SQL
Baze podataka 1

dr Miloš CVETANOVIĆ

SQL

- Nije samo upitni jezik
- Rane 1970-te IBM System R, Sequel
- ANSI : SQL-86, SQL-89, SQL-92, SQL:1999, SQL:2003, SQL:2006, SQL:2008, SQL:2011, SQL2016
- SQL delovi:
    1. Data-definition language (DDL)
    2. Data-manipulation language (DML)
    3. DCL, PSM …
- SQL-92 i po nešto SQL:2003

Fudbalski savez

Posmatrani sistem je fudbalski savez koji evidentira utakmice odigrane u toku jedne sezone. Za svaku utakmicu se trajno čuvaju podaci o tome koji timovi su igrali, u kom kolu je ta utakmica odigrana (i koje godine), kakav je ishod utakmice bio, koji su sve igrači igrali i na kojim pozicijama. Za svakog fudbalera se pamti ime i tim za koji igra, dok se za timove pamti naziv i mesto iz koga dolaze. Pretpostavka je da fudbaleri ne mogu da menjaju tim u kome igraju, u toku sezone. Takođe je potrebno trajno evidentirati svaki postignuti gol kao i to koji fudbaler ga je postigao, u kom minutu i koji je to gol bio po redu na posmatranoj utakmici. Pored golova trajno se evidentiraju i svi kartoni, žuti i crveni, dodeljeni na utakmici i to kom fudbaleru i u kom minutu.

DDL

- Definisanje tabela (tipova atributa, integritetska ograničenja, indeksa, fizičkog rasporeda, …)
- Definisanje
- Osnovni tipovi:
    1. char(n)
    2. varchar(n) - maksimalna dužina? nvarchar?
    3. int - zavisi od mašine
    4. smallint
    5. numeric(p,d) - realna vrednost sa p cifara od kojih je d iza zapete
    6. real, double precizion - pokretna zapeta
    7. float(n) - pokretna zapeta sa preciznošću od najmanje n cifara
    8. boolean
    9. timestamp – takođe i date, time
- Svakom domenu pripada i NULL

DDL

CREATE TABLE Tabela (DefinicijaKolone , … [OgranicenjeTabele , …]) ;

DefinicijaKolone ::= Kolona { Tip | Domen } OgranicenjeKolone …

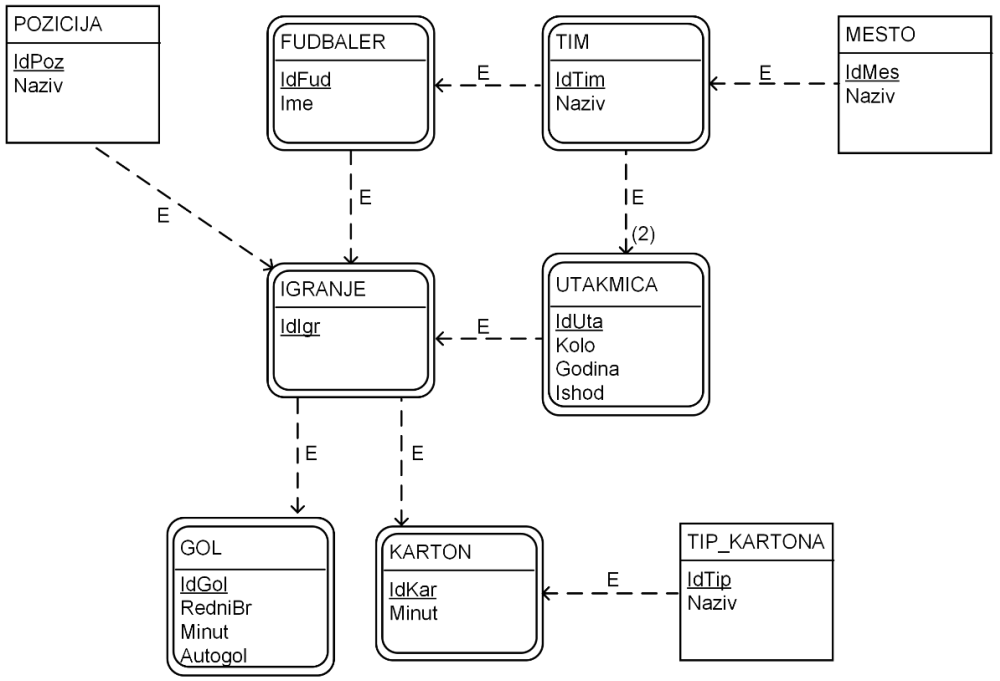
NOT NULL
UNIQUE
PRIMARY KEY
CHECK (Predikat)
DEFAULT = Const

REFERENCES Tabela [(Kolona)]
[ON UPDATE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
[ON DELETE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]

UNIQUE (Kolona , …)
PRIMARY KEY (Kolona , …)
CHECK (Predikat)
FOREIGN KEY (Kolona , …)
REFERENCES Tabela [(Kolona , …)]
[ON UPDATE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]
[ON DELETE {NO ACTION|CASCADE|SET NULL|SET DEFAULT}]

DDL

Sastaviti SQL skript kojim se formira tabela UTAKMICA, ukoliko je poznato da utakmicu igraju dva različita tima čije se šifre nalaze u tabeli TIM, da ishod utakmice može biti iz skupa vrednosti {X-nerešeno, 1-pobeda domaćih, 2-pobeda gostiju} i da postoji svega 30 kola u kojima se utakmice igraju. U toku sezone svaki par timova odigra dve utakmice, pri čemu je jedna na domaćem, a druga na gostujućem terenu.




```
CREATE TABLE UTAKMICA (
    IdUta INT PRIMARY KEY,
    Kolo INT NOT NULL CHECK (Kolo BETWEEN (1 and 30),
    Godina INT NOT NULL CHECK (Godina > 2020),
    Ishod CHAR NOT NULL CHECK (Ishod IN ( '1', '2', 'X')),
    IdTDomaci INT NOT NULL REFERENCES TIM(IdTim) ON UPDATE CASCADE,
    IdTGost INT NOT NULL REFERENCES TIM(IdTim) ON UPDATE CASCADE,
    CHECK (IdTDomaci <> IdTGost),
    UNIQUE (IdTDomaci, IdTGost)
);
```

DDL

NaredbaIzmeneTabele ::= ALTER TABLE Tabela SpecIzmeneTabele ;
SpecIzmeneTabele ::= SpecIzmeneKolona | SpecIzmeneOgranicenjaTabele
SpecIzmeneKolona ::= SpecIzmeneJedneKolone , …
SpecIzmeneJedneKolone ::= DodavanjeKolone | IzmenaKolone | UklanjanjeKolone
DodavanjeKolone ::= ADD [COLUMN] DefinicijaKolone
IzmenaKolone ::= DodavanjePodrazumevanja | UklanjanjePodrazumevanja
DodavanjePodrazumevanja ::= ALTER [COLUMN] Kolona SET DEFAULT = Const
UklanjanjePodrazumevanja ::= ALTER [COLUMN] Kolona DROP DEFAULT
UklanjanjeKolone ::= DROP [COLUMN] Kolona { RESTRICT | CASCADE }
SpecIzmeneOgranicenjaTabele ::= SpecIzmeneJednogOgranicTabele , …
SpecIzmeneJednogOgranicTabele ::= SpecDodavanjaOgranicenja | SpecUklanjanjaOgranicenja
SpecDodavanjaOgranicenja ::= ADD CONSTRAINT OgranicenjeTabele
SpecUklanjanjaOgranicenja ::= DROP CONSTRAINT { RESTRICT | CASCADE }

NaredbaUklanjanjaTabele ::= DROP TABLE Tabela ;

DML

- Dohvatanje podataka (upiti)
- Izmena podataka (dodavanje, izmena, brisanje)

SELECT FROM WHERE GROUP BY HAVING

- FROM određuje odakle se redovi dohvataju
- WHERE određuje redove koji će biti u rezultatu ili koji će biti grupisani
- GROUP BY određuje koji redovi ulaze u istu grupu (odnosno po kojim kolonama se grupiše)
- HAVING određuje grupe koje će biti u rezultatu, a na osnovu:
    1. jedne ili više kolona po kojima se grupiše
    2. agregiranih vrednosti nad jednom ili više ostalih kolonama (po kojima se ne grupiše)
        agregirane vrednosti:
        agregacija vrednosti kolona nad svim redovima koji se nalaze u grupi

- ORDER BY - deo kursora?

DML

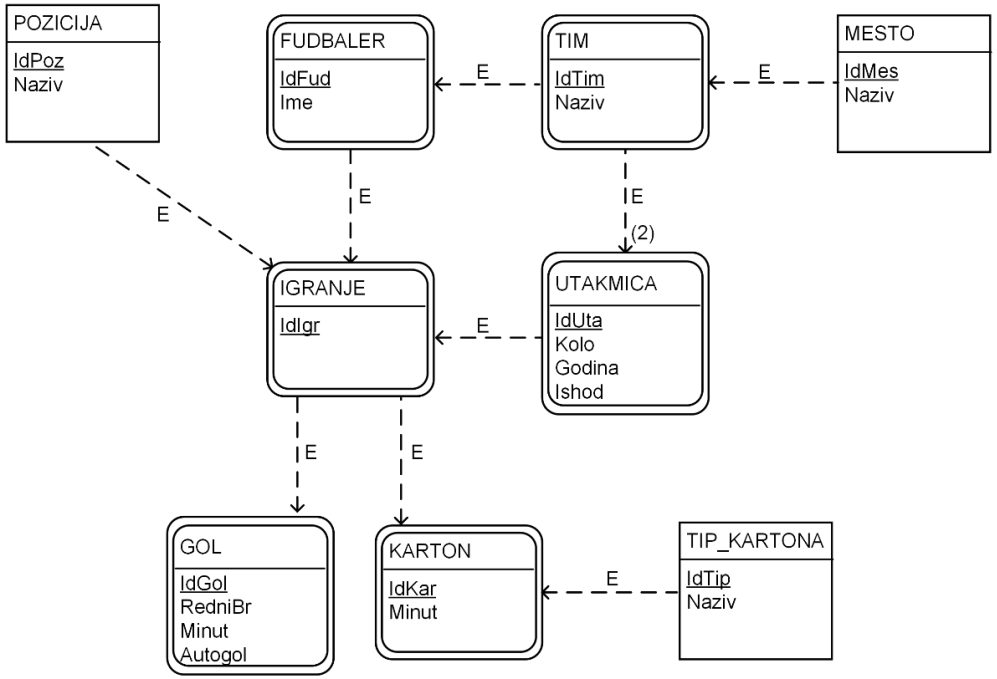
SELECT S-ListaKolona [, G-Lista]
FROM Tabela | Pogled
[WHERE R-Predikat]
[GROUP BY G-ListaKolona [HAVING G-Predikat]] ;

S-ListaKolona ::= Kolona | * | { [ALL | DISTINCT] R-Izraz , …}
R-Izraz ::= Kolona | Kolona Operator Kolona | R-Izraz | SlozenPredikat
Operator ::= - operator nad kolonama (npr. algebarski za brojne vrednosti)
R-Predikat ::= R-Izraz { <|>|=|<=|>=|<> } R-Izraz | SlozenPredikat
G-Lista ::= G-Izraz , …
G-ListaKolona ::= Kolona , …
SlozenPredikat ::= [NOT] R-Predikat [AND | OR SlozenPredikat]
R-Izraz { <|>|=|<=|>=|<> } R-Izraz
R-Izraz [NOT] BETWEEN Izraz AND R-Izraz
Kolona IS [NOT] NULL
ZnakovniIzraz [NOT] LIKE ZnakovnaMaska
R-Izraz [NOT] IN (Konstanta , …)
R-Izraz { <|>|=|<=|>=|<> } ANY (Konstanta , …)
R-Izraz { <|>|=|<=|>=|<> } ALL (Konstanta , …)

Agregatne funkcije :
SUM (Kolona)
AVG (Kolona)
MIN (Kolona)
MAX (Kolona)
COUNT (*)
COUNT ([ALL | DISTINCT] Kolona , …)

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)



DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

1. Sastaviti SQL upit koji vraća imena svih fudbalera.

```
SELECT Ime
FROM FUDBALER;
```

2. Sastaviti SQL upit koji vraća sva različita imena fudbalera.

```
SELECT DISTICNT Ime
FROM FUDBALER;
```

3. Sastaviti SQL upit koji vraća sve podatke o svim utakmicama.

```
SELECT *
FROM UTAKMICA;
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

4. Sastaviti SQL upit koji vraća sve podatke o utakmicama odigranim u 2019. god.

```
SELECT *
FROM UTAKMICA
WHERE Godina = 2019;
```

5. Sastaviti SQL upit koji vraća imena fudbalera čije ime počinje slovom R.

```
SELECT Ime
FROM FUDBALER
WHERE Ime LIKE 'R%';
```

% - nula, jedan ili više znakova
_ - jedan znak

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

6. Sastaviti SQL upit koji vraća ukupan broj fudbalera.

```
SELECT COUNT (*)
FROM FUDBALER;
```

7. Sastaviti SQL upit koji vraća ukupan broj različitih imena fudbalera.

```
SELECT COUNT (DISTINCT Ime)
FROM FUDBALER;
```

8. Sastaviti SQL upit koji vraća ukupan broj autogolova (na svim utakmicama zajedno).

```
SELECT COUNT(*)
FROM GOL
WHERE Autogol = TRUE;  -- mnogi DBMS nemaju Boolean kao tip podatka (INT ili BIT)
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

9. Sastaviti SQL upit koji vraća prvu i poslednju godinu u kojoj je tim sa šifrom tima 3 odigrao bar jednu utakmicu.

```
SELECT MIN(Godina), MAX(Godina) -- prva je najmanja, a poslednja je najveća u evidenciji
FROM UTAKMICA
WHERE IdTimD=3 OR IdTimG=3;
```

10. Sastaviti SQL upit koji vraća prvu i poslednju godinu u kojoj je tim sa šifrom tima 3 odigrao bar jednu utakmicu sa nerešenim rezultatom.

```
SELECT MIN(Godina), MAX(Godina)
FROM UTAKMICA
WHERE (IdTimD=3 OR IdTimG=3);
    AND Ishod = 'X';
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

11. Sastaviti SQL upit koji za svakog fudbalera koji je igrao na bar jednoj utakmici vraća šifru fudbalera i ukupan broj utakmica koje je taj fudbaler odigrao.

```
SELECT IdFud, COUNT(*)
FROM IGRANJE
GROUP BY IdFud;
```

12. Sastaviti SQL upit koji za svakog fudbalera koji je igrao bar 3 utakmice vraća šifru fudbalera i ukupan broj utakmica koje je taj fudbaler odigrao.

```
SELECT IdFud, COUNT(*)
FROM IGRANJE
GROUP BY IdFud
HAVING COUNT(*) > 2;
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

13. Sastaviti SQL upit koji za svakog fudbalera koji je igrao bar 3 utakmice na poziciji šifre 5 vraća šifru fudbalera i ukupan broj takvih utakmica koje je taj fudbaler odigrao.

```
SELECT IdFud, COUNT(*)
FROM IGRANJE
WHERE IdPoz = 5
GROUP BY IdFud
HAVING COUNT(*) > 2;
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

14. Sastaviti SQL upit koji za svakog fudbalera vraća šifru, ime i naziv tima u kome igra.

```
SELECT F.IdFud, F.Ime, T.Naziv
FROM FUDBALER F, TIM T
WHERE F.IdTim = T.IdTim;
```

15. Sastaviti SQL upit koji za svakog fudbalera koji je igrao bar 3 utakmice vraća ime fudbalera i ukupan broj utakmica koje je taj fudbaler odigrao.

```
SELECT F.Ime, COUNT(*)
FROM FUDBALER F, IGRANJE I
WHERE F.IdFud = I.IdFud
GROUP BY F.IdFud, F.Ime
HAVING COUNT(*) > 2;
```

- a za svakog fudbalera,
bez obzira na broj odigranih utakmica?
(čak i bez ijedne odigrane)

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

16. Sastaviti SQL upit koji za svakog fudbalera vraća ime fudbalera i ukupan broj utakmica koje je taj fudbaler odigrao. (bez obzira na broj odigranih utakmica)

```
SELECT F.Ime, COUNT(*)
FROM FUDBALER F, IGRANJE I
WHERE F.IdFud = I.IdFud
GROUP BY F.IdFud, F.Ime
UNION ALL
SELECT Ime, 0
FROM FUDBALER
WHERE IdFud NOT IN (SELECT IdFud FROM IGRANJE);   -- nekorelisani podupit
```

Kombinovani upiti:
UNION
UNION ALL - ne eliminiše duplikate
INTERSECT
EXCEPT

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

17. Sastaviti SQL upit koji vraća imena fudbalere koji su igrali na svim utakmicama svog tima.

```
SELECT F.Ime
FROM FUDBALER F    -- korelisani podupiti
WHERE (SELECT COUNT (*)  -- broj utakmica tima za koji igra posmatrani fudbaler
        FROM UTAKMICA U
        WHERE F.IdTim = U.IdTimD OR F.IdTim = U.IdTimG
       )
       =
       (SELECT COUNT (*) -- broj utakmica na kojima je igrao posmatrani fudbaler
        FROM IGRANJE I
         WHERE F.IdFud = I.IdFud
        );  -- pod uslovom da igrači ne menjaju timove
```

DML

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

18. Sastaviti SQL upit koji vraća imena fudbalere koji su igrali na svim utakmicama svog tima.
(drugo rešenje za zadatak 17)

```
SELECT F.Ime
FROM FUDBALER F
WHERE NOT EXISTS ((SELECT *
                   FROM UTAKMICA U
                   WHERE F.IdTim = U.IdTimD OR F.IdTim = U.IdTimG
                      AND U.IdUta NOT IN (SELECT I.IdUta
                                          FROM IGRANJE I
                                          WHERE F.IdFud = I.IdFud
                                          )
                   );
```

DML

INSERT INTO Tabela [(Kolona , …)]
{ VALUES (Konstanta , …) } | Upit ;

19. Sastaviti SQL script kojim se u tabelu UTAKMICA dodaje informacija da je su na utakmici sa šifrom 100 u 19. kolu 2022. godine igrali timovi sa šiframa 10 i 15 i da je domaćin pobedio.

```
INSERT INTO UTAKMICA
VALUES (100, 19, 2022, '1', 10, 15);
```

UPDATE Tabela
SET { Kolona = R-Izraz | Upit } , …
[WHERE R-Predikat] ;

20. Sastaviti SQL script kojim se ishod postavlja na nerešeno u svim utakmicama na kojima je tim sa šifrom 10 bio domaćin.

```
UPDATE UTAKMICA
SET Ishod = 'X'
WHERE IdTimD = 10;
```

DELETE FROM Tabela
[WHERE R-Predikat] ;

21. Sastaviti SQL script kojim se brišu sve utakmice na kojima je tim sa šifrom 15 bio domaćin.

```
DELETE FROM UTAKMICA
WHERE IdTimD = 15;
```

DDL

CREATE VIEW Pogled [(Kolona , …)] AS Upit ;
DROP VIEW Pogled ;

- Prednosti upotrebe pogleda:
  - Pojednostavljenje komplikovanih i čestih upita
  - Rešenje problema viška podataka u svodnim upitima
  - Olakšano uspostavljanje kontrole pristupa

- Ažurabilnost pogleda:
  - Upit mora biti nad jednom tabelom
  - Upit sme da sadrži samo imena kolona
  - Upit ne sme da sadrži DISTINCT kaluzulu
  - Upit ne sme biti svodni
  - Ponekad: mora da ima PRIMARY KEY, i/ili da nema podupite

DDL

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

22. Sastaviti SQL skript koji vraća imena fudbalera koji su postigli najveći broj golova.

```
CREATE VIEW FudbalerGolovi (IdFud, BrojGolova)
AS SELECT I.IdFud, COUNT (*)
    FROM IGRANJE I, GOL G
    WHERE I.IdIgr = G.IdIgr
    AND G.Autogol = FALSE
    GROUP BY I.IdFud;

SELECT F.Ime
FROM FUDBALER F
WHERE (SELECT FG.BrojGolova FROM FudbalerGolovi FG WHERE FG.IdFud = F.IdFud)
          = (SELECT MAX(BrojGolova) FROM FudbalerGolovi);
```

DDL

CREATE ASSERTION Pravilo CHECK (Ogranicenje) ;

DROP ASSERTION Pravilo ;

CONSTRAINT Pravilo CHECK (Ogranicenje)
[{ INITIALLY DEFERRED | INITIALLY IMMEDIATE }]
[{ DEFERRABLE | NOT DEFERRABLE }] ;

SET CONSTRAINT { Pravilo, … | ALL }{ DEFERRED | IMMEDIATE }

- Trenutak provere važenja ograničenja zavisi od nivoa na kome je definisano:
  - Kolona (provera zahteva jednu kolonu u jednom redu jedne tabele)
  - Tabela (provera zahteva jedan ili više redova jedne tabele)
  - Baza podataka (provera zahteva više tabela)

DDL

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

23. Sastaviti SQL skript kojim se formira ograničenje da fudbaler ne može da postigne gol nakon crvenog kartona.

```
CREATE ASSERTION NemogucGol
        CHECK ( NOT EXISTS (SELECT *
                            FROM KARTON K, GOL G
                            WHERE K.IdIgr = G.IdIgr
                               AND K.IdTip = (SELECT IdTip
                                              FROM TIP_KARTONA
                                              WHERE Naziv = 'CRVENI')
                               AND K.Minut < G.Minut
                             )
                );
```

DML

NULL

- Neohodno je obratiti pažnju na pojavu NULL prilikom
  1. aritmetičkih operacija (+, -, *, /) – ukoliko je jedan operand NULL i rezultat je NULL
  2. operacija poređenja (<, >, =, <>) – poređenje sa NULL vraća NULL
     AND – TRUE AND NULL je NULL, FALSE AND NULL je FALSE, NULL AND NULL je NULL
     OR – TRUE OR NULL je TRUE, FALSE OR NULL je NULL, NULL OR NULL je NULL
     NOT – NOT NULL je NULL
  3. skupovnih operacija (UNION, INTERSECTION, EXCEPT) – sve NULL tumači kao da su jednaki

- WHERE klauzula posmatra NULL istovetno kao da je FALSE
- Predikat IS NULL, odnosno IS NOT NULL
- Testiranje da li je rezultat poređenja nepoznat koristeći IS UNKNOWN, IS NOT UNKNOWN
- DISTICNT klauzula sve NULL tumači kao da su jednaki (što je suprotno od posmatranja =)
- Agregatne funkcije – sve funkcije ignorišu NULL, izuzev COUNT(*)
  – za praznu kolekciju vraćaju NULL, izuzev COUNT koji vraća 0
- Predikati ALL, ANY, SOME, EVERY, IN, BETWEEN se mogu prikazati koristeći AND, OR, NOT
- CHECK klauzula posmtra NULL istovetno kao da je TRUE

DML – SQL92, SQL99

CASE Verzija 1
SelectorCASE ::= CASE Slucaj Slucaj . . . [ELSE S-Izraz] END
Slucaj ::= WHEN R-Predikat THEN S-Izraz

CASE Verzija 2
SelectorCASE ::= CASE Operand Provera Provera . . . [ELSE S-Izraz] END
Provera ::= WHEN Vrednost THEN S-Izraz

COALESCE()
COALESCE (< izraz1 >, < izraz2 >, ..., n)
⇔
CASE WHEN < izraz1 > IS NOT NULL THEN < izraz1 >
ELSE COALESCE (< izraz2 >, ..., n)
END

NULLIF()
NULLIF (< izraz1 >, < izraz2 >)
⇔
CASE WHEN < izraz1 > = < izraz2 > THEN NULL
ELSE <izraz1> END

DML – SQL92, SQL99

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

24. Sastaviti SQL skript kojim se za svaku utakmicu vraća šifru utakmice, broj žutih kartona, broj crvenih kartona datih na toj utakmici, ali samo za utakmice na kojima je dat bar jedan karton.

```
SELECT I.IdUta,
        SUM(CASE WHEN T.Naziv = 'ZUTI' THEN 1 ELSE 0 END) AS BrZutih,
        SUM(CASE WHEN T.Naziv = 'CRVENI' THEN 1 ELSE 0 END) AS BrCrvenih,
FROM IGRANJE I, KARTON K, TIP_KARTONA T
WHERE I.IdIgr = K.IdIgr
   AND K.IdTip = T.IdTip
GROUP BY I.IdUta;
```

DML – SQL92, SQL99

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

25. Sastaviti SQL skript kojim se za svaku utakmicu na kojoj je postignuto tačno 3 gola vraća minut u kome je postignut najraniji gol, a u suprotnom vraća ukupan broj autogolova, ali samo za utakmice na kojima je postignut bar jedan gol.

```
SELECT I.IdUta,
        CASE WHEN COUNT(*) = 3 THEN MIN (G.Minut)
                ELSE SUM(CASE WHEN G.Autogol = TRUE THEN 1 ELSE 0 END) END
FROM IGRANJE I, GOL G
WHERE I.IdIgr = G.IdIgr
GROUP BY I.IdUta;
```

DML – SQL92, SQL99

JOIN

Spajanjetabela ::= TabelarniIzraz IzrazJOIN TabelarniIzraz
TabelarniIzraz ::= Tabela | (R-Upit) | SpajanjeTabela [AS Nadimak]
IzrazJOIN ::= DekartovProizvod | PrirodnoSpajanje | OstaloSpajanje
DekartovProizvod ::= Tabela CROSS JOIN Tabela
PrirodnoSpajanje ::= Tabela NATURAL JOIN Tabela
OstaloSpajanje ::= Tabela VrstaSpajanja JOIN Tabela OsnovSpajanje
VrstaSpajanje ::= VrstaUnutrasnje | VrstaSpoljno
VrstaUnutrasnje ::= _ | INNER
VrstaSpoljno ::= LEFT | RIGHT | FULL [OUTER]
OsnovSpajanja ::= OsnovUslova | OsnovJednakostiKolona
OsnovUslova ::= ON R-Predikat
OsnovJednakostKolona ::= USING ({ Kolona, Kolona }, …)

DML – SQL92, SQL99

JOIN

T1	a	x
	1	R
	2	V
	3	NULL

T2	b	x
	7	R
	8	S
	9	NULL

SELECT * FROM T1 INNER JOIN T2 ON (T1.x =T2.x)

a	T1.x	B	T2.x
1	R	7	R

SELECT * FROM T1 LEFT OUTER JOIN T2 ON (T1.x =T2.x)

a	T1.x	B	T2.x
1	R	7	R
2	V	NULL	NULL
3	NULL	NULL	NULL

SELECT * FROM T1 FULL OUTER JOIN T2 ON (T1.x =T2.x)

a	T1.x	B	T2.x
1	R	7	R
2	V	NULL	NULL
3	NULL	NULL	NULL
NULL	NULL	8	S
NULL	NULL	9	NULL

DML – SQL92, SQL99

MESTO (IdMes, Naziv)
POZICIJA (IdPoz, Naziv)
TIM (IdTim, Naziv, IdMes)
FUDBALER (IdFud, Ime, IdTim)
UTAKMICA (IdUta, Kolo, Godina, Ishod, IdTimD, IdTimG) KK: {IdTimD, IdTimG}
IGRANJE (IdIgr, IdFud, IdUta, IdPoz) KK: {IdFud, IdUta}
TIP_KARTONA (idTip, Naziv)
KARTON (IdKar, Minut, IdTip, IdIgr) KK: {IdIgr, IdTip}
GOL (IdGol, RedniBr, Minut, Autogol, IdIgr)

26. Sastaviti SQL skript koji za svaki tim vraća naziv tima i imena svih fudbalera u tom timu čije ime ime počinje slovom T.

```
SELECT TIM.Naziv, FUDBALER.Ime
FROM TIM LEFT OUTER JOIN FUDBALER ON TIM.IdTim = FUDBALER.IdTim
WHERE FUDBALER.Ime LIKE 'T%';
```

```
SELECT TIM.Naziv, FUDBALER.Ime
FROM TIM LEFT OUTER JOIN FUDBALER ON TIM.IdTim = FUDBALER.IdTim
AND FUDBALER.Ime LIKE 'T%';
```

```
-- prvi upit neće vratiti timove koji nemaju nijednog fudbalera čije ime počinje sa T
-- drugi upit će vratiti čak i timove koji nemaju nijednog fudbalera čije ime počinje sa T
```

DML – SQL92, SQL99

WITH [RECURSIVE] Pogled [(Kolona, ..)] AS (Upit) OsnovniUpit

Ako bi se za svakog fudbalera evidentirao i fudbaler koji mu je bio mentor:

…

TIM (IdTim, Naziv, IdMes)

FUDBALER (IdFud, Ime, IdTim, IdFudMentor)

27. Sastaviti SQL skript koji vraća šifre i imena svih fudbalera kojima je fudbaler sa šifrom 12 bio direktni ili indirektni mentor. (gde indirektni mentor jeste mentor mentoru)

```
WITH RECURSIVE Mentori (IdFud, Ime, IdFudMentor)
AS (SELECT F.IdFud, F.Ime, F.IdFudMentor
      FROM FUDBALER F
      WHERE F.IdFud = 12  -- pocetak pretrage
      UNION
      SELECT F.IdFud, F.Ime, F.IdFudMentor
      FROM Mentori M, FUDBALER F
      WHERE M.IdFud = F.IdFudMentor
     )
-- Glavni upit
SELECT IdFud, Ime, IdFudMentor
FROM Mentori
WHERE IdFud <> 12;
```