

# Baze podataka

## SQL

# SQL (Structured Query Language)

SQL je strukturirani upitni jezik koji omogućava pristup podacima u sistemima za upravljanje relacionim bazama podataka (MySQL, Oracle, Microsoft SQL Server i drugi)

Korisnici mogu da:

definišu šemu baze podataka,  
smeste podatke u tako definisanu bazu,  
manipulišu tim podacima (obrišu, izmene ili dodaju nove slogove),  
opisuju kakve podatke žele da dobiju iz baze.

# SQL - DML i DDL

SQL možemo podeliti na **dva dela**:

Data Definition Language (**DDL**)

Data Manipulation Language (**DML**)

Naredbe SQL-a se završavaju simbolom tačka-zarez ( )

SQL ne razlikuje mala i velika slova  
(*SELECT* je isto što i *select*)

# SQL - DDL

DDL deo SQL-a omogućava :

kreiranje i izmene baze,

kreiranje, izmene i brisanje tabela baze,

definisane ključeva i indeksa,

definisane veza između tabela i

definisane ograničenja između tabela.

# SQL - DDL

Najvažniji iskazi u DDL-u su:

**CREATE DATABASE** – kreiranje nove baze

**ALTER DATABASE** – izmena baze

**CREATE TABLE** – kreiranje tabele

**ALTER TABLE** – izmena tabele

**DROP TABLE** – brisanje tabele

**CREATE INDEX** – kreiranje indeksa (ključa za pretraživanje)

**DROP INDEX** – brisanje indeksa

# SQL - DML

**DML** definiše sledeće naredbe za ažuriranje i pretraživanje baze podataka :

komande za ažuriranje:

**INSERT INTO** – unos novih slogova u bazu

**UPDATE** – izmene na podacima u bazi

**DELETE** – brisanje slogova iz baze

komanda za pretraživanje (kreiranje upita):

**SELECT** – prikaz izabranih podataka (slogova) iz baze

# iskaz INSERT INTO

Koristi se za unos novog reda u tabelu, bilo da se unose vrednosti svih kolona (A) ili samo izabranih kolona (B).

- A. Unos podataka u sve kolone (2 oblika iskaza INSERT):

Navode se samo vrednosti koje se unose:

```
INSERT INTO ime_tabele  
VALUES (vrednost1, vrednost2, vrednost3,...)
```

Navode se imena kolona u koja se unose podaci i odgovarajuće vrednosti koje se unose:

```
INSERT INTO ime_tabele (kolona1, kolona2, kolona3,...)  
VALUES (vrednost1, vrednost2, vrednost3,...)
```

- B. Unos podataka u tačno određene kolone (ne u sve)

Tada se u listi kolona navode samo željene kolone

# iskaz UPDATE

Vrši izmenu postojećih redova u tabeli

```
UPDATE ime_tabele  
  SET kolona1=vrednost1, kolona2=vrednost2,...  
  WHERE kolona=vrednost
```

WHERE klauzulu koristimo da bismo odredili koje tačno redove menjamo

Ako izostavimo ovaj uslov, izmena će se izvršiti za sve redove table



# iskaz DELETE

Briše redove iz tabele koji zadovoljavaju uslov dat sa WHERE klauzulom:

```
DELETE FROM ime_tabele  
WHERE kolona=vrednost
```

WHERE klauzulu koristimo da bismo odredili koje tačno redove brišemo.

Ako izostavimo WHERE uslov, svi redovi tabele će biti obrisani:

```
DELETE FROM ime_tabele
```

ili

```
DELETE * FROM ime_tabele
```

# iskaz SELECT

Prikaz izabranih podataka iz baze:

```
SELECT * FROM ime_tabele
```

```
SELECT lista_kolona FROM ime_tabele
```

gde je

```
lista_kolona: ime_kol1, ime_kol2, ...
```

```
SELECT DISTINCT lista_kolona FROM ime_tabele
```

( DISTINCT je ključna reč za prikaz različitih podataka )

Rezultati izvršavanja SELECT iskaza se smeštaju u tabelu koja se zove **result-set**

# klauzula WHERE

WHERE klauzula služi za izdvajanje samo onih slogova koji zadovoljavaju postavljeni uslov:

```
SELECT lista_kolona
```

```
FROM ime_tabele
```

```
WHERE ime_kolone operator vrednost
```

Operatori mogu da se primenjuju na različite tipove podataka (numeričke, tekstualne, logičke, datumske).

# Operatori u klauzuli WHERE

## Relacioni operatori:

= jednako, != različito, > veće, < manje,  
>= veće ili jednako, <= manje ili jednako.

## Logički operatori:

IN - u skupu konkretnih vrednosti,

AND - logički operator i,

OR - logički operator ili.

BETWEEN - u zadanom opsegu,

LIKE - pronalaženje sličnog ili istog.

# klauzula ORDER BY

Ključna reč za sortiranje rezultata upita po nekoj koloni

```
SELECT ime_kolone/a  
FROM ime_tabele  
ORDER BY ime_kolone/a ASC|DESC
```

Po default-u, rezultati se sortiraju po rastućem redosledu (ASC)

Ako želimo opadajući redosled sortiranja, koristimo ključnu reč DESC

# klauzula LIMIT

```
SELECT ime_kolone/a  
FROM ime_tabele  
LIMIT number
```

Koristi se za određivanje broja redova koje vraća postavljeni upit.

Korisna je kod tabela koje imaju veliki broj redova, jer upiti mogu usporiti rad aplikacije.

# LIKE operator

Služi za pretraživanje i pronalaženje određenih kolona u tabeli.

Znak % služi za zamenu jednog ili više nepoznatih karaktera pri pretraživanju baze podataka.

# IN operator

```
SELECT ime_kolone/a  
FROM ime_tabele  
WHERE ime_kolone  
      IN (vrednost1,vrednost2,...)
```

Omogućava zadavanje skupa vrednosti u WHERE klauzuli.



# BETWEEN operator

```
SELECT ime_kolone/a  
FROM ime_tabele  
WHERE ime_kolone  
BETWEEN vrednost1 AND vrednost2
```

Za označavanje skupa podataka između dve zadate vrednosti u WHERE klauzuli

Te vrednosti mogu biti brojevi, tekst ili datumi

U MySQL-u **BETWEEN** operator u skup traženih podataka ubraja i granične vrednosti

**NOT BETWEEN** označava skup podataka **izvan intervala** obuhvaćenog graničnim vrednostima ove naredbe

# SQL FUNKCIJE

SQL agregatne funkcije vraćaju vrednost, izračunatu iz svih vrednosti u koloni koja je zadata kao argument:

AVG() – prosečna vrednost

COUNT() – broj redova

FIRST() – prva vrednost

LAST() – poslednja vrednost

MAX() – najveća vrednost

MIN() – najmanja vrednost

SUM() – zbir vrednosti

# SQL FUNKCIJE

SQL skalarne funkcije vraćaju vrednost, na osnovu vrednosti koja se zadaje kao argument funkcije:

UCASE() – konvertuje u velika slova

LCASE() – konvertuje u mala slova

MID() – izdvaja karaktere iz tekstualnog polja

LEN() – vraća dužinu tekstualnog polja

ROUND() – zaokružuje numerička polja na određeni broj decimala

NOW() – vraća tekući sistemski datum i vreme

FORMAT() – formatiranje prikaza

# klauzula GROUP BY

Koristi se u kombinaciji sa agregatnim funkcijama da bi se rezultati grupisali po jednoj ili više kolona:

```
SELECT ime_kolone, agregatna_funkcija(ime_kolone)
FROM ime_tabele
WHERE ime_kolone operator vrednost
GROUP BY ime_kolone
```

Ako vršimo grupisanje po više kolona, tada ih navodimo iza GROUP BY razdvojene zarezima:

```
SELECT ....
GROUP BY ime_kolone1, ime_kolone2,...
```

# klauzula HAVING

Ova klauzula je dodata u SQL jer WHERE ne može da se koristi sa agregatnim funkcijama

```
SELECT ime_kolone, agregatna_funkcija(ime_kolone)
FROM ime_tabele
WHERE ime_kolone operator vrednost
GROUP BY ime_kolone
HAVING agregatna_funkcija(ime_kolone) operator
vrednost
```