

Visoka škola strukovnih studija za  
informacione i komunikacione tehnologije  
Beograd

MySQL & SQL server

Kreiranje baze podataka

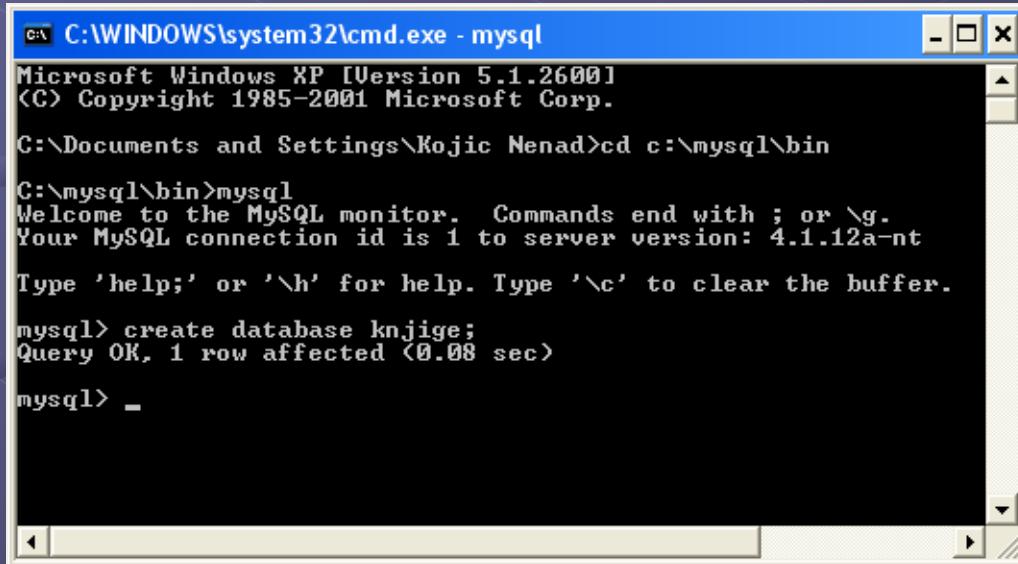
Dr Nenad Kojić

# Kreiranje baze podataka

- Nakon prijave na MySQL server, u komandnoj liniji treba kucati naredbu za kreiranje baze podataka

**CREATE DATABASE IMEBAZE;**

Ako se kreira baza "knjige"



The screenshot shows a Windows XP command prompt window titled 'C:\WINDOWS\system32\cmd.exe - mysql'. The window displays the following MySQL session:

```
C:\> C:\WINDOWS\system32\cmd.exe - mysql
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\>Documents and Settings\Kojic Nenad>cd c:\mysql\bin

C:\mysql\bin>mysql
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 4.1.12a-nt
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database knjige;
Query OK, 1 row affected (0.08 sec)

mysql> _
```

Odgovor:

*Query OK, 1 row affected*

označava da je proces uspešno obavljen.

# Kreiranje tabele baze podataka

- Nakon kreiranja baze treba kreirati njene tabele. Svaka tabela se posebno definiše. Sintaksa je:

```
CREATE TABLE imetabele (kolone);
```

- Ime tabele definiše svaku od tabela, dok kolone koje ta tabela sadrži treba nabrajati jednu iza druge i imena odvajati zarezima.
- Svaka kolona treba da sadrži ime i tip podataka u njoj.

# Primer kreiranja tabele

```
mysql> use knjige
```

```
create table kupci (
```

```
IDkupca int unsigned not null auto_increment primary key,
```

```
ime char(50) not null,
```

```
grad char(30) not null,
```

```
naslov char(100),
```

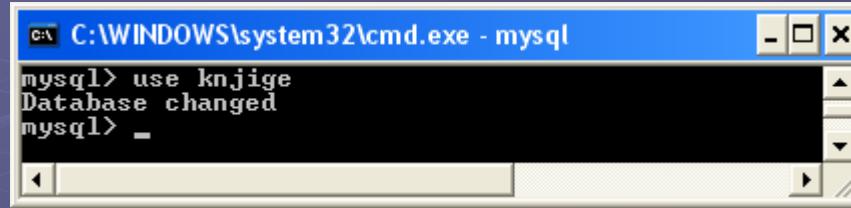
```
iznos float(5,2),
```

```
kolicina tinyint unsigned,
```

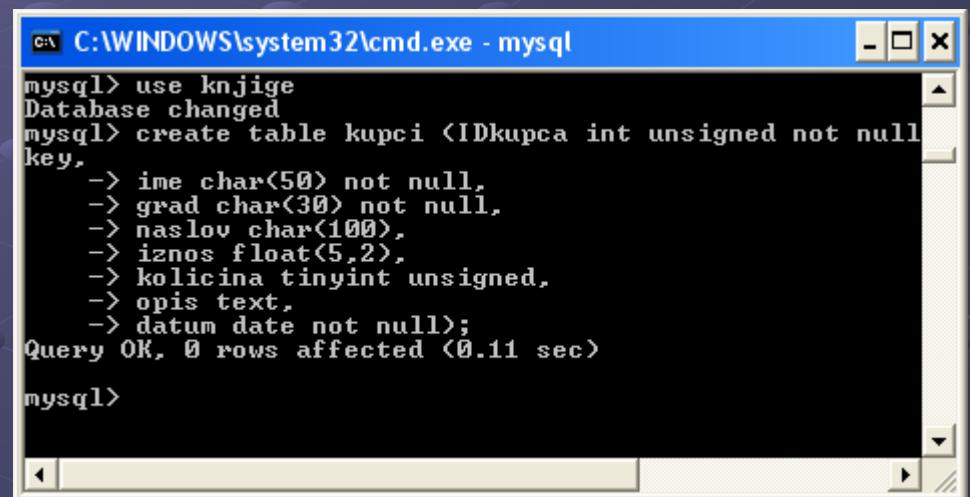
```
opis text,
```

```
datum date not null
```

```
);
```

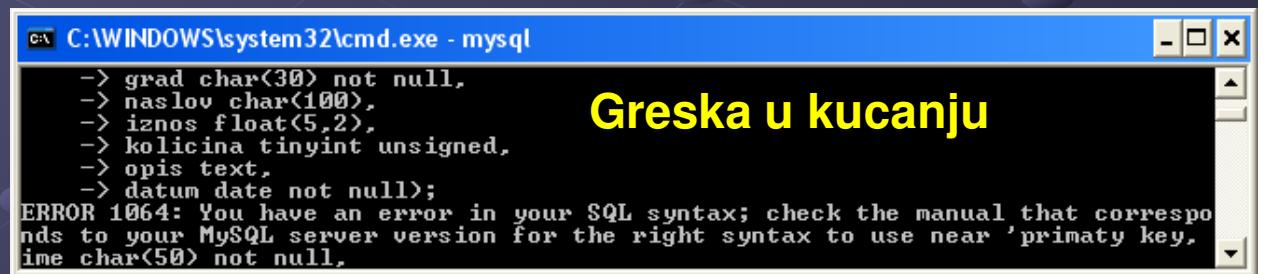


```
C:\WINDOWS\system32\cmd.exe - mysql
mysql> use knjige
Database changed
mysql> -
```



```
C:\WINDOWS\system32\cmd.exe - mysql
mysql> use knjige
Database changed
mysql> create table kupci (IDkupca int unsigned not null
key,
-> ime char(50) not null,
-> grad char(30) not null,
-> naslov char(100),
-> iznos float(5,2),
-> kolicina tinyint unsigned,
-> opis text,
-> datum date not null);
Query OK, 0 rows affected (0.11 sec)

mysql>
```

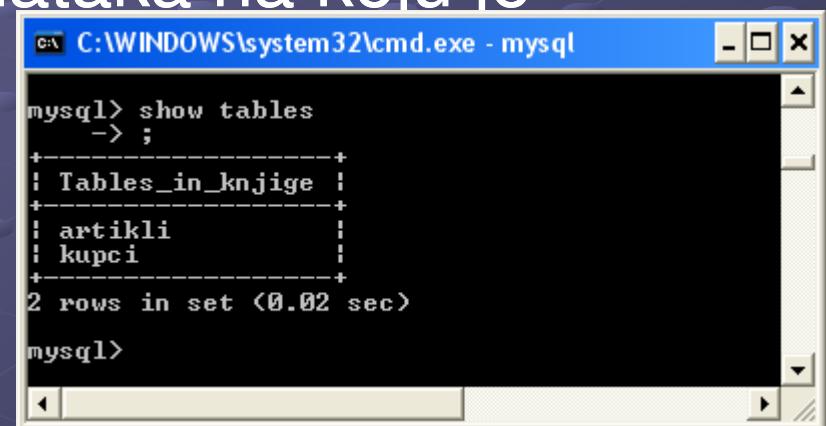


Greska u kucanju

```
C:\WINDOWS\system32\cmd.exe - mysql
-> grad char(30) not null,
-> naslov char(100),
-> iznos float(5,2),
-> kolicina tinyint unsigned,
-> opis text,
-> datum date not null);
ERROR 1064: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'primary key,
ime char(50) not null,
```

# Prikaz informacija o bazi

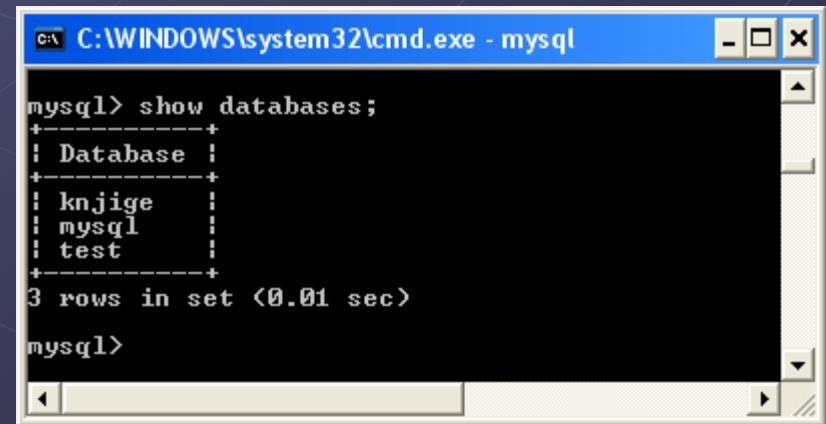
- SHOW TABLES - komanda prikazuje sve kreirane tabele u bazi podataka na koju je izvršena prijava



```
mysql> show tables
-> ;
+-----+
| Tables_in_knjige |
+-----+
| artikli           |
| kupci             |
+-----+
2 rows in set <0.02 sec>

mysql>
```

- SHOW DATABASES - prikazuje spisak svih baza na serveru

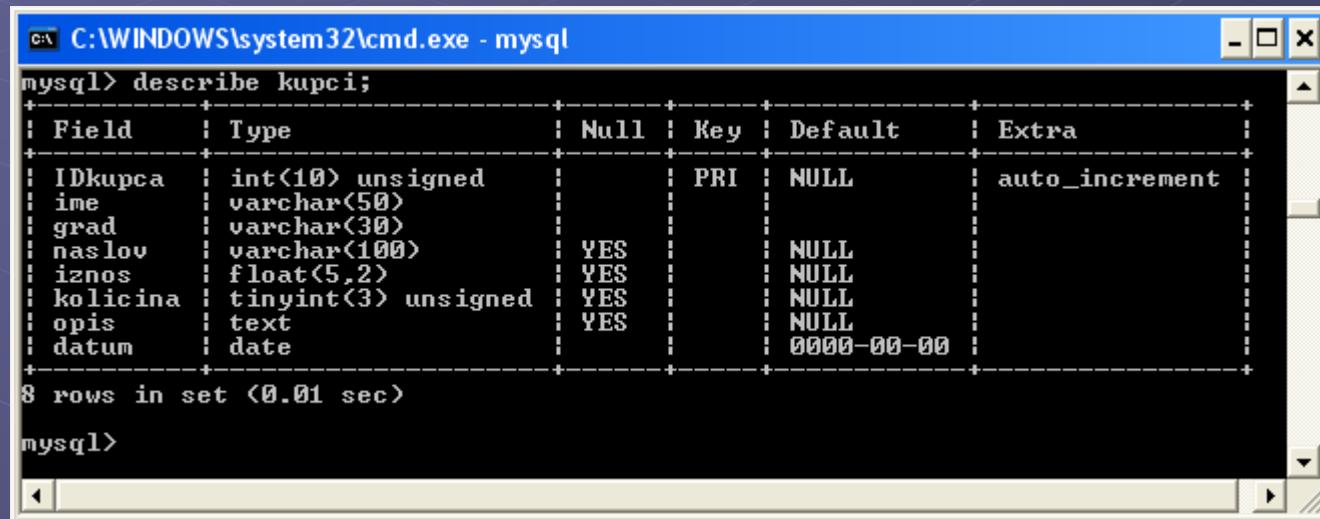


```
mysql> show databases;
+-----+
| Database |
+-----+
| knjige   |
| mysql    |
| test     |
+-----+
3 rows in set <0.01 sec>

mysql>
```

# Prikaz informacija o tabeli

- DESCRIBE imetabele - Precizno prikazuje sve informacije za traženu tabelu u bazi



The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe - mysql". The user has run the command "mysql> describe kupci;". The output is a table showing the structure of the "kupci" table:

Field	Type	Null	Key	Default	Extra
IDkupca	int<10> unsigned		PRI	NULL	auto_increment
ime	varchar(50)				
grad	varchar(30)				
naslov	varchar(100)	YES		NULL	
iznos	float<5,2>	YES		NULL	
kolicina	tinyint<3> unsigned	YES		NULL	
opis	text	YES		NULL	
datum	date			0000-00-00	

Below the table, the message "8 rows in set (0.01 sec)" is displayed. The MySQL prompt "mysql>" is visible at the bottom of the window.

# Izrada indeksa

- Ako se u toku rada zahteva veliki broj pretraga po koloni koja nije indeksirana, onda će se poboljšanje performansi poboljšati dodavanjem indeksa za tu kolonu.

```
CREATE [UNIQUE|FULLTEXT] INDEX imeindeksa
ON imetabele (imekolone [(broj)] [ASC|DESC], ... );
```

- Broj je opcioni parametar i služi za indeksiranje samo prvih n znakova u polju.
- [ASC|DESC] definišu da li je indeksiranje izvršeno rastućim ili opadajućim redosledom. Inicijalno je rastući.

# Unošenje podataka u bazu 1

- Unošenje podataka vrši se naredbom INSERT
  - Sintaksa naredbe:

`INSERT [INTO] tabela [(kolona1, kolona2, kolona3, ...)]  
VALUES(vrednost1, vrednost2, vrednost3, ...);`

- Vrednosti kolona pisati pod znacima ' ili ", sem ako su brojevi ili datumi.
  - Ako se upisuje ceo red (sve kolone) sintaksa je:

`INSERT [INTO] tabela  
VALUES(vrednost1, vrednost2, vrednost3, ...);`

Primer:

```
insert into kupci values (NULL, 'Pera Peric', 'Beograd', 'Svet oko nas', 2000, 1,  
'knjiga za decu', 2007-03-01);
```

# Unošenje podataka u bazu 2

- Upis se može vršiti i za pojedine kolone proizvoljnim redosledom

Primer:

```
insert into kupci (naslov, kolicina, grad) values ('Mali princ', 2, 'Vranje');
```

- Isti efekat dobija se primenom set-a kao

Primer:

```
insert into kupci  
set naslov= 'Mali princ', kolicina= 2, grad= 'Vranje' ;
```

- Ako je kolona definisana kao auto\_increment, tada je kod upisa dovoljno samo prvi put, definisati njenu vrednost

# Unošenje podataka u bazu 3

- U tabelu se može uneti i više redova odjednom

Primer:

```
insert into kupci values
```

```
(2, 'Pera Peric', 'Beograd', 'Svet oko nas', 2000, 1, 'knjiga za decu', '2007-03-01') ,  
(3, 'Mile Vasic', 'Sabac', 'Geografija', 2500, 2, 'knjiga za omladinu', '2007-03-05') ,  
(4, 'Ana Savic', 'Valjevo', 'Bukvar', 1500, 1, 'knjiga za decu', '2007-03-15') ;
```

- Insert low\_priority - privremeno zaustavljanje sa umetanjem novog reda dok se ne završe tekuća čitanja
- Insert delayed. low\_priority - smeštanje podataka za umetanje u keš, i zadrška dok se server ne rastereti.

# Čitanje podataka iz baze

- Naredba za čitanje je SELECT sa sintaksom:

*SELECT podaci FROM tabela  
[WHERE uslovi] [GROUP BY grupisanje]  
[HAVING uslovgrupe] [ORDER BY redosled]  
[LIMIT ograničenja]  
[PROCEDURE imeprocedure(argumenti)]  
[način zaključavanja];*

# Čitanje pomoću select - from

- Čitanje sadržaja kolona ime i grad iz tabele kupci

Primer:

```
select ime, grad from kupci ;
```

```
c:\ C:\WINDOWS\system32\cmd.exe - mysql
mysql> select ime, grad from kupci;
+-----+-----+
| ime   | grad   |
+-----+-----+
| Vera Peric | Uranje |
| Pera Peric | Beograd |
+-----+-----+
2 rows in set (0.03 sec)

mysql>
```

- Čitanje svih kolona iz tabele

Primer:

```
select * from kupci ;
```

```
c:\ C:\WINDOWS\system32\cmd.exe - mysql
mysql> select * from kupci;
+-----+-----+-----+-----+-----+-----+-----+
| IDkupca | ime      | grad    | naslov      | iznos   | kolicina | opis
| datum   |           |          |             |         |          |          |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | Vera Peric | Uranje | Matematika | 999.99 | 2 | knjiga za
osnovnu skolu | 2007-03-01 |
| 2 | Pera Peric | Beograd | Svet oko nas | 999.99 | 1 | knjiga za
decu | 2007-03-01 |
+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

# Čitanje pomoću select - *where*

- Za izdvajanje dela podataka sa uslovom treba koristiti where pa uslov pretrage

Primer:

```
select * from kupci  
where grad = 'Vranje' ;
```

Kombinacija uslova

```
select * from kupci  
where grad = 'Vranje'  
or|and ime='Pera Peric';
```

```
C:\WINDOWS\system32\cmd.exe - mysql  
mysql> select * from kupci where grad ='Beograd' ;  
+-----+-----+-----+-----+-----+-----+  
| IDkupca | ime      | grad     | naslov       | iznos   | kolicina | opis  
| datum    |          |          |              |         |          |  
+-----+-----+-----+-----+-----+-----+  
| 2 | Pera Peric | Beograd | Svet oko nas | 999.99 | 1 | knjiga za  
decu | 2007-03-01 |          |              |         |          |  
+-----+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)  
mysql> _
```

- Operatori za poređenje su: =,<,>,>=,<=,  
!=, is not null, is null, between, in, not in,  
like, not like ...

# Čitanje podataka iz više tabela istovremeno

- Ukoliko se želi pročitati više podataka iz različitih tabela, a koji su na bilo koji način povezani, potrebno je izvršiti spajanje tabela.
- Za dve tabele naći sve porudžbine Pere Perića

Tabela	Kolone			
Kupci	ID kupca	Ime	Adresa	Grad
Narudzbine	ID narudzibe	ID kupca	Iznos	Datum

Primer:

```
select narudzbine.iznos, narudzbine.datum from kupci, narudzbine  
where kupci.ime = 'Pera Peric' and kupci.IDkupca = narudzbine.IDkupca;
```

Uslov spajanja je kupci.IDkupca = narudzbine.ID kupca, može biti i van baze npr  
baza4.nesto1.IDnesto1 = baza2.nesto3.IDnesto3

# Spajanje više od dve kolone

- Koji kupci su naručili knjigu Baze (ISBN = 222333)

Tabela		Kolone		
Kupci	ID kupca	Ime	Adresa	Grad
Narudzbine	ID narudzibene	ID kupca	Iznos	Datum
Proizvodi	ID narudzibene	ISBN broj	Količina	
Knjige	ISBN broj	Autor	Naslov	Cena

Primer:

```
select kupci.ime from kupci, narudzbine, proizvodi, knjige  
where kupci.ID kupca = narudzbine.ID kupca and narudzbine.ID narudzibe =  
proizvodi.IDnarudzbine and proizvodi.ISBN broj = knjige.ISBN broj and  
knjige.naslov like '%Baze%' ;
```

Def: Broj uslova za spajanje tabela je za jedan manji od broja tabela, jer je za spajanje dve tabele potreban jedan uslov.

# Levi spojevi (*left join*)

- Najčešće se koristi za slučajeve kada nema preklapanja između podataka (Naći klijente koji ništa nisu kupili !)
- Def: Ako se prilikom spajanja redova dve tabele, u desnoj tabeli ne nađe red, koji bi odgovarao redu u levoj tabeli, skupu rezultata se dodaje red, koji umesto vrednosti kolone na desnoj strani, sadrži vrednost NULL.
- Sintaksa sadrži rezervisani reč *left join* i *on*

# Primer levog spajanja 1

- Za datu bazu napisati kod kojim se pretražuju korisnici u odnosu na broj narudžbenica.

Kupci

ID Kupca	Ime	Adresa	Grad
1	Mile Vasic	Narodnih heroja 1	
2	Milos Peric	Avijaticara 7	Leskovac
3	Jovan Jovic	Ovcarsrska 55	Arilje
4	Ana Senic	Havajska 5	Bor
5	Milena Ilic	Nade Puric 4	Zajecar

Narudzbine

ID Narudzbine	ID Kupca	Iznos	Datum
1	5	6500	2007-01-05
2	3	4800	2007-01-12
3	4	7300	2007-02-07
4	5	2100	2007-02-19

Primer:

Tabela sa desne strane

```
select kupci.ID kupca, kupci.ime,  
narudzbine.ID narudzbe  
from kupci left join narudzbine  
on kupci.ID kupca = narudzbine.ID kupca;
```

Tabela sa leve strane

ID Kupci	Ime	ID Narudzbine
1	Mile Vasic	NULL
2	Milos Peric	NULL
3	Jovan Jovic	2
4	Ana Senic	3
5	Milena Ilic	1
5	Milena Ilic	4

# Primer levog spajanja 2

- Ako se traže korisnici koji ništa nisu kupili, tada treba naći vrednosti NULL u novodobijenoj koloni, i dati odgovarajuće ime korisnika

ID Kupci	Ime	ID Narudzbine
1	Mile Vasic	NULL
2	Milos Peric	NULL
3	Jovan Jovic	2
4	Ana Senic	3
5	Milena Ilic	1
5	Milena Ilic	4

**Primer:**

```
select kupci.ID kupca, kupci.ime,  
from kupci left join narudzbine
```

```
using (ID kupca)  
where narudzbine. IDnarudzbine is NULL;
```

Upotrebom USING ne navodi se tabela iz koje dolazi zajednička kolona. U tom slučaju mora postojati kolona sa istim imenom u obe tabele.

ID Kupci	Ime
1	Mile Vasic
2	Milos Peric

# Upotreba ALIJASA

- Alijasi su druga imena koja se dodeljuju privremeno ili trajno postojećim kolonama. Najčešće se koriste kao skraćena imena.

Primer:

```
select k.ime,  
from kupci as k, narudzbine as n, proizvodi as p, knjige as knj  
where k.ID kupca = n.ID kupca and n.ID narudzbine= p.ID narudzbine and  
n.ISBN = knj.ISBN and knj.naslov like '%Baze%' ;
```

Primer:

```
select kupci.ime from kupci, narudzbine, proizvodi, knjige  
where kupci. ID kupca = narudzbine. ID kupca and narudzbine. ID narudzibe =  
proizvodi.IDnarudzbine and proizvodi.ISBN broj = knjige.ISBN broj and  
knjige.naslov like '%Baze%' ;
```

# Alijasi za sopstveno spajanje

- Alijasi su obavezni kada treba spojiti tabelu samu sa sobom. Ovo je potrebno ako se traže redovi sa istim vrednostima u nekoj koloni.
- Naći sve kupce koji žive u istom gradu

**Primer:**

```
select k1.ime, k1.grad, k2.ime,  
from kupci as k1, kupci as k2  
where k1.grad = k2.grad and k1.ime != k2.ime;
```

Ovim uslovom se definiše da isti kupac, koji je u obe tabele, ne bude pronađen više puta.  
Može postojati samo jedno ime za taj grad !!!

# Učitavanje po redosledu

- Ako se redovi koje treba prikazati kao rezultat upita žele sortirati koristi se naredba ORDER BY.
- Vrednosti se sortiraju od a ka z, i od 1 na više, dok se za promenu koristi rezervisana reč DESC.

## Primer:

```
select ime, adresa, grad  
from kupci  
order by ime;
```

Ime	Adresa	Grad
Ana Senic	Havajska 5	Bor
Jovan Jovic	Ovcarska 55	Arilje
Mile Vasic	Narodnih heroja 1	
Milena Ilic	Nade Puric 4	Zajecar
Milos Peric	Avijaticara 7	Leskovac

## Primer:

```
select ime, adresa, grad  
from kupci  
order by ime desc;
```

Ime	Adresa	Grad
Milos Peric	Avijaticara 7	Leskovac
Milena Ilic	Nade Puric 4	Zajecar
Mile Vasic	Narodnih heroja 1	
Jovan Jovic	Ovcarska 55	Arilje
Ana Senic	Havajska 5	Bor



# Grupne funkcije

- Grupne funkcije se primenjuju na grupisane podatke (više kolona ili redova u kolonama)
- Neke grupne funkcije su:

Naziv	Opis
AVG (kolona)	Prosečna vrednost kolone
COUNT (stavke)	Ako je navedena kolona daje broj elemenata u koloni bez NULL, a ako se ispred naredbe doda DISTINCT dobije se broj jedinstvenih vrednosti u koloni.
MIN (kolona)	Minimalne vrednost kolone
MAX (kolona)	Maksimalna vrednost kolone
SUM (kolona)	Zbir vrednosti u koloni

# Primeri grupnih funkcija

- Izračunavanje srednje vrednosti porudžbina

**Primer:**

```
select avg(iznos)  
from narudžbine;
```

avg (iznos)
5175

- Izračunavanje srednje vrednosti porudžbina po određenim grupama dobija se grupisanjem po određenoj kategoriji

**Primer:**

```
select IDkupca, avg(iznos)  
from narudžbine;  
group by IDkupca
```

ID kupca	avg (iznos)
1	4800
2	7300
3	4300

- Traženje porudžbina koje premašuju vrednost 4500

**Primer:**

```
select IDkupca, avg(iznos)  
from narudžbine;  
group by IDkupca  
having avg(iznos) > 4500;
```

ID kupca	avg (iznos)
1	4800
2	7300

# Izbor redova koje upit treba da vrati

- Naredbom LIMIT definiše se koji redovi ( počev od kog i koliko njih) treba da budu vraćeni na zahtevani upit.
- Prvi red ima redni broj 0
- Ako se želi prikaz prvih 10 imena, tada je kod

Primer:

```
select name  
from kupci;  
limit 0, 10;
```

- Prikaz 5,6,7,8 i 9-og reda bi bio

Primer:

```
select name  
from kupci;  
limit 4, 5;
```

# Podupiti (subquery)

- Podupit je upit unutar upita
- Koristi se kada se rezultat jednog upita poredi u uslovu drugog upita.

## Primer:

```
select IDkupca, iznos  
from narudzbine;  
where iznos= (select max(iznos) from narudzbine);
```

Vraća se ID  
kupca koji ima  
najveći iznos

ID kupca	iznos
2	7300

Podupit koji traži  
maksimalnu vrednost u  
koloni iznos

- Operatori za podupite

Ime	Primer sintakse	Opis
ANY	SELECT c1 FROM t1 WHERE c1 > ANY (SELECT c1 FROM tv2);	Vraća true ako je rezultat poređenja true za barem jedan red iz podupita. Ekvivalentan operatoru ANY.
IN	SELECT c1 FROM t1 WHERE c1 IN (SELECT c1 FROM tv2);	Alijas operatora ANY.
SOME	SELECT c1 FROM t1 WHERE c1 > SOME (SELECT c1 FROM tv2);	Vraća true ako je za sve redove iz podupita rezultat poređenja true.
ALL	SELECT c1 FROM t1 WHERE c1 > ALL (SELECT c1 FROM tv2);	

# Podupiti koji vraćaju redove

- Podupiti najčešće vraćaju neku logičku promenljivu True ili False, jer su rezultat nekog poređenja.
- Podupit može da vrati i ceo red
- Ovo je najčešći slučaj kada treba naći sve redove jedne tabele koji postoji i u drugoj

## Primer:

```
select k1, k2, k3  
from tabela1;  
where (k1, k2, k3) in (select k1, k2, k3 from tabela2);
```

# Ažuriranje zapisa u bazi

- Često postoji potreba za promenom podataka u bazi (cena artikla, pdv, adresa...)
- Iskaz UPDATE i njegova sintaksa su:

**Primer:**

UPDATE imetabele

Set kolona1=izraz1, kolona2=izraz2, ...

[Where uslovi]

[ORDER by redosled]

[Limit broj]

**Primer povećanja cene knjiga za 10%:**

update knjige

set cena=cena\*1.1;

**Primer promene adrese kupca:**

update kupci

set adresa = ' Bulevar mira 12'

where IDkupca = 3;

# Naknadne izmene strukture tabela

Tabela 10.5 Izmene koje omogućava iskaz ALTER TABLE

Sintaksa	Opis
ADD [COLUMN] <i>opis_kolone</i> [FIRST   AFTER <i>kolona</i> ]	Dodaje novu kolonu na navedenom mestu. Ako mesto umetanja nije navedeno, kolona se dodaje na kraj. Parametar <i>opis_kolone</i> definiše ime kolone i tip podataka, kao u iskazu CREATE.
ADD [COLUMN] ( <i>opis_kolone</i> , <i>opis_kolone</i> ...)	Dodaje jednu ili više kolona na kraj tabele.
ADD INDEX [ <i>indeks</i> ] ( <i>kolona</i> ,...)	Dodaje indeks po navedenoj koloni ili kolonama.
ADD [CONSTRAINT] [ <i>simbol</i> ]	Od navedene kolone ili kolona formira se primjerice ključevi

# Primer izmene strukture tabela

## Primer:

- Promena tipa podatka u koloni tako da prihvata imena max dužine 90 karaktera

**alter table kupci**

modify ime char(90) not null;

- Dodavanje nove kolone (pdv) iza kolone iznos u tabeli porudžbine

**alter table porudzbine**

add pdv float (5,2) after iznos

- Brisanje kolone pdv

**alter table porudzbine**

**drop pdv;**

# Brisanje

## ● Brisanje zapisa (redova) - Delete

### Sintaksa:

```
DELETE [low_priority] [quit] [ignore] from imetabele  
[WHERE uslov]  
[ORDER BY kolone]  
[LIMIT broj]
```

Odlaganje do rasterećenja

Ubrzavanje postupka

Izvršenja promene bez  
obaveštenja

Uslov pod kojim se briše

Grupisanje kolona

Broj redova koji se brišu

### Primer:

```
DELETE from kupci  
WHERE ime='Pera';
```

## ● Brisanje tabela - Drop

### Primer:

```
DROP TABELE (naziv tabele koja se briše);
```

## ● Brisanje cele baze podataka - Drop

### Primer:

```
DROP DATABASE (naziv baze podataka koja se briše);
```

# DDL i DML

- DDL - Data Definition Language je jezik za definisanje podataka i odnosi se na naredbu Create Table.
- DML - Data Manipulation Language je jezik za manipulisanje podacima, i odnosi se na naredbe Select, Update, Insert i Delete.
- QML - Query Manipulation Language je jezik za manipulisanje upitima i odnosi se na naredbu Select

Visoka škola strukovnih studija za  
informacione i komunikacione tehnologije  
Beograd

MySQL & SQL server

Kreiranje baze podataka

Dr Nenad Kojić