

Ključevi, indeksi, spajanje tabela

Ključevi

- Primarni ključ je kolona ili skup kolona koji jednoznačno određuju ostatak podataka u svakom redu
- Primer u tabeli Radnici – IDRadnika je primarni ključ
- Ovo znači dve stvari: nijedna dva reda ne smeju imati istu vrednost u koloni IDRadnika, kao i da čak ukoliko dva radnika imaju isto ime i prezime, kolona IDRadnika obezbeđuje da oni neće biti pomešani jer se za manipulisanje njima u celoj bazi koristi kolona IDRadnika, a ne njihova imena.

Strani ključ

- **Strani ključ** je kolona u tabeli koja je primarni ključ u drugoj tabeli, što znači da sve vrednosti u koloni stranog ključa moraju imati odgovarajuće podatke u drugoj tabeli u kojoj je ta kolona primarni ključ.
- U terminologiji relacionih baza podataka, ova veza se naziva ***referencijalni integritet***.

Spajanje tabela

- Namena ključeva je da podaci iz više tabela mogu da se kombinuju, bez potrebe da se podaci ponavljaju u svim tabelama – u ovome se sastoji snaga relacionih baza podataka.
- Na primer, možete da pronađete imena onih koji su kupili stolicu bez moranja da navedete njihova puna imena u tabeli Antikviteti. To ime možete da saznate povezujući one koji su kupili stolicu sa imenima u tabeli VlasniciAntikviteta preko kolone IDVlasnika, koja *uspostavlja odnos* između podataka u ovim dvema tabelama. Da biste dobili imena onih koji su kupili stolicu, koristite ovaj upit:
- ```
SELECT PREZIMEVLASNIKA, IMEVLASNIKA
FROM VLASNICIANTIKVITETA, ANTIKVITETI
WHERE IDKUPCA = IDVLASNIKA AND KOMADNAMEŠTAJA =
'stolica';
```

# Indeksi

- Bez obzira na organizaciju tabele, u tabelama koje sadrže veliki broj slogova naći pojedini slog nije lako – za tabelu od 1000 ili više slogova pretraživanje red po red trajalo bi isuviše dugo. Da bi se obezbedilo brže pretraživanje i sortiranje tabele, u bazama podataka formiraju se posebne tabele indeksa. Na primer, ako često našu tabelu firmi pretražujemo po mestu možemo da formiramo indeks za polje Mesto. Analogija se može napraviti sa indeksom pojmova na kraju knjige – nađete traženu reč u indeksu pojmova i zatim brzo pristupite stranici na kojoj se nalazi.
- Indeks (Index) omogućuje **brzo pretraživanje i sortiranje tabela**. Indeksi se mogu kreirati za jedno polje, kao i za kombinaciju polja. Pored polja za koje se očekuje često pretraživanje i sortiranje, indekse obično kreiramo i za polja koja koristimo za povezivanje sa drugim poljima u tabeli (spoljni ključevi). Primarni ključ tabele je automatski indeksiran. Indeksi koji se sastoje iz više polja koriste se u slučaju kad pretraživanje i sortiranje samo po jednom polju daje veliki broj slogova, npr. možemo formirati indeks od polja Ime i Prezime u tabeli Partneri iz našeg primera.

# ..IS NULL

- Na početku smo opisali kolona ulica u tabeli Radnik je takva da nije nužno upisati podatak. Ipak poželjno je znati za koga nemamo podatke u tablici, a upravo to nam daje *IS NULL*.
- *SELECT ime, prezime FROM radnik WHERE ulica IS NULL*

- Ipak postoji razlika između NULL i praznog skupa znakova pa moramo staviti još uslov koji kaže da ulica može biti prazan skup znakova.
- *SELECT ime, prezime FROM radnik WHERE ulica ='' OR ulica IS NULL*

# ORDER BY

- Uz korišćenje ove naredbe možemo rezultat sortirati po nekom redosledu
- *SELECT položaj, plata, ime, prezime  
FROM radnik ORDER BY plata*

- *SELECT* položaj, plata, ime, prezime  
*FROM* radnik *ORDER BY* plata, prezime

*Ovom naredbom dobili bi smo rezultat  
sortiran od najniže ka najvišoj plati, a  
zatim po prezimenu*

- ORDER BY ASC – za rastući niz
- ORDER BY DESC – za opadajući niz
- *SELECT položaj, plata, ime, prezime  
FROM radnik ORDER BY plata DESC,  
prezime ASC*

# DISTINCT, ALL

- Da bi smo videli sve kategorije položaja u tabeli radnik uneli bi
- `SELECT položaj FROM radnik`
- Pri tome neke kategorije položaja se ponavljaju, npr. Imamo više administratora
- Gornja naredba je skraćeni oblik:
- `SELECT ALL položaj FROM radnik`
- ..i daje iste rezultate

- Da bi smo videli u rezultatima samo kategorije u koloni položaj potrebno je koristiti ključnu reč **DISTINCT**
- *SELECT DISTINCT položaj FROM radnik*

# FUNKCIJE SAKUPLJANJA

U funkcije sakupljanja spadaju

- *SUM* (suma svih vrednosti),
- *AVG* (prosečna vrijednost),
- *COUNT* (broj redova dobijenih rezultata),
- *MAX* (maksimalna vrijednost u izrazu),
- *MIN* (minimalna vrijednost u izrazu),
- *STDEV* (standardna devijacija),
- *VAR* (varijansa).

- `SELECT SUM(plata) FROM radnik`
- Rezultat je suma svih plata radnika koji se nalaze u datoj tabeli
- `SELECT SUM(plata), AVG(plata), COUNT(plata), MAX(plata), MIN(plata) FROM radnik`
- Rezultat je suma svih plata, prosečna plata, broj plata koje su unete u tabelu, minimalna plata i maksimalna plata

# MATEMATIČKE FUNKCIJE

Matematičke funkcije su:

- *ABS* (apsolutna vrijednost),
- *ACOS* (arkus kosinus),
- *ASIN* (arkus sinus),
- *ATAN* (arkus tangens),
- *COS* (kosinus),
- *COT* (kotangens),
- *LOG* (logaritam baze 2),
- *LOG10* (logaritam baze 10),
- *PI* (3,14),
- *POW*(stepenovanje),
- *RADIANS* (pretvara stepene u radijane),
- *RAND* (generator slučajnih brojeva),
- *ROUND* (zaokruživanje decimalnih brojeva na zadatu preciznost),
- *SIN* (sinus),
- *SQRT* (drugi koren),
- *TAN* (tangens).
- Parametar koji dolazi iza funkcije nalazi se u zagradama, ako ga funkcija trani inače imamo prazne zagrade kao kod *RAND()*.

- `SELECT SUM(plata), AVG(plata), COUNT(plata), MAX(plata), MIN(plata), SQRT(plata) FROM radnik`
- *Osim prethodno izračunatih funkcija saklupljanja prikazuje i kvadratni koren ukupnih plata*
- `SELECT POW(AVG(plata), 2) FROM radnik`
- *Rezultat je kvadrat prosečne plate*

# ARITMETIČKE FUNKCIJE

- Sabiranje +
- Oduzimanje –
- Deljenje /
- Množenje \*
- Moduo  $M\%N$
- Itd.

- +
- Addition:  
mysql> **SELECT 3+5;** -> 8
- -
- Subtraction:  
mysql> **SELECT 3-5;** -> -2
- -
- Unary minus. This operator changes the sign of the argument.  
mysql> **SELECT - 2;** -> -2 **Note:** If this operator is used with a BIGINT, the return value is also a BIGINT. This means that you should avoid using – on integers that may have the value of –263.
- \*
- Multiplication:  
mysql> **SELECT 3\*5;** -> 15 mysql> **SELECT 18014398509481984\*18014398509481984.0;** -> 324518553658426726783156020576256.0 mysql> **SELECT 18014398509481984\*18014398509481984;** -> 0 The result of the last expression is incorrect because the result of the integer multiplication exceeds the 64-bit range of BIGINT calculations. (See [Numeric Types.](#))
- /
- Division:  
mysql> **SELECT 3/5;** -> 0.60 Division by zero produces a NULL result:  
mysql> **SELECT 102/(1-1);** -> NULL A division is calculated with BIGINT arithmetic only if performed in a context where its result is converted to an integer.
- DIV
- Integer division. Similar to FLOOR(), but is safe with BIGINT values.  
mysql> **SELECT 5 DIV 2;** -> 2
- $N \% M$
- Modulo operation