

NULL VRIJEDNOSTI I INTEGRITET PODATAKA

VI predavanje

Dr.sc. Emir Mešković

NULL vrijednosti

- ▶ Ponekad se dešava da informacije koje treba unijeti u bazu podataka **nisu potpune**
 - ▶ neke informacije trenutno nisu poznate
 - ▶ neke informacije uopće ne postoje
 - ▶ neke informacije postoje, ali do njih nije moguće doći
- ▶ Informacije koje nedostaju prikazuju se kao **NULL-vrijednosti**
- ▶ Za korisnika/programera NULL-vrijednost je neovisna o tipu podatka kojeg predstavlja

Primjer

► Evidencija članova biblioteke

<u>CLAN</u> (<u>clanBr</u> , <u>prezime</u> , <u>ime</u> , <u>postBr</u> , <u>adresa</u> , <u>vr</u> , <u>matBrSt</u> , <u>nastZv</u>)								
12345	Pirić	Damir	75000	I. Mujezinovića 25	S	II-37/10	NULL	ne postoji ↓
34567	Pejić	Dino	75000	NULL	N	NULL	Asist.	
				↑ nepoznato		↑ ne postoji		

► Horizontalna normalizacija

- Relacija *clan* razdijeli se na fragmente prema vrijednosti atributa *Vr*, te se dobiveni fragmenti projiciraju na attribute koje određene vrste članova posjeduju.

<u>STUDENT</u> (<u>clanBr</u> , <u>prezime</u> , <u>ime</u> , <u>postBr</u> , <u>adresa</u> , <u>vr</u> , <u>matBrSt</u>)							
12345	Pirić	Damir	75000	I. Mujezinovića 25	S	II-37/10	
<u>NASTAVNIK</u> (<u>clanBr</u> , <u>prezime</u> , <u>ime</u> , <u>postBr</u> , <u>adresa</u> , <u>vr</u> , <u>nastZv</u>)							
34567	Pejić	Dino	75000	NULL	N	Asist.	

Interna pohrana i prikaz NULL vrijednosti

- ▶ NULL vrijednost **se razlikuje** od
 - ▶ vrijednosti 0 za numerički podatak
 - ▶ praznog niza za podatke tipa niza znakova
 - ▶
- ▶ Način interne pohrane NULL vrijednosti je nebitan – NULL vrijednost je neovisna o tipu podatka koji predstavlja
 - ▶ U SQL naredbama bez obzira na tip podatka koristi se “konstanta” NULL
- ▶ NULL vrijednost se interno prikazuje drugačije od bilo koje druge dozvoljene vrijednosti
 - ▶ Npr. mali cijeli brojevi imaju raspon [-32768, 32767]
 - ▶ NULL → -32768 , raspon je [-32767, 32767]
- ▶ Način na koji se NULL vrijednost prikazuje korisniku ovisi o programskom alatu koji se koristi (NULL, <null>, prazna ćelija, ...)

Pravila za rukovanje NULL vrijednostima

- ▶ Aritmetički izrazi
 - ▶ Neka je aritmetički operator $\alpha \in \{+, -, *, /\}$,
 - ▶ neka su X i Y numerički izrazi.
- ▶ Rezultat aritmetičkog izraza: $X \alpha Y$ je **NULL** ukoliko jedan ili oba operanda (X, Y) imaju vrijednost NULL
 - ▶ $10 + \text{NULL} \rightarrow \text{NULL}$
 - ▶ $\text{NULL} * 1 \rightarrow \text{NULL}$
 - ▶ $\text{NULL} - \text{NULL} \rightarrow \text{NULL}$
- ▶ Unarne operacije $+$ i $-$
 - ▶ Ako X poprimi NULL vrijednost, tada su rezultati operacija $+X$ i $-X$ također NULL
 - ▶ $-\text{NULL} \rightarrow \text{NULL}$

Pravila za rukovanje NULL vrijednostima

nastavnik

sifNast	prezNast	imeNast	koef	proc
1001	Pirić	Damir	5.67	5
1002	Đurić	Maja	6.02	NULL
1003	Žunić	Senad	4.78	10
1004	Pejić	Ema	NULL	NULL

```
SELECT sifNast, prezNast,  
       koef * (1 + proc * 0.01) AS ukupKoef,  
       -proc AS negProc  
FROM nastavnik
```

sifNast	prezNast	ukupKoef	negProc
1001	Pirić	5.95	-5
1002	Đurić	NULL	NULL
1003	Žunić	5.26	-10
1004	Pejić	NULL	NULL

Pravila za rukovanje s NULL vrijednostima

- ▶ Relacijski izrazi - **aritmetički operatori poređenja**
 - ▶ Neka je operator $Q \in \{ >, <, >=, <=, <>, = \}$
 - ▶ Neka je relacijski izraz: $X Q Y$
- ▶ U dvovalentnoj logici rezultat poređenja je uvijek ili *true* ili *false*
- ▶ U trovalentnoj logici - rezultat poređenja može biti *true*, *false* ili *unknown*:
 - ▶ ukoliko jedan ili oba operanda (X, Y) imaju vrijednost NULL - **rezultat poređenja je logička vrijednost nepoznato (*unknown*)**
 - ▶ $10 < 5 \rightarrow false$
 - ▶ $NULL \geq 15.5 \rightarrow unknown$
 - ▶ $NULL = NULL \rightarrow unknown$
 - ▶ $NULL \neq NULL \rightarrow unknown$

Pravila za rukovanje s NULL vrijednostima

- Osnovne logičke operacije - Tablice istinitosti u **trovalentnoj logici** (u prisustvu logičke vrijednosti **unknown**):

OR	<i>true</i>	<i>unknown</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>
<i>unknown</i>	<i>true</i>	<i>unknown</i>	<i>unknown</i>
<i>false</i>	<i>true</i>	<i>unknown</i>	<i>false</i>

AND	<i>true</i>	<i>unknown</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>unknown</i>	<i>false</i>
<i>unknown</i>	<i>unknown</i>	<i>unknown</i>	<i>false</i>
<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>

NOT	
<i>true</i>	<i>false</i>
<i>unknown</i>	<i>unknown</i>
<i>false</i>	<i>true</i>

NULL vrijednost i skupovi

- ▶ Neka skup S sadrži vrijednosti:
 - ▶ $S = \{1, 2, 3, \text{NULL}\}$
- ▶ NULL vrijednost u skupu S mogla bi se posmatrati na dva načina:
 - ▶ NULL vrijednost je različita od 1, 2 i 3
 - ▶ NULL vrijednost je nepoznata – može poprimiti i neku od vrijednosti 1, 2 ili 3.
- ▶ Kardinalnost skupa S je u ovom slučaju neodređena (može biti 3 ili 4)
- ▶ Narušava se i definicija skupa (nije dozvoljena pojava dvije ili više jednakih vrijednosti)

Definicija kopije

- ▶ Sistemi za upravljanje bazama podataka nisu u stanju razlikovati NULL vrijednosti
- ▶ Model rukovanja s NULL vrijednostima u skupovima:
 - ▶ dozvoljena je pojava **jedne i samo jedne** NULL vrijednosti u skupu
 - ▶ Element **e** je **kopija** jednog od elemenata u skupu:
 - ▶ ako u skupu postoji element s jednakom vrijednošću
 - ▶ ako je vrijednost elementa **e** NULL, a u skupu već postoji element s NULL vrijednošću

Definicija kopije u relaciji

- ▶ n-torka (d_1, d_2, \dots, d_n) je **kopija** n-torke (e_1, e_2, \dots, e_n) ako i samo ako:
 - ▶ $[(d_i = e_i) \vee (d_i = \text{NULL} \wedge e_i = \text{NULL})], \forall i \in \{1, \dots, n\}$

- ▶ Primjer:

ZAPOSLENIK(<u>prezime</u> , <u>ime</u> , <u>JMBG</u>)				
t_1	Pirić	Damir	NULL	nisu kopije
t_2	Đurić	Maja	2812964185017	
t_3	Pejić	Dino	NULL	kopije
t_4	Pirić	Damir	0301958180001	
t_5	Pejić	Dino	NULL	

- ▶ t_1 i t_4 nisu kopije, iako će možda NULL vrijednost u t_1 biti zamijenjena s vrijednošću 0301958180001
- ▶ t_3 i t_5 jesu kopije, iako će možda NULL vrijednosti biti zamijenjene različitim vrijednostima

Osnovne operacije s relacijama

$r(\underline{X} \quad Y)$

a	b
a	NULL
NULL	b
NULL	NULL

$s(\underline{X} \quad Y)$

c	d
NULL	b
NULL	NULL

$r \cup s = p(\underline{X} \quad Y)$

a	b
a	NULL
NULL	b
NULL	NULL
c	d

$r \cap s = pp(\underline{X} \quad Y)$

NULL	b
NULL	NULL

$s \setminus p = pppp(\underline{X} \quad Y)$

c	d
---	---

$r \setminus s = ppp(\underline{X} \quad Y)$

a	b
a	NULL



Operacija projekcije

- ▶ Prilikom obavljanja operacije projekcije potrebno je u fazi eliminacije duplikata voditi računa o definiciji kopije n-torke

- ▶ $s = \pi_{B,C}(r)$

r		
A	B	C
1	a	α
2	b	NULL
3	NULL	β
4	a	α
5	NULL	NULL
6	NULL	β
7	NULL	NULL

“međurezultat”	
B	C
a	α
b	NULL
NULL	β
a	α
NULL	NULL
NULL	β
NULL	NULL

s	
B	C
a	α
b	NULL
NULL	β
NULL	NULL

Operacija selekcije

- ▶ Prilikom obavljanja operacije selekcije $\sigma_F(r)$ dobija se relacija koja sadrži samo one n-torke relacije r za koje je vrijednost predikata F istina (*true*)

Primjer: $r(\text{prezime}, \text{postBr})$

Pirić	75000
Pejić	71000
Đurić	NULL
Žunić	72000

Selekcija: $\sigma_{\text{postBr} \neq 75000}(r)$

```
SELECT * FROM R WHERE postBr <> 75000
```

Rezultat:

<Pejić, 71000>, <Žunić, 72000>

jer se za n-torku <Pirić, 75000> uslov selekcije evaluira kao *false*,
a za n-torku <Đurić, NULL> evaluira se kao *unknown*

Operacija spajanja

- ▶ Prilikom obavljanja bilo kojeg oblika operacije spajanja (spajanje uz uslov, spajanje sa izjednačavanjem, prirodno spajanje) potrebno je voditi računa o tome da se spajaju samo one n-torke za koje uslov spajanja ima logičku vrijednost istina (*true*)
 - ▶ Prilikom obavljanja operacije Dekartovog proizvoda NULL vrijednosti nemaju uticaja

Operacija spajanja

Primjer : r (<u>prezime</u> , <u>postBr</u>)		s (<u>postBr</u> , <u>grad</u>)	
Pirić	75000	75000	Tuzla
Pejić	71000	71000	Sarajevo
Đurić	NULL	88000	Mostar
Žunić	72000		

Prirodnim spajanjem relacija dobije se:

r \bowtie s = t (<u>prezime</u> , <u>postBr</u> , <u>grad</u>)		
Pirić	75000	Tuzla
Pejić	71000	Sarajevo

Nepojavljivanje osobe s prezimenom **Žunić** rezultat je evaluiranja uslova $r.postBr = s.postBr$ s rezultatom *false*.

Nepojavljivanje osobe s prezimenom **Đurić** rezultat je evaluiranja uslova $r.postBr = s.postBr$ s rezultatom *unknown*.

Operacija spajanja

Primjer : $r (\underline{\text{prezime}}, \underline{\text{postBr}})$

Pirić	75000
Pejić	71000
Đurić	NULL
Žunić	72000

$s (\underline{\text{postBr}}, \underline{\text{grad}})$

75000	Tuzla
71000	Sarajevo
88000	Mostar
NULL	Nepoznato

Prirodnim spajanjem relacija dobije se:

$r \triangleright \triangleleft s = t (\underline{\text{prezime}}, \underline{\text{postBr}}, \underline{\text{grad}})$

Pirić	75000	Tuzla
Pejić	71000	Sarajevo

Nepojavljivanje osobe s prezimenom **Žunić** rezultat je evaluiranja uslova $r.\text{postBr} = s.\text{postBr}$ s rezultatom *false*.

Nepojavljivanje osobe s prezimenom **Đurić** rezultat je evaluiranja uslova $r.\text{postBr} = s.\text{postBr}$ s rezultatom *unknown*.

Vanjsko spajanje - Outer join

- sve n-torke iz **r** pojavit će se ukoliko koristimo operaciju lijevog vanjskog spajanja **r * ▷◁ s**
 - n-torkama “lijeve” relacije za koje ne postoje odgovarajuće n-torke u “desnoj” relaciji se kao vrijednosti atributa iz “desne” relacije postavljaju NULL vrijednosti

```
SELECT * FROM R LEFT OUTER JOIN S
      ON R.PostBroj = S.PostBroj
```

t (<u>prezime</u>	<u>postBr</u>	<u>postBr</u>	<u>grad</u>)
Pirić	75000	75000	Tuzla
Pejić	71000	71000	Sarajevo
Đurić	NULL	NULL	NULL
Žunić	72000	NULL	NULL

Vanjsko spajanje - Outer join

- sve n-torke iz **s** pojavit će se ukoliko koristimo operaciju desnog vanjskog spajanja **$r \triangleright \triangleleft * s$**
 - n-torkama “lijeve” relacije za koje ne postoje odgovarajuće n-torke u “desnoj” relaciji se kao vrijednosti atributa iz “desne” relacije postavljaju NULL vrijednosti

```
SELECT * FROM R RIGHT OUTER JOIN S  
ON R.PostBroj = S.PostBroj
```

t (<u>prezime</u>	<u>postBr</u>	<u>postBr</u>	<u>grad</u>)
Pirić	75000	75000	Tuzla
Pejić	71000	71000	Sarajevo
NULL	NULL	88000	Mostar

Vanjsko spajanje - Outer join

→ sve n-torke iz **obje relacije** pojavit će se ukoliko koristimo operaciju punog vanjskog spajanja $r * \bowtie * s$

```
SELECT * FROM R FULL OUTER JOIN S
      ON R.PostBroj = S.PostBroj
```

t (prezime	postBr	postBr	grad)
Pirić	75000	75000	Tuzla
Pejić	71000	71000	Sarajevo
Đurić	NULL	NULL	NULL
Žunić	72000	NULL	NULL
NULL	NULL	88000	Mostar

Integritet baze podataka

- ▶ Pojam integriteta baze podataka odnosi se na **ispravnost (konzistentnost)** i **istinitost** podataka sadržanih u bazi.
- ▶ Neispravni ili netačni podaci mogu biti posljedica:
 - ▶ slučajne pogreške kod unosa ili ažuriranja
 - ▶ pogreške aplikacijskog programa ili sistema
- ▶ Integritet baze podataka može biti narušen i zbog posljedica diverzije ili sabotaze, međutim o tome brine poseban dio SUBP koji je zadužen za sigurnost baze podataka.
- ▶ **Integritetska ograničenja** osiguravaju da izmjene podataka koje obavljaju korisnici ne rezultiraju narušavanjem konzistentnosti podataka

Integritetska ograničenja

- ▶ Definicije integritetskih ograničenja su sastavni dio sheme baze podataka
- ▶ Definicije integritetskih ograničenja se pohranjuju u riječnik podataka baze podataka
 - ▶ Na taj način pravila definirana integritetskim ograničenjima postaju nezaobilazna za svakog korisnika sistema
 - ▶ SUBP provjerava integritetska ograničenja pri obavljanju svake operacije koja mijenja sadržaj baze podataka
 - ▶ U trenutku završetka operacije nad podacima baza podataka mora biti u stanju u kojem su zadovoljena sva integritetska ograničenja
 - ▶ SUBP odbija obaviti operacije koje nemaju to svojstvo ili obavlja kompenzacijske akcije koje osiguravaju da su na kraju sva integritetska ograničenja zadovoljena

Integritetska ograničenja

- ▶ **Osnovna integritetska ograničenja**
 - ▶ Entitetski integritet (Entity integrity)
 - ▶ Integritet ključa (Key integrity)
 - ▶ Referencijski integritet (Referential integrity)
- ▶ **Korisnička integritetska ograničenja**
 - ▶ Domenski integritet (Domain integrity)
 - ▶ Ograničenja NULL vrijednosti (Constraints on NULL)
 - ▶ Opća integritetska ograničenja (General integrity constraints)

Entitetski integritet

- ▶ (Codd, 1970) - Vrijednost primarnog ključa kao cjeline, ne smije biti jednaka NULL vrijednosti.
- ▶ Ako je primarni ključ relacije složen, niti jedna njegova komponenta ne smije poprimiti NULL

Primjer:

$NASTAVNIK = \{ sifNas, prezNast \}$

$K_{NASTAVNIK} = \{ sifNas \} \rightarrow sifNas$ ne smije biti NULL

$ISPIT = \{ matbr, sifPred, datlsp, ocj, sifNas \}$

$K_{ISPIT} = \{ matbr, sifPred, datlsp \}$

$\rightarrow matBr, sifPred, datlsp$ ne smiju biti NULL

Integritet ključa

- ▶ U relaciji ne smiju postojati dvije n-torke s jednakim vrijednostima ključa
 - ▶ Ovo ograničenje vrijedi za sve moguće ključeve

Primjer:

$NASTAVNIK = \{ sifNas, jmbgNast, prezNast \}$

$PK_{NASTAVNIK} = \{ sifNas \}$ $K2_{NASTAVNIK} = \{ jmbgNas \}$

- u relaciji nastavnik (NASTAVNIK) ne smiju postojati dvije n-torke koje imaju jednake vrijednosti atributa sifNas
- u relaciji nastavnik (NASTAVNIK) ne smiju postojati dvije n-torke koje imaju jednake vrijednosti atributa jmbgNas

Referencijski integritet

- ▶ Referencijski integritet se odnosi na konzistentnost među n-torkama dvije relacije (ili n-torkama iste relacije).
- ▶ Neformalno: n-torka iz jedne relacije koja se poziva (referencira) na drugu relaciju se može pozvati (referencirati) samo na postojeće n-torke (primarne ključeve) u toj relaciji

Referencijski integritet

- ▶ Ako u relacijskoj shemi R postoji strani ključ FK_R koji odgovara primarnom ključu relacijske sheme S PK_S , tada svaka vrijednost stranog ključa u relaciji $r(R)$ $t_i(FK_R)$ mora biti:
 - ▶ ili jednaka vrijednosti primarnog ključa neke n -torke iz relacije $s(S)$ $t_j(PK_S)$, tj. $t_i(FK_R) = t_j(PK_S)$
 - ▶ ili jednaka NULL vrijednosti, tj. $t_i(FK_R) = \text{NULL}$
- ▶ Relacija $r(R)$ se naziva pozivajuća, a relacija $s(S)$ se naziva pozivana relacija
 - ▶ Relacije $r(R)$ i $s(S)$ ne moraju nužno biti različite relacije
- ▶ Referencijski integritet se odnosi na ograničenje koje proizilazi iz definicije stranog ključa

Referencijski integritet

Primjer

<i>osoba</i> (<i>sifra</i>	<i>prezime</i>	<i>postBr</i>)	<i>grad</i> (<i>postBr</i>	<i>nazivGrada</i>)
100	Pirić	75000	75000	Tuzla
107	Đurić	NULL	71000	Sarajevo
109	Pejić	72000	88000	Mostar

- ▶ Relacije *osoba* i *grad* **ne zadovoljavaju pravilo referencijskog integriteta** jer u relaciji *osoba* postoji vrijednost stranog ključa (72000) za koju ne postoji odgovarajuća n-torka u relaciji *grad*.
- ▶ n-torka <107, Đurić, NULL> ne narušava referencijski integritet

Referencijski integritet

- ▶ Postoje slučajevi kad strani ključ iz $r(R)$ **ne smije biti jednak NULL vrijednosti**. To vrijedi za slučaj kad se pravila referencijskog integriteta sukobe s nekim drugim pravilima integriteta (npr. pravilom entitetskog integriteta ili ograničenjem NULL vrijednosti)
- ▶ strani ključ relacije $r(R)$ koji je ujedno dio primarnog ključa relacije $r(R)$ **ne smije poprimiti NULL vrijednost!**

STUDENT = { *matbr*, *prezime*, *ime* }

$K_{\text{STUDENT}} = \{ \textit{matbr} \}$

ISPIT = { *matbr*, *sifPred*, *datlsp*, *ocj*, *sifNas* }

$K_{\text{ISPIT}} = \{ \textit{matbr}, \textit{sifPred}, \textit{datlsp} \}$

→ *matBr* ne smije poprimiti NULL vrijednost!

Domenski integritet

- ▶ Definiše domenu atributa - specificira skup vrijednosti koje atribut smije poprimiti
- ▶ Primjer: $MJESTO = \{ pbr, nazMjesto \}$
- ▶ Domena atributa pbr je skup cijelih brojeva iz intervala [10000, 99999]
 - ▶ Vrijednost atributa pbr u svakoj n-torki relacije mjesto(MJESTO) mora biti cijeli broj iz intervala [10000, 99999]

Ograničenja NULL vrijednosti

- ▶ Za određene attribute se može definirati ograničenje prema kojem vrijednost atributa ne smije poprimiti NULL vrijednost
- ▶ Primjer: $RADNIK = \{ \text{sifRad}, \text{imeRad}, \text{prezRad}, \text{email} \}$
 - ▶ Vrijednost atributa imeRad ne smije poprimiti NULL vrijednost niti u jednoj n-torci relacije radnik(RADNIK)
 - ▶ Vrijednost atributa prezRad ne smije poprimiti NULL vrijednost niti u jednoj n-torci relacije radnik(RADNIK)

Opća integritetska ograničenja

- ▶ Opća integritetska ograničenja su ograničenja općeg (generalnog) oblika
 - ▶ Npr. određuju se dozvoljeni odnosi među pojedinim atributima
- ▶ Primjer: **ZAPOSLENIK** = {sifra, prezime, starost, staz }
 - ▶ Npr. može se definisati da **Starost \geq Staz + 16** i **Starost \leq Staz + 65**
 - ▶ Ovo integritetsko ograničenje proizilazi iz (za ovaj primjer izmišljenog) zakonskog ograničenja da se osobe mlađe od 16 godina ili starije od 65 godina ne smiju zapošljavati

Implementacija integritetskih ograničenja

- ▶ Potrebno je definisati:
 - ▶ pod kojim uslovima se definitivno odbija obavljanje operacije koja bi narušila integritetska ograničenja
 - ▶ pod kojim uslovima se obavlja operacija uz obavljanje nekih kompenzacijskih operacija
- ▶ Ograničenja entitetskog integriteta i integriteta ključa **nužno moraju zadovoljiti**
 - ▶ ne smije biti nikakvog odstupanja
 - ▶ ne postoje kompenzacijske operacije koje se mogu izvesti

Implementacija integritetskih ograničenja

- ▶ Referencijski integritet za kritične operacije, npr. operaciju brisanja dozvoljava sljedeće strategije
 - ▶ ciljna n-torka **se ne može obrisati** ako u bazi postoje odgovarajuće pozivajuće n-torke
 - ▶ uz brisanje ciljne n-torke treba izvesti **brisanje svih pozivajućih n-torki** kojima je vrijednost stranog ključa jednaka vrijednosti primarnog ključa ciljne n-torke
 - ▶ kao dio operacije brisanja ciljne n-torke, **vrijednosti stranih ključeva u pozivajućim n-torkama postavljaju se na NULL**

Implementacija entitetskog integriteta u SQL-u

- ▶ osigurava se definisanjem primarnog ključa s pomoću naznake **PRIMARY KEY**
- ▶ time SUBP osigurava:
 - ▶ ključni atributi relacije ne smiju imati vrijednost **NULL**
 - ▶ jedinstvenost ključa

```
CREATE TABLE ispit (  
  matbr      INTEGER  
,sifPred    INTEGER  
,datIsp     DATE  
,ocj        SMALLINT  
,sifNas      INTEGER  
,PRIMARY KEY (matbr, sifPred, datIsp)
```

```
CREATE TABLE student (  
  matbr      INTEGER PRIMARY KEY  
,prezime    CHAR(20)  
,ime        CHAR(20)  
) ;
```

SUBP osigurava:
entitetski integritet i
integritet ključa

Implementacija integriteta ključa u SQL-u

- ▶ osigurava se s pomoću naznake **UNIQUE**
- ▶ time SUBP osigurava:
 - ▶ jedinstvenost ključa

```
CREATE TABLE mjesto(  
  pbr          INTEGER  
, nazMjesto   CHAR(25)  
, sifKanton   INTEGER  
, UNIQUE(nazMjesto, sifKanton);
```

```
CREATE TABLE student (  
  matbr      INTEGER PRIMARY KEY  
, prezime   CHAR(20)  
, ime       CHAR(20)  
, jmbgStud  CHAR(13) UNIQUE  
);
```

SUBP osigurava:
integritet ključa

Implementacija domenskog integriteta u SQL-u

- ▶ djelimično je definisan samom definicijom tipa podatka
 - ▶ npr. definisanjem podatka tipa SMALLINT određena je njegova domena kao skup cijelih brojeva u intervalu -32767 do 32767
- ▶ tačnije određenje domene atributa može se postići definisanjem prilikom kreiranja tablice s pomoću naznake **CHECK**:

```
CREATE TABLE ispit (  
  matbr      INTEGER  
, sifPred   INTEGER  
, datIsp    DATE  
, ocj       SMALLINT  
, sifNas    INTEGER  
, PRIMARY KEY (matbr
```

```
CHECK (ocj BETWEEN 1 AND 5)
```

SUBP osigurava:
domenski integritet

```
CREATE TABLE ispit (  
  ...  
, ocj       SMALLINT  
, PRIMARY KEY (matbr, sifPred, datIsp)  
, CHECK (ocj BETWEEN 1 AND 5);
```


Implementacija općih integritetskih ograničenja u SQL-u

- ▶ **CHECK** naznaka se također može koristiti za definiranje ograničenja odnosa među vrijednostima atributa u istoj n-torci
- ▶ Ograničenje koje se tiče odnosa među vrijednostima atributa se ne može napisati neposredno uz definiciju atributa

SUBP osigurava:

ograničenje odnosa između
vrijednosti atributa u istoj n-torci

```
CREATE TABLE radnik(  
    matbr      INTEGER PRIMARY KEY  
    , prezime  CHAR(20)  
    , ime      CHAR(20)  
    , datRod   DATE  
    , datZap   DATE  
    , CHECK (ADDDATE(datRod, INTERVAL 16 YEAR) >= datZap  
            AND ADDDATE(datRod, INTERVAL 65 YEAR) >= datZap));
```



Implementacija referencijskog integriteta u SQL-u

- ▶ Osigurava se s pomoću naznake **FOREIGN KEY**:
 - ▶ Definira skup atributa koji je strani ključ u relaciji
 - ▶ Definira relaciju i skup atributa na koji referencira strani ključ

```
CREATE TABLE ispit (  
    matbr      INTEGER  
,sifPred    INTEGER  
,datIsp     DATE  
,ocj        SMALLINT  
,sifNas      INTEGER REFERENCES nastavnik(sifNas)  
,PRIMARY KEY (matbr, sifPred, datIsp)  
,FOREIGN KEY (matbr) REFERENCES student(matbr)  
,FOREIGN KEY (sifPred) REFERENCES predmet(sifPred) );
```

SUBP osigurava referencijalni integritet:

Strani ključ u relaciji ispit (atribut matbr) poziva se na primarni ključ u relaciji student (matbr)

Podrazumijeva se da su u relacijama *student*, *predmet* i *nastavnik* pomoću PRIMARY KEY definirana ograničenja entitetskog integriteta

Implementacija referencijskog integriteta u SQL-u

- ▶ Pri definiciji ograničenja referencijskog integriteta moguće je specificirati da li će i koje kompenzirajuće akcije SUBP izvesti prilikom pokušaja narušavanja ograničenja izvođenjem operacije brisanja ili ažuriranja ciljne n-torke
 - ▶ SUBP odbija operacije brisanja ili ažuriranja ciljne n-torke
 - ▶ **ON DELETE NO ACTION, ON UPDATE NO ACTION**
 - ▶ SUBP obavlja operaciju brisanja ili ažuriranja ciljne n-torke i pri tome vrijednosti stranog ključa u n-torkama koje se pozivaju na obrisanu ili ažuriranu n-torku postavlja na NULL ili *default* vrijednost
 - ▶ **ON DELETE SET NULL, ON UPDATE SET NULL**
 - ▶ **ON DELETE SET DEFAULT, ON UPDATE SET DEFAULT**
 - ▶ SUBP obavlja operaciju brisanja ili ažuriranja ciljne n-torke i pri tome kaskadno briše pozivajuće n-torke, odnosno vrijednosti stranog ključa u n-torkama koje se pozivaju na ažuriranu n-torku postavlja na novu vrijednost primarnog ključa ciljne n-torke
 - ▶ **ON DELETE CASCADE, ON UPDATE CASCADE**

Implementacija referencijskog integriteta u SQL-u

► Referencijski integritet definisan uz obavljanje kompenzacijskih akcija

```
CREATE TABLE ispit (  
    matbr      INTEGER  
,sifPred     INTEGER  
,datIsp      DATE  
,ocj         SMALLINT  
,sifNas      INTEGER  
,PRIMARY KEY (matbr, sifPred, datIsp)  
,FOREIGN KEY (matbr) REFERENCES student(matbr)  
    ON DELETE CASCADE  
,FOREIGN KEY (sifPred) REFERENCES predmet(sifPred)  
,FOREIGN KEY (sifNas) REFERENCES nastavnik(sifNas)  
    ON DELETE SET NULL  
);
```

Obrisat će se n-torke iz relacije *student* i sve n-torke iz relacije *ispit* koje se pozivaju na obrisane n-torke relacije *student*

Operacija brisanja n-torki iz relacije *predmet* će biti odbijena – korisnik ili aplikacija će dobiti poruku o pogrešci


Obrisat će se n-torke iz relacije *nastavnik*, a vrijednosti stranog ključa (*sifNas*) u relaciji *ispit* koje se pozivaju na obrisane n-torke će se postaviti na NULL

Imenovanje integritetskih ograničenja

- ▶ Naziv integritetskog ograničenja se navodi opcionalno
 - ▶ Korisnik ili aplikacija će pri pokušaju obavljanja naredbe koja narušava integritetsko ograničenje dobiti informaciju o kojem se tačno integritetskom ograničenju radi

```
CREATE TABLE ispit (  
    matbr      INTEGER  
,sifPred     INTEGER  
,datIsp      DATE  
,ocj         SMALLINT CONSTRAINT ocjNotNull NOT NULL  
,sifNas      INTEGER  
,CONSTRAINT pkIspit PRIMARY KEY (matbr, sifPred, datIsp)  
,CONSTRAINT fkIspitStud FOREIGN KEY (matbr) REFERENCES student(matbr)  
,CONSTRAINT fkIspitPred FOREIGN KEY (sifPred) REFERENCES predmet(sifPred)  
,CONSTRAINT fkIspitNas FOREIGN KEY (sifNas) REFERENCES nastavnik(sifNas)  
,CONSTRAINT chkOcj CHECK (ocj BETWEEN 1 AND 5)  
);
```

U MySQL-u nije moguće imenovanje
na mjestu definicije atributa



- ▶ Integritetsko ograničenje se može ukloniti **ALTER TABLE** naredbom:
 - ▶ ALTER TABLE ispit DROP CONSTRAINT chkOcj;

Ograničenja integriteta – za atribut

CREATE TABLE tableName

({columnName dataType [DEFAULT defaultExpr] [columnConstraint [, ...] |
tableConstraint} [, ...])

Column constraints:

[CONSTRAINT constraintName]

{NOT NULL | UNIQUE | PRIMARY KEY | CHECK (expression) |

REFERENCES reftable [(refcolumn)] [ON DELETE action] [ON UPDATE action] }

action: NO ACTION, CASCADE, SET NULL, SET DEFAULT

expression – odnosi se samo na dotičnu kolonu, rezultat mora biti logička
vrijednost (boolean)

Ograničenja integriteta – za tablicu

```
CREATE TABLE tableName  
( {columnName dataType [DEFAULT defaultExpr] [ columnConstraint ] [, ...]  
  | tableConstraint} [, ...] )
```

Table constraints:

```
[CONSTRAINT constraintName]  
{UNIQUE (columnName [, ...]) |  
  PRIMARY KEY (columnName [, ...]) |  
  CHECK (expression) |  
  FOREIGN KEY (columnName [, ...]) REFERENCES reftable [(refcolumn [, ...])]  
    [ ON DELETE action] [ON UPDATE action] }
```