplina JOIN RegistrovaniUcesnik ON (Disciplina.DisciplinaID# = RegistrovaniUcesnik.DisciplinaID#fk)

WHERE Disciplina.DisciplinaID# = 'rap'AND RegistrovaniUcesnik.UcesnikID#fk NOT IN(SELECT Sudija.SudijaID#fk FROM Sudija WHERE Sudija.DisciplinaID#fk = 'rap')

),

DisciplinaLongsword AS

(

SELECT Disciplina.CenaDiscip AS Longsword, RegistrovaniUcesnik.UcesnikID#fk

FROM Disciplina JOIN RegistrovaniUcesnik ON (Disciplina.DisciplinaID# = RegistrovaniUcesnik.DisciplinaID#fk)

WHERE Disciplina.DisciplinaID# = 'lon' AND RegistrovaniUcesnik.UcesnikID#fk NOT IN(SELECT Sudija.SudijaID#fk FROM Sudija WHERE Sudija.DisciplinaID#fk = 'lon')

),

DisciplinaSidesword AS (

SELECT Disciplina.CenaDiscip AS Sidesword, RegistrovaniUcesnik.UcesnikID#fk

FROM Disciplina JOIN RegistrovaniUcesnik ON (Disciplina.DisciplinaID# = RegistrovaniUcesnik.DisciplinaID#fk)

WHERE Disciplina.DisciplinaID# = 'sid' AND RegistrovaniUcesnik.UcesnikID#fk NOT IN(SELECT Sudija.SudijaID#fk FROM Sudija WHERE Sudija.DisciplinaID#fk = 'sid') /\*Sudijama se ne naplacuje kotizacija \*/

)

/\*GLAVNI UPIT POGLEDA\*/

SELECT DISTINCT Ucesnik.UcesnikID#, Ucesnik.Ime, Ucesnik.Prezime,

ISNULL(DisciplinaRapir.Rapir, 0.0000) AS Rapir,

ISNULL( DisciplinaLongsword.Longsword, 0.0000) AS Longsword,

ISNULL(DisciplinaSidesword.Sidesword, 0.0000) AS Sidesword,

(ISNULL(DisciplinaRapir.Rapir, 0.0000) +ISNULL( DisciplinaLongsword.Longsword, 0.0000) + ISNULL(DisciplinaSidesword.Sidesword, 0.0000)) AS UkupnaNaplata

/\*Sabiramo polja cene za Svaku disciplinu i --Pa taj podatak upisujemo u polje UkupnaNaplata\*/

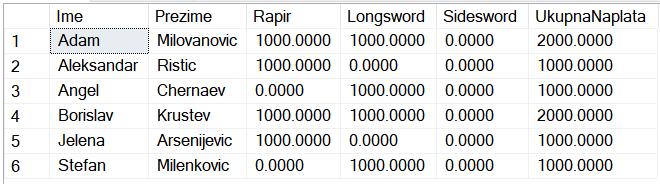
FROM Ucesnik LEFT JOIN DisciplinaRapir ON (Ucesnik.UcesnikID# = DisciplinaRapir.UcesnikID#fk) LEFT JOIN DisciplinaLongsword ON (Ucesnik.UcesnikID# = DisciplinaLongsword.UcesnikID#fk)

LEFT JOIN DisciplinaSidesword ON (Ucesnik.UcesnikID# = DisciplinaSidesword.UcesnikID#fk)

GO

/\*Upit nad pogledom\*/

SELECT\* FROM pUkupna\_Naplata\_Disciplina



Rezultat upita nad pogledom – “pUkupna\_Naplata\_Disciplina”

## 9.1.5 Pogled

Za podmodel ‘registracija’ je neophodno napraviti pogled **pPregled\_Objedinjenih \_Naplata**, koji objedinjuje prvobitne poglede **pUkupna\_Naplata\_Dodataka** **I pUkupna\_Naplata\_Disciplina,** pri čemu je neophodnan prikaz naplate svih dodataka, naplata svih disciplina kao I objedinjena naplatu troškova turninra za učesnika(Discipline + Dodaci).

USE Swordplay

GO

CREATE VIEW pPregled\_Objedinjenih\_Naplata AS

SELECT Ucesnik.UcesnikID#, Ucesnik.Ime, Ucesnik.Prezime,

ISNULL(Dodaci.UkupnaNaplata,0) AS NaplataDodataka,

ISNULL(Discipline.UkupnaNaplata, 0) AS NaplataDisciplina,

(ISNULL(Dodaci.UkupnaNaplata,0) + ISNULL(Discipline.UkupnaNaplata, 0)) ObjedinjenaNaplata

FROM pUkupna\_Naplata\_Dodataka AS Dodaci FULL JOIN pUkupna\_Naplata\_Disciplina AS Discipline

ON( Dodaci.UcesnikID# = Discipline.UcesnikID#)

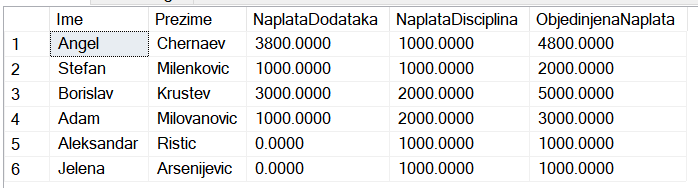
RIGHT OUTER JOIN Ucesnik ON( Discipline.UcesnikID# = Ucesnik.UcesnikID#)

GO

--Upit nad pogledom

SELECT \*

FROM pPregled\_Objedinjenih\_Naplata



Rezultat upita nad pogledom

# 9.2 Podsistem Turnir

## 9.2.1 Procedura

--Organizatori, pre naplate usluga turnira prisutnim učesnicima, imaju potrebu na na dan turnira zavedu učesnike turnira kao takmičare ili sudije prateci licni potpis učesnika u izjašnjenju da je spreman da učestvuje u jednoj ili više disciplina u kapacitetu takmičara Ili sudije za odabrane discipline.

Neophodno je realizovati ovu baznu proceduru kao preduslov transakcije podsistema ‘turnir’ koja će se definisati u nastavku.

USE Swordplay

GO

CREATE Procedure kdpUnosTakmicara\_Sudija

@SifraUcesnika CHAR(3),

@SifraDiscipline CHAR(3),

@SudijaTakmicar VARCHAR(10)

AS

BEGIN TRY

SET XACT\_ABORT ON /\*Automatsko sprecavanje daljeg izvrsenja naredbi ukoliko dodje do greske\*/

/\*Provera da li postoji registrovani ucesnik u bazi\*/

IF (SELECT UcesnikID# FROM Ucesnik WHERE UcesnikID# = @SifraUcesnika) IS NULL

BEGIN

PRINT'Ne postoji registrovani ucesnik sa tom sifrom';

END

/\*Provera da li postoji parametar discipline u bazi\*/

IF @SifraDiscipline NOT IN (SELECT DisciplinaID# FROM Disciplina)

BEGIN

PRINT 'Odabrana sifra Discipline ne odgovara postojecim disciplinama turnira';

END

/\*Provera da li Registrovani ucesnik vec postoji zaveden kao takmicar\*/

IF @SifraUcesnika IN(SELECT TakmicarID#fk FROM Takmicar WHERE DisciplinaID#fk = @SifraDiscipline)

BEGIN

PRINT 'Ucesnik je vec zaveden kao takmicar za datu disciplinu';

END

/\*Provera da li da li Registrovani ucesnik vec postoji zaveden kao Sudija \*/

IF @SifraUcesnika IN (SELECT SudijaID#fk FROM Sudija WHERE DisciplinaID#fk = @SifraDiscipline)

BEGIN

PRINT 'Ucesnik je vec zaveden kao sudija za datu disciplinu';

END

BEGIN /\*Zapocinjemo niz naredbi\*/

BEGIN TRANSACTION /\*Zapocinjemo eksplicitnu transakciju\*/

/\*Azuriranje tabela Takmicar/Sudija u zavisnosti od unetog parametra @SudijaTakmicar\*/

/\* Proveravamo da li korisnik u pozivu procedure zeli da smesti novog takmicara ili sudiju \*/

IF @SudijaTakmicar = 'takmicar'

BEGIN

INSERT INTO Takmicar(TakmicarID#fk, DisciplinaID#fk)

VALUES (@SifraUcesnika,@SifraDiscipline);

PRINT 'Unet je novi Takmicar';

END

IF @SudijaTakmicar = 'sudija'

BEGIN

INSERT INTO Sudija(SudijaID#fk, DisciplinaID#fk)

VALUES(@SifraUcesnika, @SifraDiscipline)

PRINT'Unet je novi Sudija';

END

IF @SudijaTakmicar != 'sudija' AND @SudijaTakmicar != 'Takmicar'

BEGIN

PRINT'Morate navesti sudiju ili takmicara kao parametar zavodjenja registrovanog ucesnika u turnir'

END

/\*Potvrdjujemo unete promene\*/

COMMIT TRANSACTION

END /\*Zatvaramo niz naredbi\*/

END TRY

BEGIN CATCH

BEGIN

PRINT 'Doslo je do pojave greske!';

PRINT 'Proverite ulazne parametre';

PRINT ERROR\_MESSAGE();

ROLLBACK TRANSACTION;

PRINT 'Procedura je ponistila promene';

END

END CATCH

GO

--Poziv procedure za unos Takmicara ili Sudije

--Unos sudija

--Sudije za rapir

EXEC kdpUnosTakmicara\_Sudija

@SifraUcesnika = '006',

@SifraDiscipline ='rap',

@SudijaTakmicar = 'sudija'

GO

EXEC kdpUnosTakmicara\_Sudija

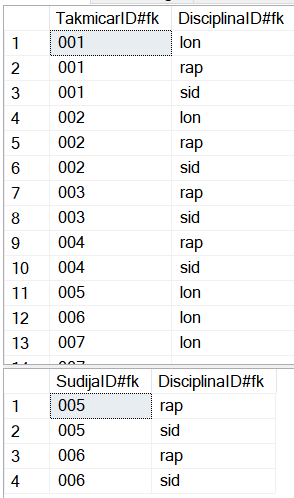
@SifraUcesnika = '001',

@SifraDiscipline ='rap',

@SudijaTakmicar = 'takmicar'

GO

/\* . . . \*/



Rezultat azuriranja tabela procedurom – “kdpUnosTakmicara\_Sudija”

## 9.3 Integrisani Model Mačevalački Turnir

### 9.3.1 Bazna Funkcija

Kako se svakom prisutnom registrovanom učesniku na licu mesta vrši naplata disciplina i dodataka, neophodno je krerirati vidljivo stanje naplata po učesniku pomoću kojeg volonteri na prijemu učesnika mogu da vode evidenciju o naplatama i dugovanjima bilo kojeg pojedinačnog učesnika.

U tom cilju je potrebno je kreirati baznu funkciju **kdfPrikaz\_Racuna**(prefiks kdf – korisčničiki definisana funkcija) koja će kao argument prihvatiti šifru učesnika a vratiti tabelu gde se nalazi , broj dodataka, cena dodataka, disciplina I ukupna naplata za sve odabrane dodatke I discipline za navedenu šifru učesnika.

Ova bazna funkcija za svoju realizaciju pozajmljuje tabele iz oba podmodela, budući da koristi pogled ’**pUukupna\_Naplata\_Disciplina’**, koji objedinjuje tabele ‘Sudija’ I ‘Takmičar’ sa podmodelom ‘registracija’

U nastavku sledi programski kod za kreiranje bazne funkcije. Kao sto moze da se primeti, funkcija koristi predhodno definisane poglede **pUkupna\_Naplata\_Dodataka, pUukupna\_Naplata\_Disciplina** kao I **pPregled\_Objedinjenih \_Naplata**.

CREATE FUNCTION kdfPrikaz\_Racuna(@SifraUcesnika CHAR(3))

RETURNS TABLE

AS

RETURN SELECT

Objedinjeno.UcesnikID#,

Objedinjeno.Ime,

Objedinjeno.Prezime,

Dodaci.Dorucak,

Dodaci.NaplataDorucka,

Dodaci.Rucak,

Dodaci.NaplataRucka,

/\*(Dodaci.NaplataRucka / Dodaci.Rucak) AS Cena\_Jednog\_Rucka,\*/

Dodaci.Nocenje,

Dodaci.NaplataNocenja,

Discipline.Rapir,

Discipline.Longsword,

Discipline.Sidesword,

Discipline.UkupnaNaplata AS KotizacijaDisciplina,

Objedinjeno.ObjedinjenaNaplata

FROM pPregled\_Objedinjenih\_Naplata AS Objedinjeno

JOIN pUkupna\_Naplata\_Dodataka AS Dodaci

ON (Objedinjeno.UcesnikID# = Dodaci.UcesnikID#)

JOIN pUkupna\_Naplata\_Disciplina AS Discipline

ON(Discipline.UcesnikID# = Objedinjeno.UcesnikID#

AND Discipline.UcesnikID# = Dodaci.UcesnikID#)

WHERE Objedinjeno.UcesnikID# = @SifraUcesnika

GO

-- Poziv funkcije

/\*Sudije ne placaju kotizaciju turnira u kojima sude (U ovom slucaju, 001 i 002 sude za disciplinu rapir\*/

SELECT \* FROM kdfPrikaz\_Racuna('001');

SELECT \* FROM kdfPrikaz\_Racuna('002');

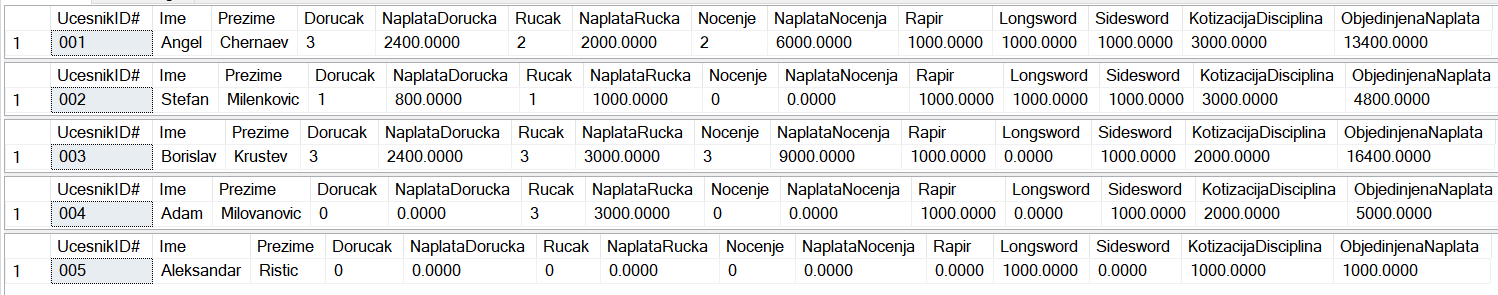
/\*Takmicari (Placaju punu kotizaciju za svaku disciplinu az koju su se registrovali \*/

SELECT \* FROM kdfPrikaz\_Racuna('003');

SELECT \* FROM kdfPrikaz\_Racuna('004');

SELECT \* FROM kdfPrikaz\_Racuna('005');

SELECT \* FROM kdfPrikaz\_Racuna('006');



Redovi koji vracaju uzastopni pozivi funkcije - “kdfPrikaz\_Racuna”

# 9.2.2 Poslovna transakcija I bazna procedura podsistema Turnir

Podsistem Turnir je predvidjen da sadrži procedure I poglede neophodne za uspešno praćenje odvijanja turnira, rezultata mečeva, statistike učesnika svih eliminacionih mečeva po fazama I disciplinama, kao I nalaženje prva tri mesta za bilo koju od tri discipline.

Organizatori turnira imju potrebu da čuvaju podatke o odrzanim mečevima I statistikama iz istih meceva. Buduci raspored I uparenje takmičara po mečevima se preko procedure pravovremeno unosi u bazu, dok se za one tekuce mečeve belezi I baznom procedurom unosi statistika ostvarenih poena svakog takmičara, kao I broj izazova sudijinih odluka.



Slika3. Primer Eliminacionog turnira iz discipline ‘Rapir’

## 9.2.3 Bazna Procedura

Neophodno je kreirati baznu proceduru **kdpUnos\_Novog\_Meca** kojom ce na turniru po uparivanju takmicara da se kreiraju mecevi sa datim takmicarima u disciplini tekuceg turnira. Procedura prihvata parametre za unos sifre plavog I crvenog boraca, sifre prvog I drugog sudije, kao I sifru discipline. Pomocu datih parametara, procedura vezuje tabele ‘FazaTakmičenja’, ‘Takmičar’, ‘Sudija’ I ‘Disciplina’ u jednoj novoj instanci tabele ‘Meč’.

CREATE PROCEDURE kdpUnos\_Novog\_Meca

--potrebni parametri

@redbr int,

@discID CHAR(3),

@plaviID CHAR(3),

@crveniID CHAR(3),

@sud1ID CHAR(3),

@sud2ID CHAR(3),

@fazaID CHAR(6)

AS

IF @redbr IN (SELECT RedniBrojMeca# FROM Mec)

BEGIN

PRINT 'Redni broj meca vec postoji u bazi';

END

IF @discID NOT IN (SELECT DisciplinaID# FROM Disciplina)

BEGIN

PRINT 'Navedena disciplina nije deo ovog turnir';

END

BEGIN TRY

/\*Proveravamo da li su uneseni ucesnici i sudije registrovani na disciplinu meca\*/

IF (@plaviID IN(SELECT ucesnikID#fk FROM RegistrovaniUCesnik WHERE disciplinaID#fk = @discID)

AND @crveniID IN (SELECT ucesnikID#fk FROM RegistrovaniUCesnik WHERE disciplinaID#fk = @discID)

AND @sud1ID IN (SELECT ucesnikID#fk FROM RegistrovaniUCesnik WHERE disciplinaID#fk = @discID)

AND @sud2ID IN (SELECT ucesnikID#fk FROM RegistrovaniUCesnik WHERE disciplinaID#fk = @discID))

BEGIN

SET XACT\_ABORT ON

BEGIN TRANSACTION

INSERT INTO Mec(RedniBrojMeca#,

DisciplinaID#fk,

PlaviBoracID#fk,

PlaviDisciplinaID#fk,

CrveniBoracID#fk,

CrveniDisciplinaID#fk,

Sudija1ID#fk,

Sudija1DisciplinaID#fk,

Sudija2ID#fk,

Sudija2DsciplinaID#fk,

FazaID#fk)

/\*Za disciplinu svakog ucesnika meca unosimo isti parametar\*/ VALUES

(@redbr, @discID,@plaviID,

@discID, /\*disciplina plavog \*/

@crveniID,

@discID,

@sud1ID,

@discID,

@sud2ID, /\*disciplina drugog sudije\*/

@discID,

@fazaID)

COMMIT TRANSACTION

END

ELSE PRINT'Neki od navedenih ucesnika nisu registrovani za navedenu disciplinu';

END TRY

BEGIN CATCH

PRINT 'Doslo je do pojave greske!';

PRINT'-----Proverite ulazne parametre -------';

PRINT ERROR\_MESSAGE();

/\*Ponistavamo sve promene insert naredbi do kojih je doslo pre greske\*/

ROLLBACK TRANSACTION

PRINT 'Procedura je ponistila promene';

END CATCH;

GO

/\*Pozivi procedure \*/

/\* UNOS ELIMINACIONOG TURNIRA DISCIPLINE RAPIR \*/

EXEC kdpUnos\_Novog\_Meca

@redBr = 1,

@discID = 'rap',

@plaviID = '007',

@crveniID = '008',

@sud1ID = '005',

@Sud2ID = '006',

@fazaID = 'grup'

EXEC kdpUnos\_Novog\_Meca

@redBr = 2,

@discID = 'rap',

@plaviID = '009',

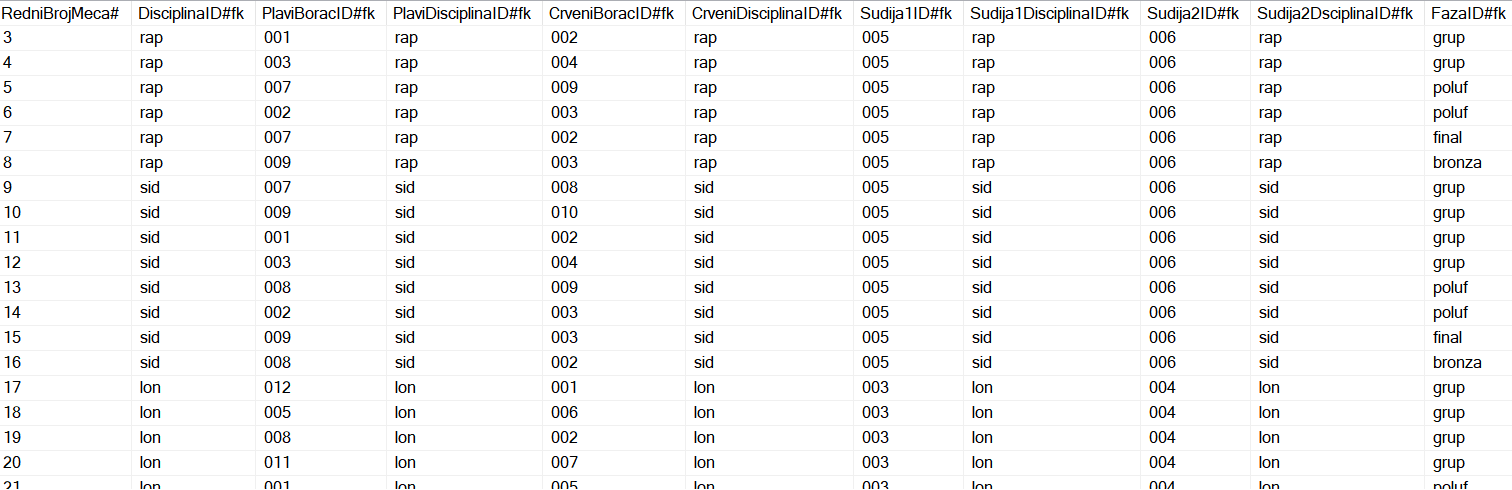
@crveniID = '010',

@sud1ID = '005',

@Sud2ID = '006',

@fazaID = 'grup'

/\* . . . \*/



Prikaz azuriranog sadrzaja tabele ‘Mec’ preko procedure – “kdpUnos\_Novog\_Meca”

## 9.2.4 Bazna Procedura

Za potrebe praćenja meceva u podmodelu ‘Turnir’ je neophodno krerirati baznu proceduru ‘**kdpUnos\_Statistike\_Takmicara’** pomoću koje će da se zavode rezultati pojedinačnih takmičara za unete mečeve. Procedura kao parametre prima broj ostvarenih pogodaka I broj poena za svaki pogodak (jedan, dva, tri, četir), takodje u proceduru se unosi broj izazova (ako ih ima), disciplina turnira kao I broj meca I šifra učesnika za kojeg se zavodi statistika.

USE Swordplay

Go

CREATE PROCEDURE kdpUnos\_Statistike\_Takmicara

--potrebni parametri

@redbrMec int, /\*redni broj meca\*/

@discID CHAR(3), /\*null polja, svakako nece proci IF grane u proceduri \*/

@takID CHAR(3), /\*koje traze par ID podataka u odgovarajucoj tabeli\*/

@jedanP INT = 0, /\*jedan poen,pa dva,pa tri,pa cetiri\*/

@dvaP INT = 0,

@triP INT = 0, /\*0 - jer neki poeni nece nuzno biti ostvareni u toku meca\*/

@cetiriP INT = 0,

@brIzazova INT,

@ishod CHAR(6) /\*ishod meca(pobeda/poraz)\*/

AS

DECLARE @ukupnoPoena INT = 0;

SET @ukupnoPoena = @jedanP + @dvaP\*2 + @triP\*3 + @cetiriP\*4;

IF @redbrMec NOT IN (SELECT RedniBrojMeca# FROM Mec)

BEGIN

PRINT 'Redni broj meca vec postoji u bazi'; --prvo postoji li mec

END

ELSE

IF @discID NOT IN (SELECT DisciplinaID#fk FROM Mec WHERE Mec.RedniBrojMeca# = @redbrMec)

BEGIN

PRINT 'Navedena disciplina nije deo navedenog meca'; --onda provera da neko nije greskom ubacio pogresnu disciplinu za postojeci mec

END

ELSE

IF @takID NOT IN (SELECT PlaviBoracID#fk FROM Mec) AND @takID NOT IN (SELECT CrveniBoracID#fk FROM Mec)

BEGIN

PRINT 'Navedeni takmicar nije ucesnik navedenog meca';

END

BEGIN TRY

SET XACT\_ABORT ON

INSERT INTO Statistika(RedniBrojMeca#fk,

TakmicarID#fk,

DisciplinaID#fk,

JedanPoen,

DvaPoena,

TriPoena,

CetiriPoena,

UkupnoPoena,

BrojIzazova,

IshodMeca )

VALUES

(@redbrMec, --Redni broj meca

@takID,

@discID,

@jedanP, --jedan poen,pa dva,pa tri,pa cetiri

@dvaP,

@triP,

@cetiriP,

@ukupnoPoena,

@brIzazova, --broj izazova

@ishod

)

END TRY

BEGIN CATCH

PRINT 'Doslo je do greske!';

PRINT'-------PROVERITI ULAZNE PARAMETRE--------';

PRINT 'Informacije o gresci ' + ERROR\_MESSAGE();

END CATCH

Go

/\*Unos statistike za odigrane meceve\*/

/\*Mec 1\*/

EXEC kdpUnos\_Statistike\_Takmicara

@discID ='rap', @redbrMec = 1,

@takID = '007', @jedanP = 0,

@dvaP = 1, @triP = 1,

@cetiriP = 1, @brIzazova = 0,

@ishod = 'pobeda'

EXEC kdpUnos\_Statistike\_Takmicara

@discID ='rap', @redbrMec = 1,

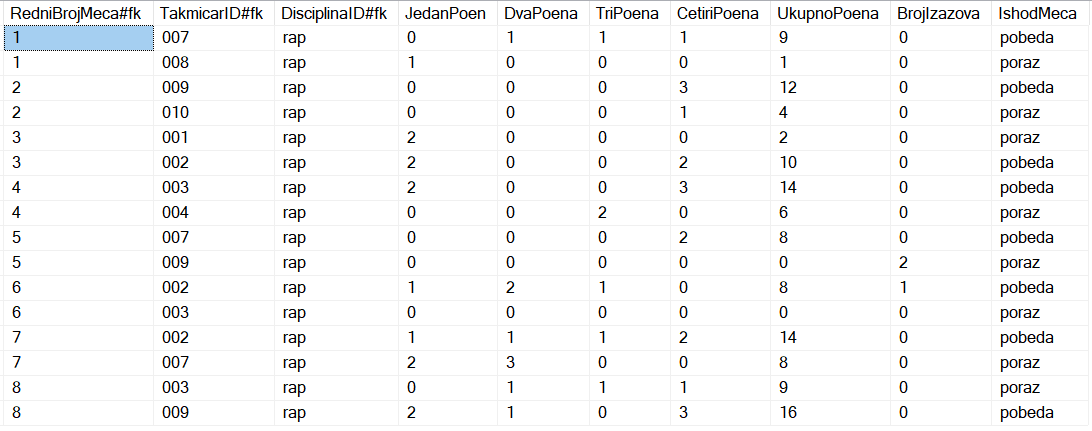
@takID = '008', @jedanP = 1,

@dvaP = 0, @triP = 0,

@cetiriP = 0, @brIzazova = 0,

@ishod = 'poraz'

/\*. . .\*/



Rezultat azuriranja tabela pozivom procedure –‘kdpUnos\_Statistike\_Takmicara

## 9.2.5 Pogled

Za podmodel Turnir, neophodno je kreirati pogled **pRezultati\_Meceva** (prefiks p za pogled) kroz koji ce moci da se vide podaci o takmicarima za svaki odrzani mec, pri cemu se ispisuje ime I prezime takmicara, klub kojem pripadaju, broj ostvarenih poena u mecu kao I ishod meca za svakog takmicara u mecu. Za meceve je potrebno prikazivati u kojoj su se fazi turnira odrzali, kao I naziv discipline turnira kojoj odigrani mec pripada.

CREATE VIEW pRezultati\_Meceva AS

WITH

--Prva izvedena tabela vrsi selekciju podataka o svim Takmicarima kao i njihovoj statistici za one meceve u kojima su bili pobednici

Pobednik AS

(Select Ucesnik.UcesnikID#, Ucesnik.Ime + Ucesnik.Prezime AS Pobednik,Klub.NazivKluba, Statistika.UkupnoPoena,Statistika.JedanPoen, Statistika.DvaPoena,

Statistika.TriPoena, Statistika.CetiriPoena, Statistika.IshodMeca, Statistika.BrojIzazova, Statistika.RedniBrojMeca#fk, Statistika.DisciplinaID#fk FazaID#fk

FROM Klub JOIN Ucesnik ON(Klub.KlubID# = Ucesnik.KlubID#fk)

JOIN Statistika ON(Ucesnik.UcesnikID# = Statistika.TakmicarID#fk)

JOIN Mec ON(Statistika.RedniBrojMeca#fk = Mec.RedniBrojMeca#)

WHERE ishodMeca = 'pobeda'

),

--Druga izvedena tabela vrsi selekciju podataka o svim Takmicarima kao i njihovoj statistici za one meceve u kojima su bili gubitnici

Gubitnik AS

(Select Ucesnik.UcesnikID#, Ucesnik.Ime + Ucesnik.Prezime AS Gubitnik, Klub.NazivKluba, Statistika.UkupnoPoena, Statistika.JedanPoen, Statistika.DvaPoena,

Statistika.TriPoena, Statistika.CetiriPoena, Statistika.IshodMeca, Statistika.BrojIzazova, Statistika.RedniBrojMeca#fk, Statistika.DisciplinaID#fk, FazaID#fk

FROM Klub JOIN Ucesnik ON(Klub.KlubID# = Ucesnik.KlubID#fk)

JOIN Statistika ON(Ucesnik.UcesnikID# = Statistika.TakmicarID#fk)

JOIN Mec ON(Statistika.RedniBrojMeca#fk = Mec.RedniBrojMeca#)

WHERE ishodMeca = 'poraz'

)

SELECT

Pobednik.UcesnikID# AS Sifra\_Pobednika,

Pobednik.RedniBrojMeca#fk As Broj\_Meca,

Pobednik.Pobednik, -->Ime i prezime pobednika

Pobednik.NazivKluba AS Pob\_Klub,

Pobednik.JedanPoen AS Pob\_1pt,

Pobednik.DvaPoena AS Pob\_2pt,

Pobednik.TriPoena AS Pob\_3pt,

Pobednik.CetiriPoena AS Pob\_4pt,

Pobednik.UkupnoPoena AS Pob\_Ukupno\_Poena,

Pobednik.BrojIzazova AS Pob\_Broj\_Izazova,

Gubitnik.UcesnikID# AS Sifra\_Gubitnika,

Gubitnik.Gubitnik, -->Ime i prezime gubitnika

Gubitnik.NazivKluba AS Gub\_Klub,

Gubitnik.JedanPoen AS Gub\_1pt,

Gubitnik.DvaPoena AS Gub\_2pt,

Gubitnik.TriPoena AS Gub\_3pt,

Gubitnik.CetiriPoena AS Gub\_4pt,

Gubitnik.UkupnoPoena AS Gub\_Ukupno\_Poena,

Gubitnik.BrojIzazova AS Gub\_Broj\_Izazova,

Gubitnik.FazaID#fk AS Faza\_Takmicenja,

Gubitnik.DisciplinaID#fk AS Disciplina\_Turnira

FROM Pobednik JOIN Gubitnik ON (Pobednik.RedniBrojMeca#fk = Gubitnik.RedniBrojMeca#fk)

GO

-- Upit nad pogledom

SELECT Broj\_Meca, Pobednik, Pob\_Klub, Pob\_Ukupno\_Poena, Gubitnik, Gub\_Klub, Gub\_Ukupno\_Poena, Faza\_Takmicenja, Disciplina\_Turnira

FROM pRezultati\_Meceva



Pogled – “pRezultati\_ Meceva”

## 9.2.6 Bazna Funkcija

Za potrebe turnira neophodno je kreirati baznu funkciju koja će vratiti tabelu sa redovima prva tri mesta osvojena na disciplini turnira. Bazna funkcija koja ima samo jedan parametar koji prihvata šifru discipline za koju vraća rezultat prva tri osvojena mesta.

CREATE FUNCTION kdfMedalje\_Po\_Disciplini(@SifraDiscipline CHAR(3))

RETURNS TABLE

RETURN

WITH

/\*Zlato dobijamo tako sto nalazimo pobednika za finale-fazu takmicenja za datu disciplinu \*/

Zlato AS (SELECT TOP(1) Ucesnik.Ime, Ucesnik.Prezime, UcesnikID#

From Ucesnik Join Mec ON(Ucesnik.UcesnikID# = Mec.PlaviBoracID#fk OR Ucesnik.UcesnikID# = Mec.CrveniBoracID#fk)

JOIN Statistika ON(Statistika.RedniBrojMeca#fk = Mec.RedniBrojMeca# AND Statistika.TakmicarID#fk = Ucesnik.UcesnikID#)

WHERE Mec.FazaID#fk = 'final' AND Statistika.IshodMeca = 'pobeda' AND Mec.DisciplinaID#fk = @SifraDiscipline),

/\*Srebro dobijamo tako sto nalazimo gubitnika u za finale-fazu takmicenja date discipline \*/

SREBRO AS (SELECT TOP(1) Ucesnik.Ime, Ucesnik.Prezime, UcesnikID#

From Ucesnik Join Mec ON(Ucesnik.UcesnikID# = Mec.PlaviBoracID#fk OR Ucesnik.UcesnikID# = Mec.CrveniBoracID#fk)

JOIN Statistika ON(Statistika.RedniBrojMeca#fk = Mec.RedniBrojMeca# AND Statistika.TakmicarID#fk = Ucesnik.UcesnikID#)

WHERE Mec.FazaID#fk = 'final' AND Statistika.IshodMeca = 'poraz' AND Mec.DisciplinaID#fk = @SifraDiscipline ),

/\*Bronzu dobijamo tako sto nalazimo pobednika za 'bronza-mec' fazu takmicenja date discipline \*/

Bronza AS (SELECT TOP(1) Ucesnik.Ime, Ucesnik.Prezime, UcesnikID#

From Ucesnik Join Mec ON(Ucesnik.UcesnikID# = Mec.PlaviBoracID#fk OR Ucesnik.UcesnikID# = Mec.CrveniBoracID#fk)

JOIN Statistika ON(Statistika.RedniBrojMeca#fk = Mec.RedniBrojMeca# AND Statistika.TakmicarID#fk = Ucesnik.UcesnikID#)

WHERE Mec.FazaID#fk = 'bronza' AND Statistika.IshodMeca = 'pobeda' AND Mec.DisciplinaID#fk = @SifraDiscipline )

SELECT Zlato.ime AS Zlato\_Ime, Zlato.Prezime AS Zlato\_Prezime, Zlato.UCesnikID# AS Zlato\_ID,

Srebro.ime AS Srebro\_Ime, Srebro.Prezime AS Srebro\_Prezime, Srebro.UCesnikID# AS Srebro\_ID,

Bronza.ime AS Bronza\_Ime, Bronza.Prezime AS Bronza\_Prezime, Bronza.UCesnikID# AS Bronza\_ID

FROM Zlato, Srebro, Bronza

--Poziv funkcije preko upita

SELECT \*

FROM kdfMedalje\_Po\_Disciplini('rap')



SlikaN+1. Rezultat poziva funkcije za Turnir discipline rapir

## 9.2.7 Bazna Funkcija

Organizatorima turnira je neophodno da imaju pregled celokupne statistike pojedinačnog učesnika za celo trajanje turnira. Zato je neophodno kreirati funkciju **kdfStatistika\_Po\_Ucesniku,** pri cemu bi funkcija vraćala tabelu koja prikazuje sve relevantne statističke podatke o učesniku I njegovim mečevima za vreme turnira. Funkcija se realizuje koristeci pogled ‘**pRezultati\_Meceva**’.

CREATE FUNCTION kdfStatistika\_Po\_Ucesniku(@SifraUcesnika CHAR(3))

RETURNS TABLE

RETURN

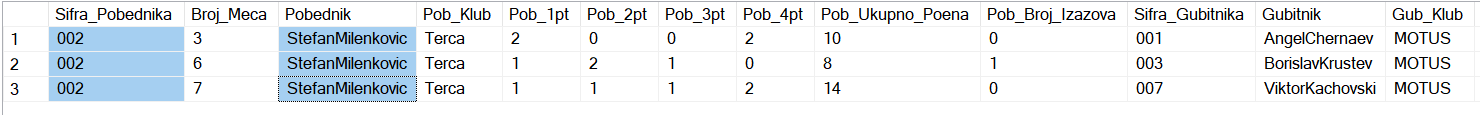
SELECT \*

FROM pRezultati\_Meceva AS mecevi

WHERE mecevi.Sifra\_Gubitnika = @SifraUcesnika OR mecevi.Sifra\_Pobednika = @SifraUcesnika

SELECT \*

FROM kdfStatistika\_Po\_Ucesniku('002')



SlikaN+2. Rezultat poziva funkcije ‘kdfStatistika\_Po\_Ucesniku’

# 10. Literatura

1. Majstorović M.: Baye podataka, skripta, Visoka škola strukovnih studija za informacione tehnologije, Beograd, 2013.
2. M. Majstorović, Uvod u baze podataka Visoka škola strukovnih studija za informacione tehnologije, Beograd, 2015, ISBN 978-86-89007-03-9
3. Majstorović M.: Napredne baze podataka, ITS, Beograd, 2017, ISBN 978-86-89007-19-0.
4. Majstorović M.: Baze podataka, predavanja I vežbe za predmet u ppt formi, školska 2019/2020.(na studentskom portalu).