COP 4710: Database Systems
Fall 2006

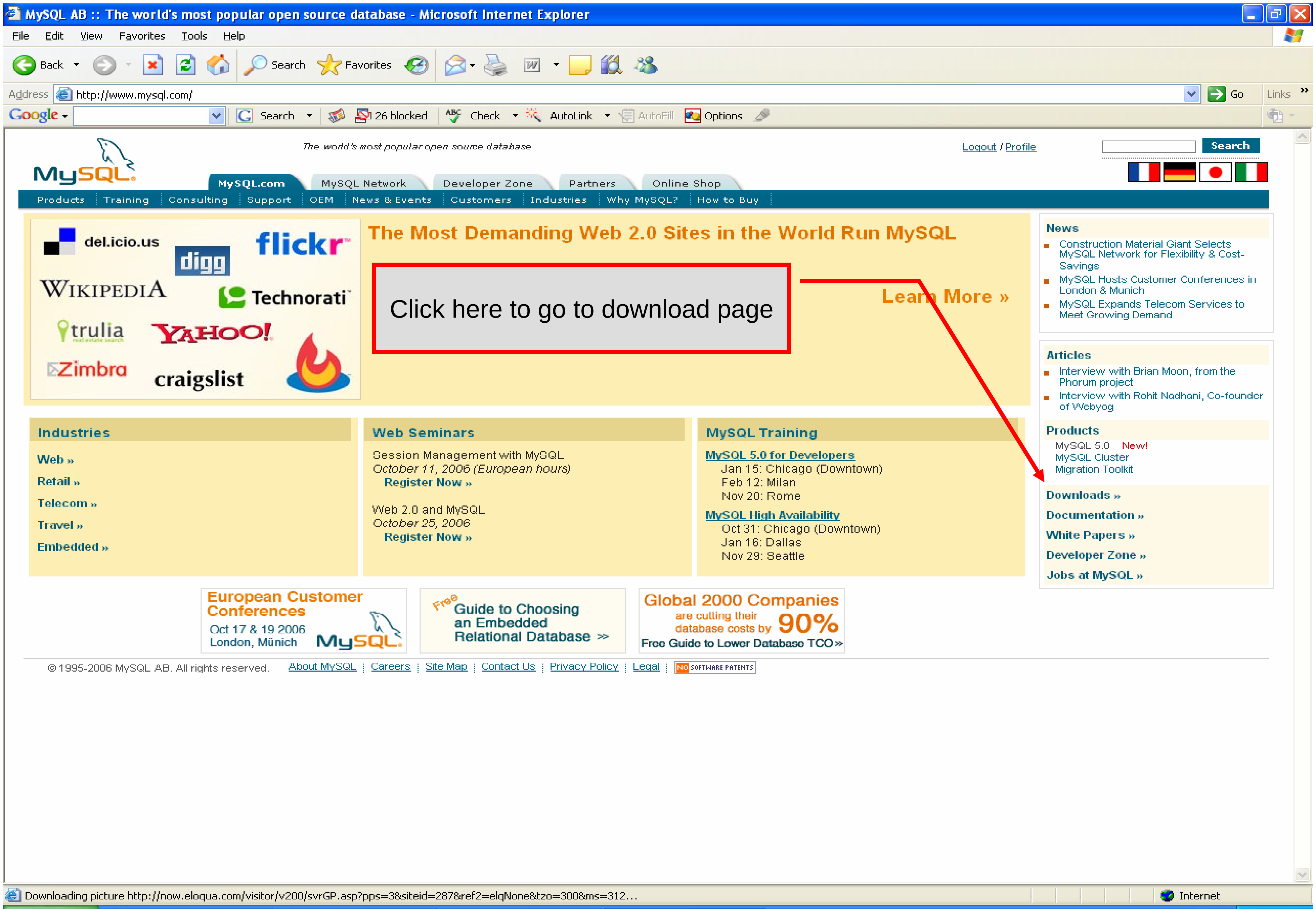
Introduction To MySQL

Instructor :	Mark Llewellyn
	markl@cs.ucf.edu
	ENG3 236, 823-2790
	http://www.cs.ucf.edu/courses/cop4710/fall2006

School of Electrical Engineering and Computer Science
University of Central Florida

MySQL RDBMS

- MySQL is a database server (although it does come with a set of simple client programs). The current stable version is 5.0.24 and can be downloaded from www.mysql.com. (Any of the versions of MySQL 5.0.15 or greater will be fine for our purposes.)
- It is typically used in thin client environments. In other words, it is used in client-server systems where the bulk of the processing and storage takes place on the server, and the client is little more than a dumb terminal.
- MySQL performs multithreaded processing, which means that multiple clients are allowed to connect to it and run queries simultaneously. This makes MySQL extremely fast and well suited to client-server environments such as Web sites and other environments that process numerous transactions for multiple users.



MySQL AB :: The world's most popular open source database - Microsoft Internet Explorer

http://www.mysql.com/

The world's most popular open source database

MySQL.com | MySQL Network | Developer Zone | Partners | Online Shop

Products | Training | Consulting | Support | OEM | News & Events | Customers | Industries | Why MySQL? | How to Buy

The Most Demanding Web 2.0 Sites in the World Run MySQL

Click here to go to download page

Learn More »

News

- Construction Material Giant Selects MySQL Network for Flexibility & Cost-Savings
- MySQL Hosts Customer Conferences in London & Munich
- MySQL Expands Telecom Services to Meet Growing Demand

Articles

- Interview with Brian Moon, from the Phorum project
- Interview with Rohit Nadhani, Co-founder of Webyog

Products

MySQL 5.0 New!
MySQL Cluster
Migration Toolkit

Downloads »
Documentation »
White Papers »
Developer Zone »
Jobs at MySQL »

Industries

Web »
Retail »
Telecom »
Travel »
Embedded »

Web Seminars

Session Management with MySQL
October 11, 2006 (European hours)
Register Now »

Web 2.0 and MySQL
October 25, 2006
Register Now »

MySQL Training

MySQL 5.0 for Developers
Jan 15: Chicago (Downtown)
Feb 12: Milan
Nov 20: Rome

MySQL High Availability
Oct 31: Chicago (Downtown)
Jan 16: Dallas
Nov 29: Seattle

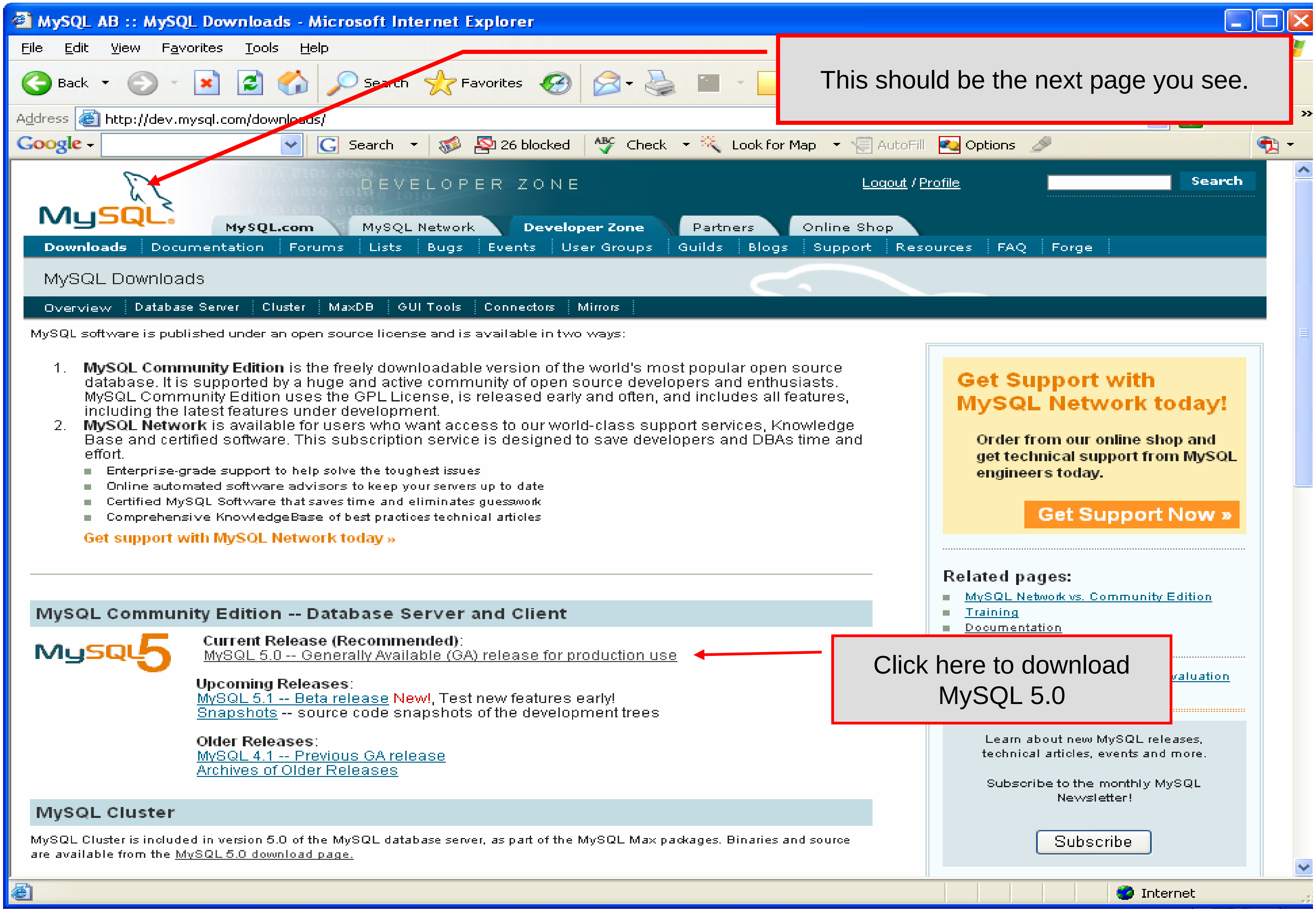
European Customer Conferences Oct 17 & 19 2006 London, München

Free Guide to Choosing an Embedded Relational Database >>

Global 2000 Companies are cutting their database costs by 90% Free Guide to Lower Database TCO »

© 1995-2006 MySQL AB. All rights reserved. About MySQL | Careers | Site Map | Contact Us | Privacy Policy | Legal

Downloading picture http://now.eloqua.com/visitor/v200/svrGP.asp?pps=3&siteid=287&ref2=elqNone&tzo=300&ms=312...



This should be the next page you see.

Click here to download MySQL 5.0

MySQL AB :: MySQL Downloads - Microsoft Internet Explorer

Address http://dev.mysql.com/downloads/

MySQL Downloads

MySQL software is published under an open source license and is available in two ways:

1. **MySQL Community Edition** is the freely downloadable version of the world's most popular open source database. It is supported by a huge and active community of open source developers and enthusiasts. MySQL Community Edition uses the GPL License, is released early and often, and includes all features, including the latest features under development.
2. **MySQL Network** is available for users who want access to our world-class support services, Knowledge Base and certified software. This subscription service is designed to save developers and DBAs time and effort.
   - Enterprise-grade support to help solve the toughest issues
   - Online automated software advisors to keep your servers up to date
   - Certified MySQL Software that saves time and eliminates guesswork
   - Comprehensive KnowledgeBase of best practices technical articles

   Get support with MySQL Network today »

MySQL Community Edition -- Database Server and Client

Current Release (Recommended):
MySQL 5.0 -- Generally Available (GA) release for production use

Upcoming Releases:
MySQL 5.1 -- Beta release New!, Test new features early!
Snapshots -- source code snapshots of the development trees

Older Releases:
MySQL 4.1 -- Previous GA release
Archives of Older Releases

MySQL Cluster

MySQL Cluster is included in version 5.0 of the MySQL database server, as part of the MySQL Max packages. Binaries and source are available from the MySQL 5.0 download page.

Get Support with MySQL Network today!

Order from our online shop and get technical support from MySQL engineers today.

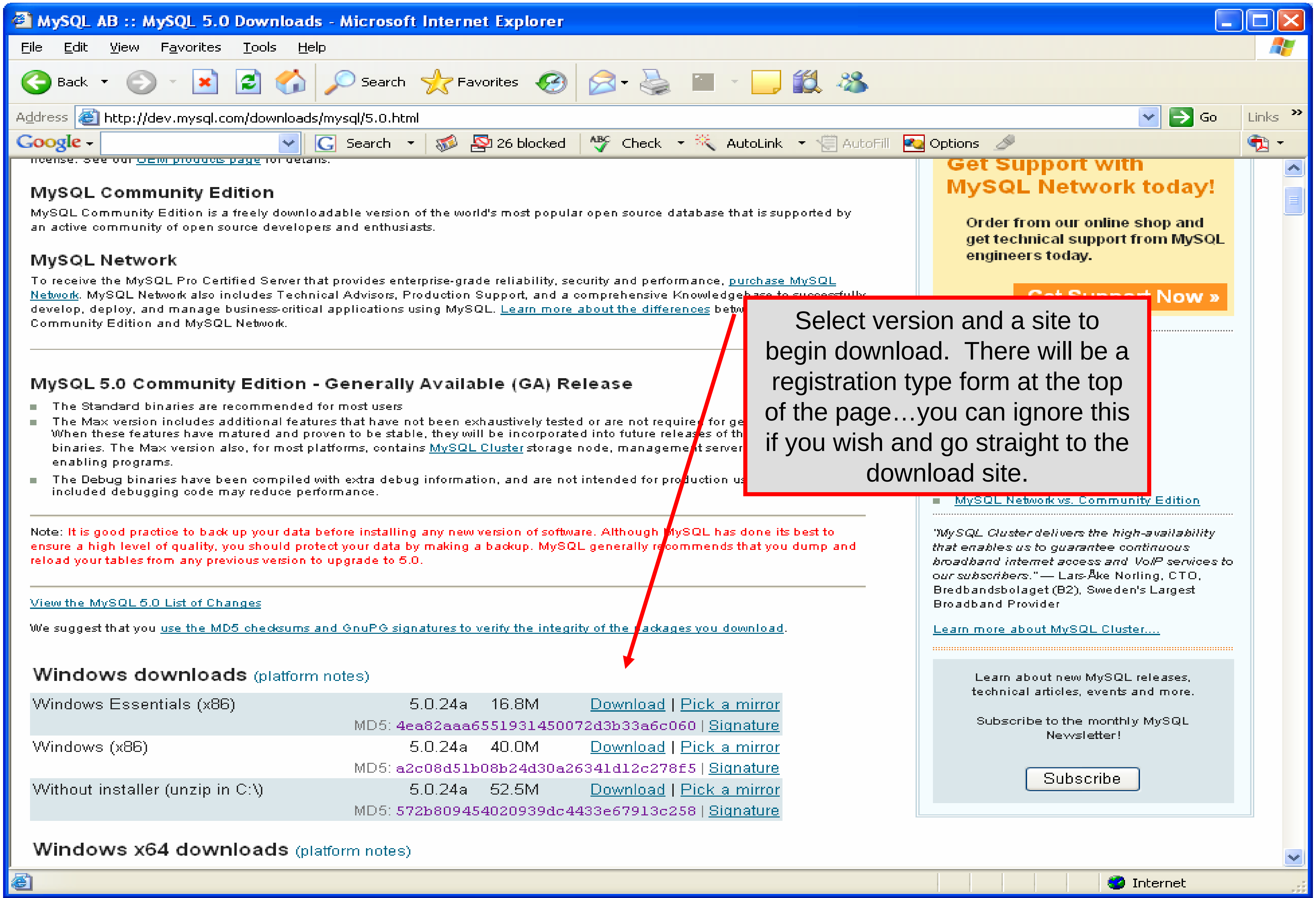
Get Support Now »

Related pages:
- MySQL Network vs. Community Edition
- Training
- Documentation

Learn about new MySQL releases, technical articles, events and more.

Subscribe to the monthly MySQL Newsletter!

File Edit View Favorites Tools Help
Back Search Favorites
Google Search 26 blocked Check Look for Map AutoFill Options
DEVELOPER ZONE
Logout / Profile
Search
MySQL.com MySQL Network Developer Zone Partners Online Shop
Downloads Documentation Forums Lists Bugs Events User Groups Guilds Blogs Support Resources FAQ Forge
Overview Database Server Cluster MaxDB GUI Tools Connectors Mirrors
Subscribe
Internet


MySQL AB :: MySQL 5.0 Downloads - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://dev.mysql.com/downloads/mysql/5.0.html
Go Links
Google Search 26 blocked Check AutoLink AutoFill Options
Internet

license. See our OEM products page for details.

MySQL Community Edition

MySQL Community Edition is a freely downloadable version of the world's most popular open source database that is supported by an active community of open source developers and enthusiasts.

MySQL Network

To receive the MySQL Pro Certified Server that provides enterprise-grade reliability, security and performance, purchase MySQL Network. MySQL Network also includes Technical Advisors, Production Support, and a comprehensive Knowledgebase to successfully develop, deploy, and manage business-critical applications using MySQL. Learn more about the differences betw Community Edition and MySQL Network.

MySQL 5.0 Community Edition - Generally Available (GA) Release

- The Standard binaries are recommended for most users
- The Max version includes additional features that have not been exhaustively tested or are not required for ge When these features have matured and proven to be stable, they will be incorporated into future releases of th binaries. The Max version also, for most platforms, contains MySQL Cluster storage node, management server enabling programs.
- The Debug binaries have been compiled with extra debug information, and are not intended for production us included debugging code may reduce performance.

Note: It is good practice to back up your data before installing any new version of software. Although MySQL has done its best to ensure a high level of quality, you should protect your data by making a backup. MySQL generally recommends that you dump and reload your tables from any previous version to upgrade to 5.0.

View the MySQL 5.0 List of Changes

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

Windows downloads (platform notes)

Windows Essentials (x86)	5.0.24a	16.8M	Download | Pick a mirror
			MD5: 4ea82aaa6551931450072d3b33a6c060 | Signature
Windows (x86)	5.0.24a	40.0M	Download | Pick a mirror
			MD5: a2c08d51b08b24d30a26341d12c278f5 | Signature
Without installer (unzip in C:\)	5.0.24a	52.5M	Download | Pick a mirror
			MD5: 572b809454020939dc4433e67913c258 | Signature

Windows x64 downloads (platform notes)

Get Support with MySQL Network today!

Order from our online shop and get technical support from MySQL engineers today.

Get Support Now »

MySQL Network vs. Community Edition

"MySQL Cluster delivers the high-availability that enables us to guarantee continuous broadband internet access and VoIP services to our subscribers." — Lars-Åke Norling, CTO, Bredbandsbolaget (B2), Sweden's Largest Broadband Provider

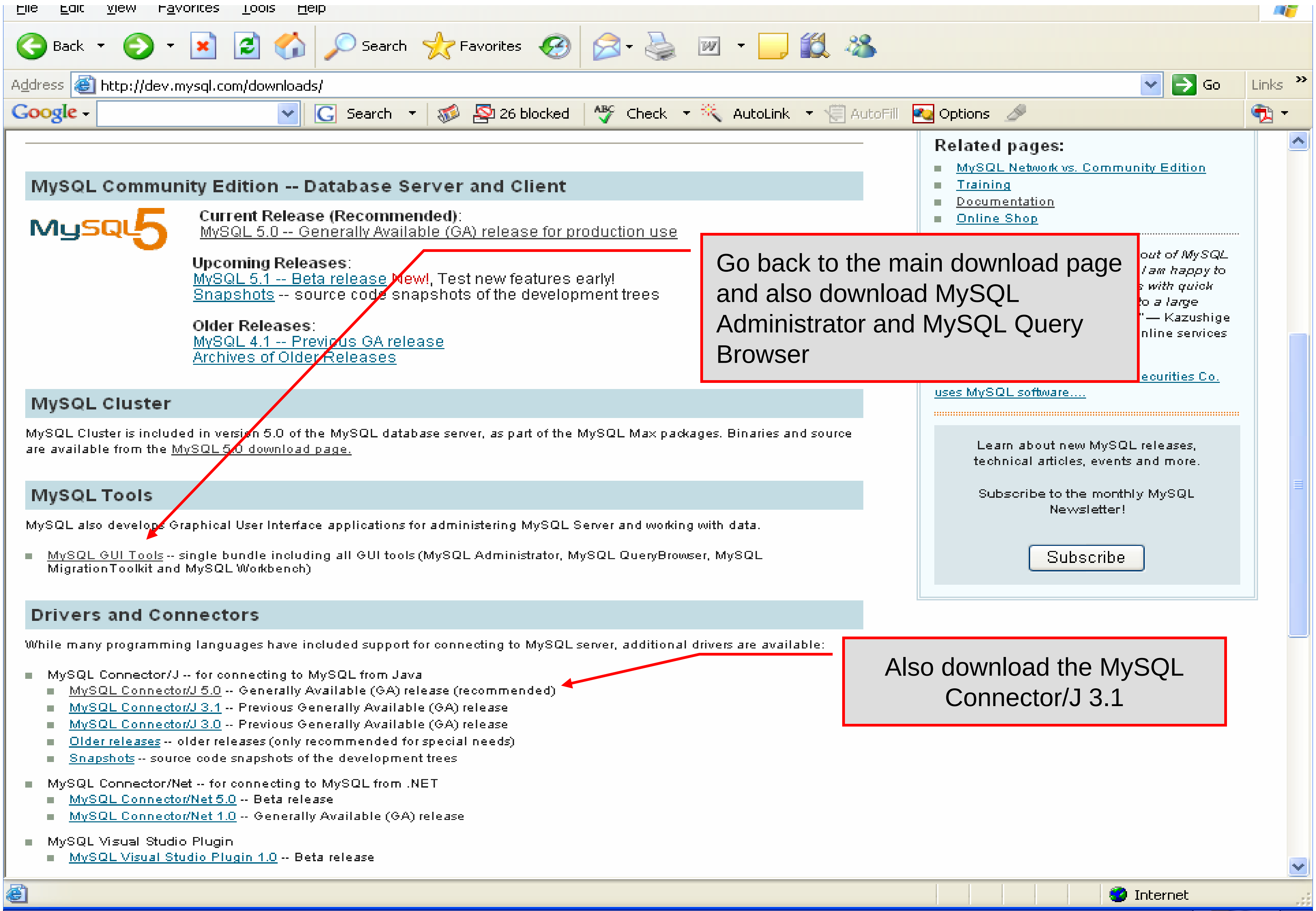
Learn more about MySQL Cluster....

Learn about new MySQL releases, technical articles, events and more.

Subscribe to the monthly MySQL Newsletter!

Subscribe

Select version and a site to begin download. There will be a registration type form at the top of the page…you can ignore this if you wish and go straight to the download site.



MySQL Community Edition -- Database Server and Client

Current Release (Recommended):
MySQL 5.0 -- Generally Available (GA) release for production use

Upcoming Releases:
MySQL 5.1 -- Beta release New!, Test new features early!
Snapshots -- source code snapshots of the development trees

Older Releases:
MySQL 4.1 -- Previous GA release
Archives of Older Releases

MySQL Cluster

MySQL Cluster is included in version 5.0 of the MySQL database server, as part of the MySQL Max packages. Binaries and source are available from the MySQL 5.0 download page.

MySQL Tools

MySQL also develops Graphical User Interface applications for administering MySQL Server and working with data.

- MySQL GUI Tools -- single bundle including all GUI tools (MySQL Administrator, MySQL QueryBrowser, MySQL MigrationToolkit and MySQL Workbench)

Drivers and Connectors

While many programming languages have included support for connecting to MySQL server, additional drivers are available:

- MySQL Connector/J -- for connecting to MySQL from Java
  - MySQL Connector/J 5.0 -- Generally Available (GA) release (recommended)
  - MySQL Connector/J 3.1 -- Previous Generally Available (GA) release
  - MySQL Connector/J 3.0 -- Previous Generally Available (GA) release
  - Older releases -- older releases (only recommended for special needs)
  - Snapshots -- source code snapshots of the development trees
- MySQL Connector/Net -- for connecting to MySQL from .NET
  - MySQL Connector/Net 5.0 -- Beta release
  - MySQL Connector/Net 1.0 -- Generally Available (GA) release
- MySQL Visual Studio Plugin
  - MySQL Visual Studio Plugin 1.0 -- Beta release

Related pages:
- MySQL Network vs. Community Edition
- Training
- Documentation
- Online Shop

out of MySQL
I am happy to
with quick
to a large
" — Kazushige
nline services

ecurities Co.
uses MySQL software....

Learn about new MySQL releases, technical articles, events and more.

Subscribe to the monthly MySQL Newsletter!

Subscribe

Go back to the main download page and also download MySQL Administrator and MySQL Query Browser

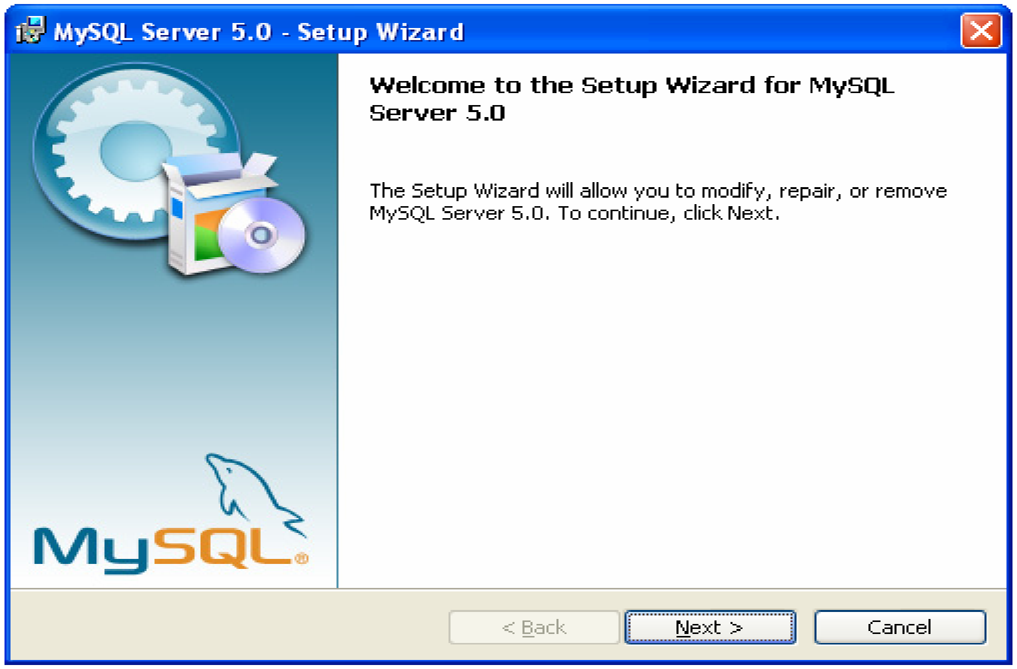
Also download the MySQL Connector/J 3.1

Installing MySQL 5.0

- Once you've got MySQL downloaded, go through the installation process. It may vary somewhat depending on platform.
- I've illustrated the basic install on Windows XP over the next few pages, just to give you an idea of what you should be seeing.

Installing MySQL 5.0

- Once you've got MySQL downloaded, go through the installation process. It may vary somewhat depending on platform.
- I've illustrated the basic install on Windows XP over the next few pages, just to give you an idea of what you should be seeing.
- Once the Window installer is running you should see the following window appear:

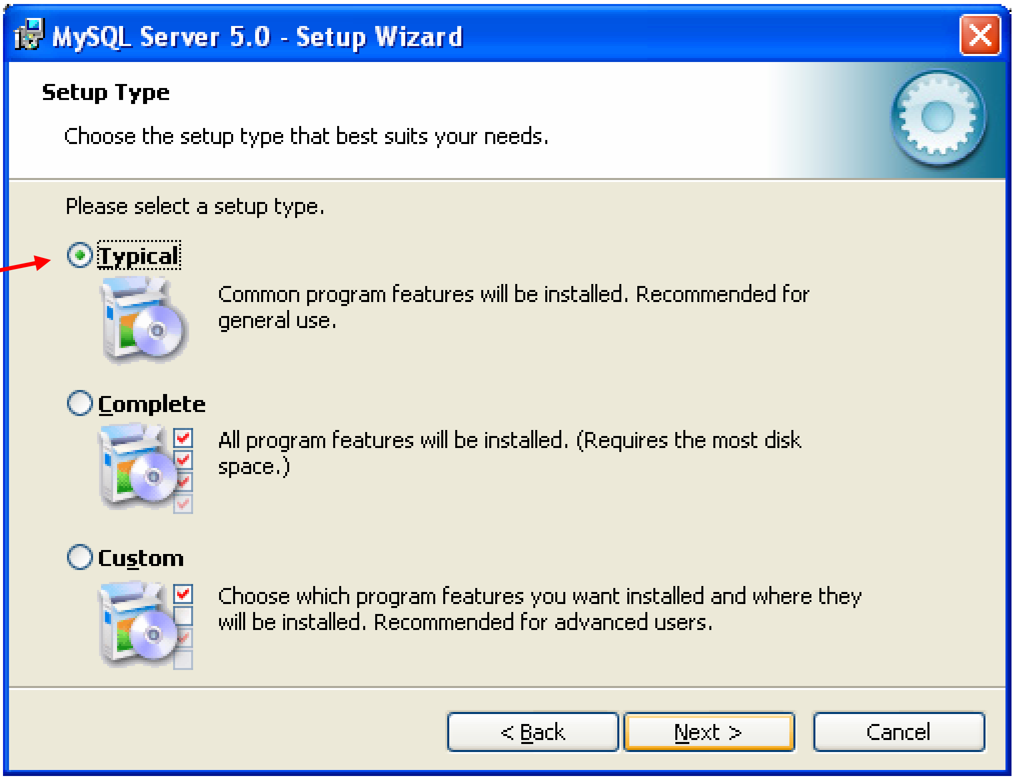

MySQL Server 5.0 - Setup Wizard

Welcome to the Setup Wizard for MySQL Server 5.0

The Setup Wizard will allow you to modify, repair, or remove MySQL Server 5.0. To continue, click Next.

Installing MySQL 5.0 (cont.)

Your choice here. For this course, a typical set-up will be fine.



MySQL Server 5.0 - Setup Wizard

Setup Type

Choose the setup type that best suits your needs.

Please select a setup type.

[✓ Typical]

Common program features will be installed. Recommended for general use.

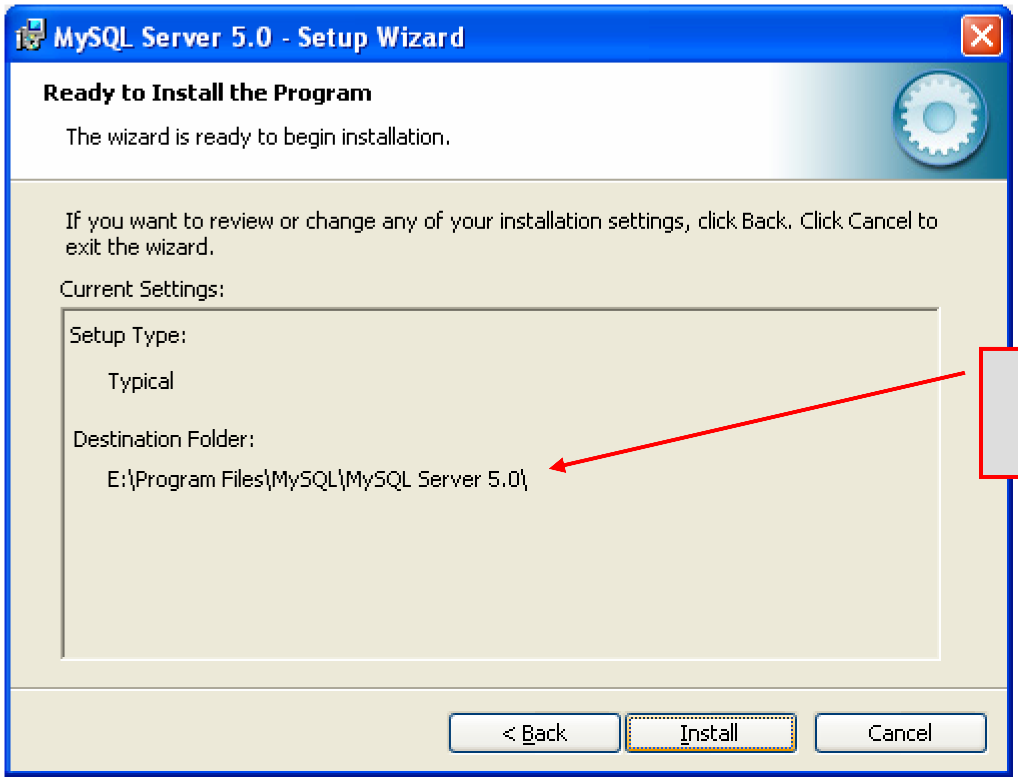
Complete

All program features will be installed. (Requires the most disk space.)

Custom

Choose which program features you want installed and where they will be installed. Recommended for advanced users.

Installing MySQL 5.0 (cont.)



Ready to Install the Program

The wizard is ready to begin installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Current Settings:

Setup Type:

Typical

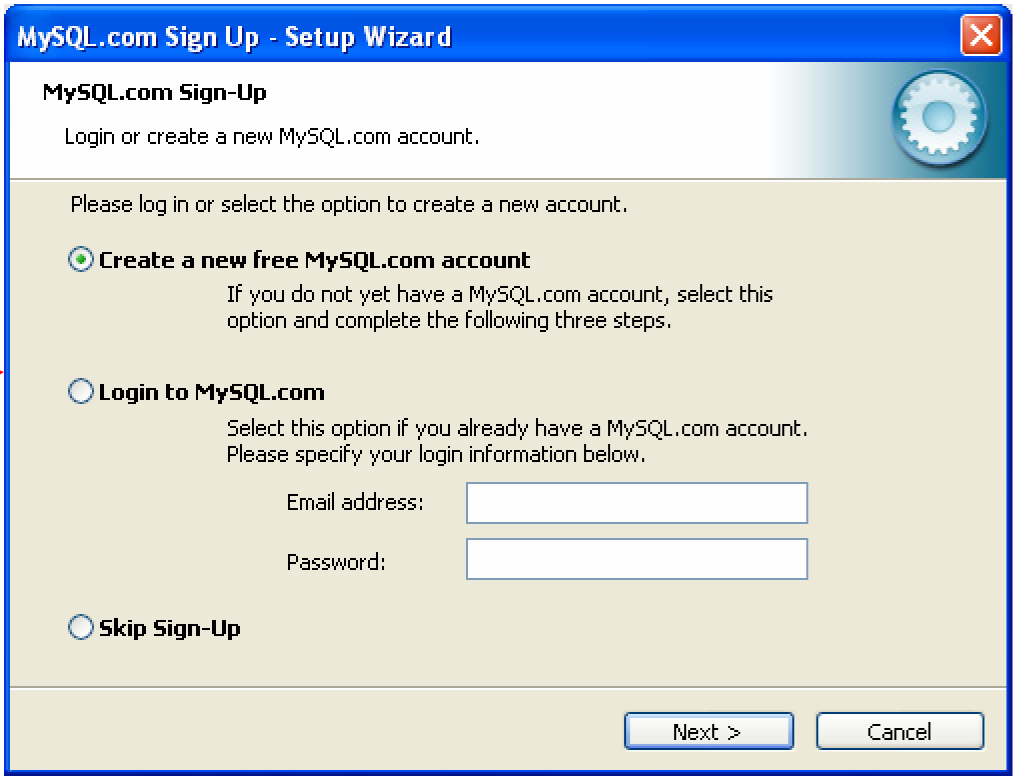
Destination Folder:

E:\Program Files\MySQL\MySQL Server 5.0\

Select the destination folder for the install.

Installing MySQL 5.0 (cont.)

Again, your choice here. If you want to skip the sign-up that's fine.



MySQL.com Sign Up - Setup Wizard

MySQL.com Sign-Up

Login or create a new MySQL.com account.

Please log in or select the option to create a new account.

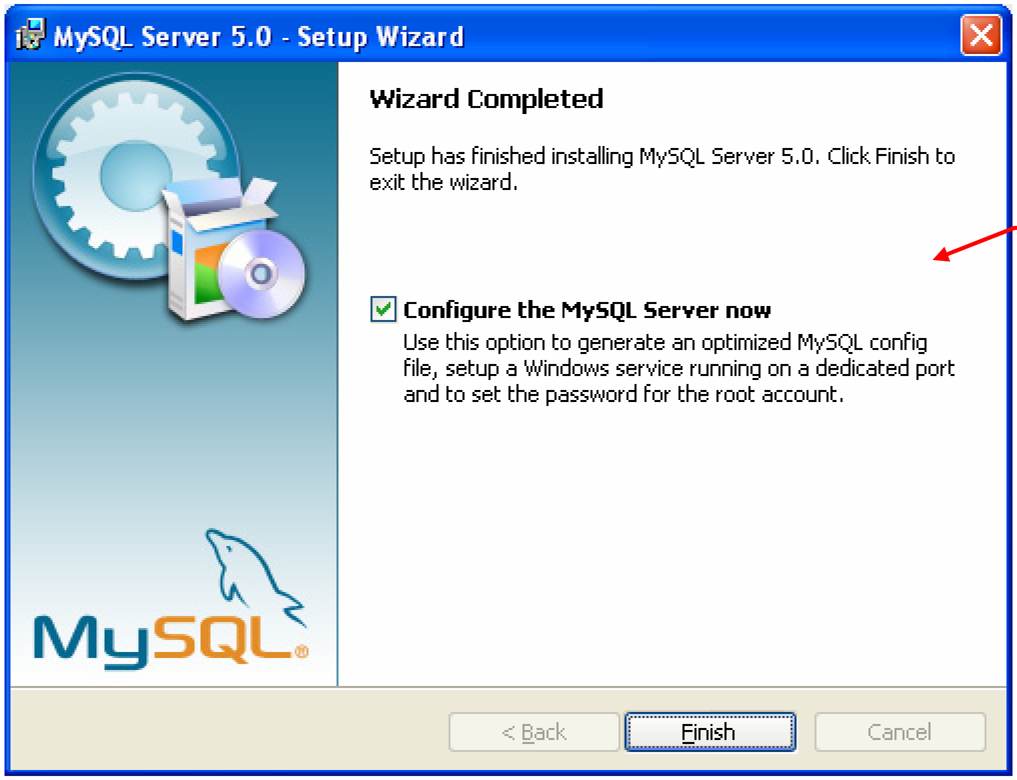
[✓ **Create a new free MySQL.com account**]
If you do not yet have a MySQL.com account, select this option and complete the following three steps.

Login to MySQL.com
Select this option if you already have a MySQL.com account. Please specify your login information below.

Email address:	
Password:	

Skip Sign-Up

Installing MySQL 5.0 (cont.)


MySQL Server 5.0 - Setup Wizard

Wizard Completed

Setup has finished installing MySQL Server 5.0. Click Finish to exit the wizard.

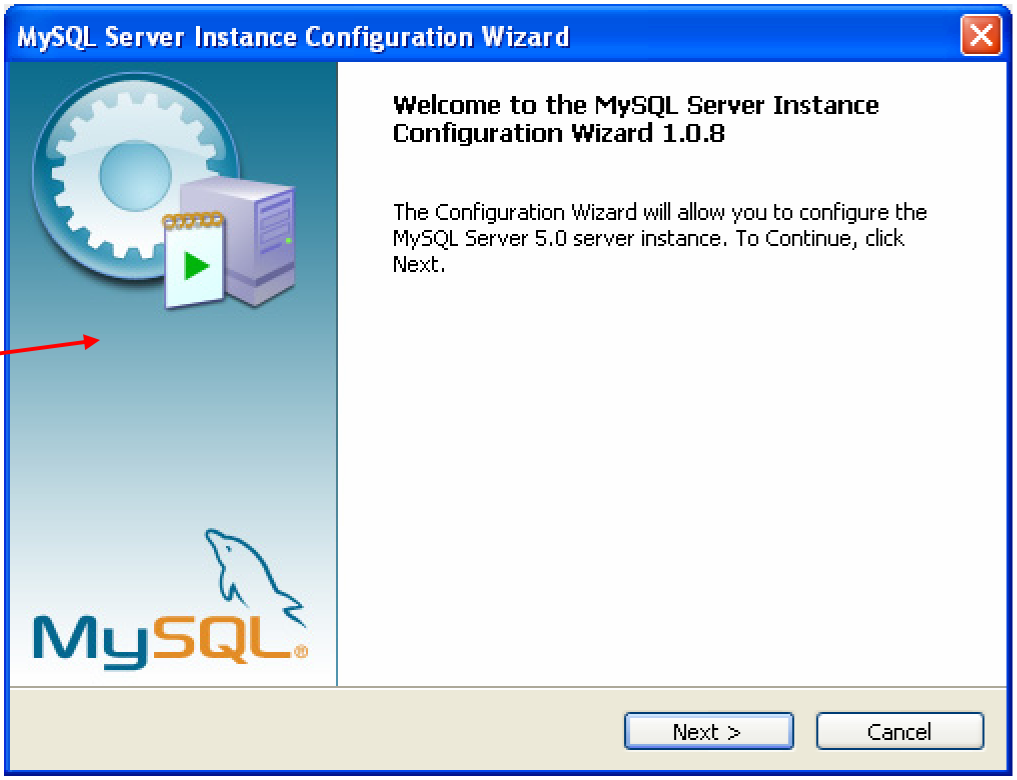
[✓] **Configure the MySQL Server now**

Use this option to generate an optimized MySQL config file, setup a Windows service running on a dedicated port and to set the password for the root account.

If everything has good well up to this point, you should see a window similar to this one. Click the Finish button, cross your fingers, and hang-on while the installer configures your system and gets MySQL up and running as a service.

Installing MySQL 5.0 (cont.)

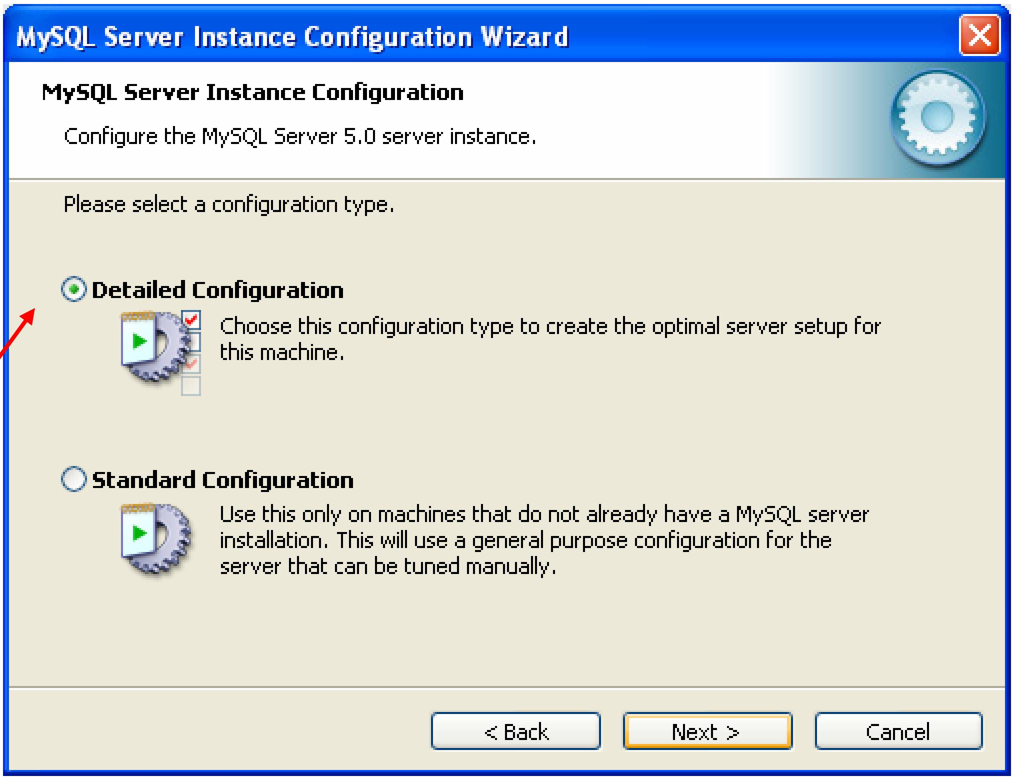
Initial server configuration window


MySQL Server Instance Configuration Wizard

Welcome to the MySQL Server Instance Configuration Wizard 1.0.8

The Configuration Wizard will allow you to configure the MySQL Server 5.0 server instance. To Continue, click Next.

Installing MySQL 5.0 (cont.)



MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please select a configuration type.

[✓ **Detailed Configuration**]

Choose this configuration type to create the optimal server setup for this machine.

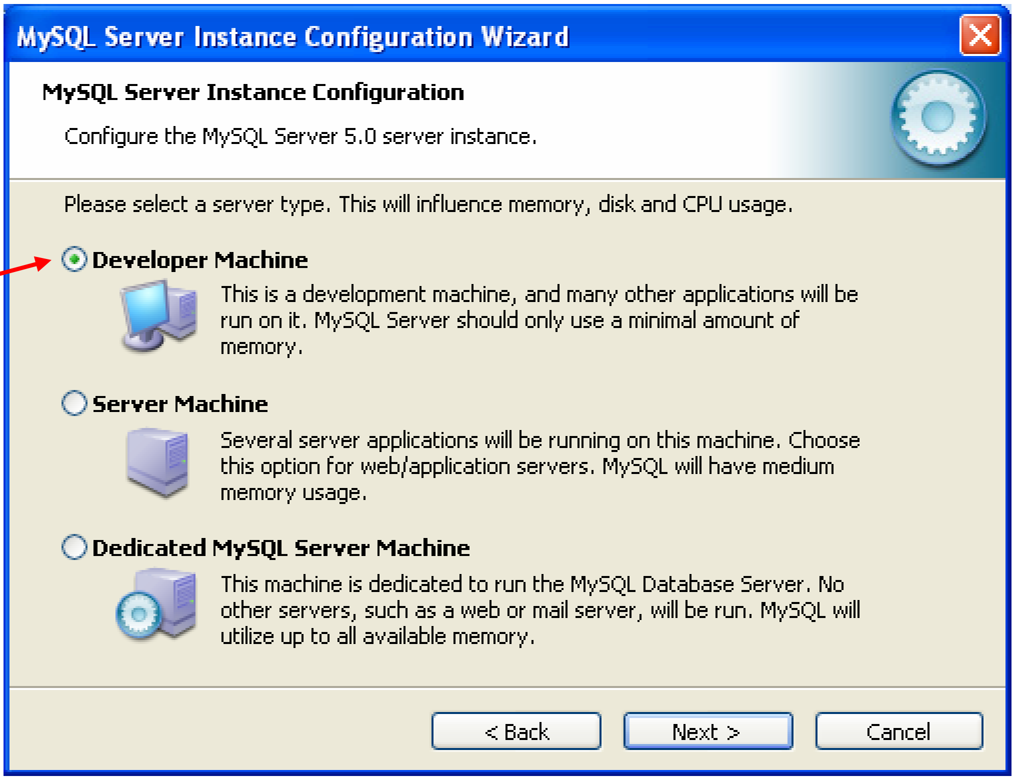
Standard Configuration

Use this only on machines that do not already have a MySQL server installation. This will use a general purpose configuration for the server that can be tuned manually.

Your choice here. If you are not sure if there is already a MySQL server on your machine, choose the detailed configuration setting.

Installing MySQL 5.0 (cont.)

Choose the developer machine option



MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please select a server type. This will influence memory, disk and CPU usage.

[✓ **Developer Machine**]

This is a development machine, and many other applications will be run on it. MySQL Server should only use a minimal amount of memory.

Server Machine

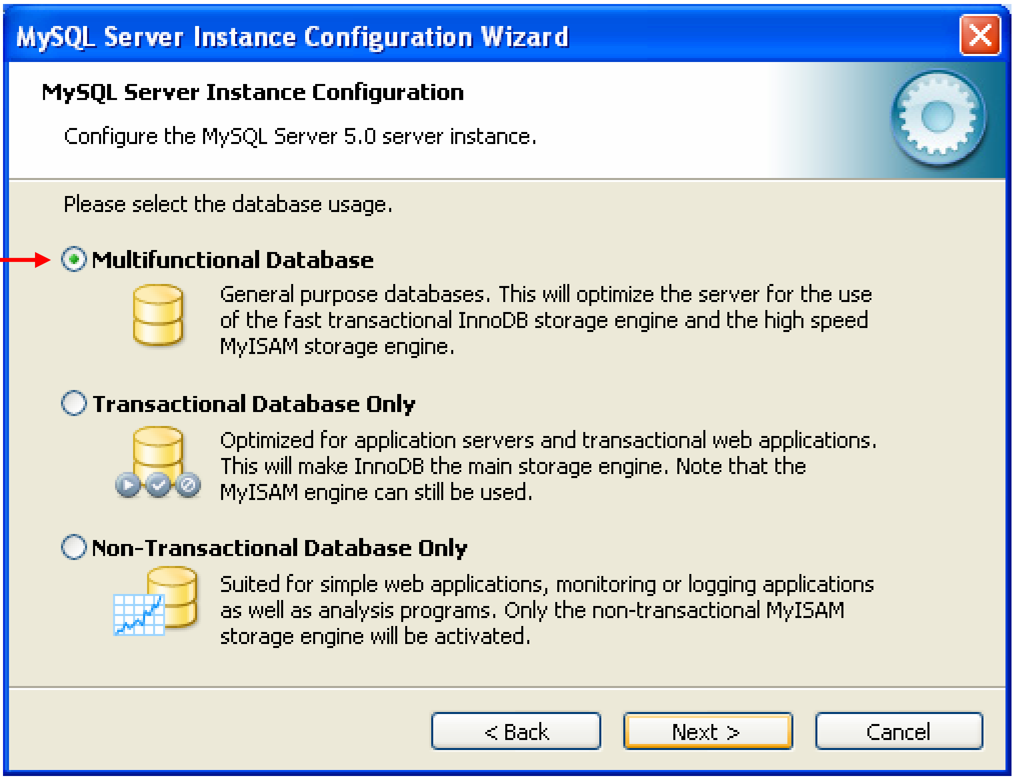
Several server applications will be running on this machine. Choose this option for web/application servers. MySQL will have medium memory usage.

Dedicated MySQL Server Machine

This machine is dedicated to run the MySQL Database Server. No other servers, such as a web or mail server, will be run. MySQL will utilize up to all available memory.

Installing MySQL 5.0 (cont.)

Choose the multifunctional database option



MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please select the database usage.

[✓ **Multifunctional Database**]

General purpose databases. This will optimize the server for the use of the fast transactional InnoDB storage engine and the high speed MyISAM storage engine.

Transactional Database Only

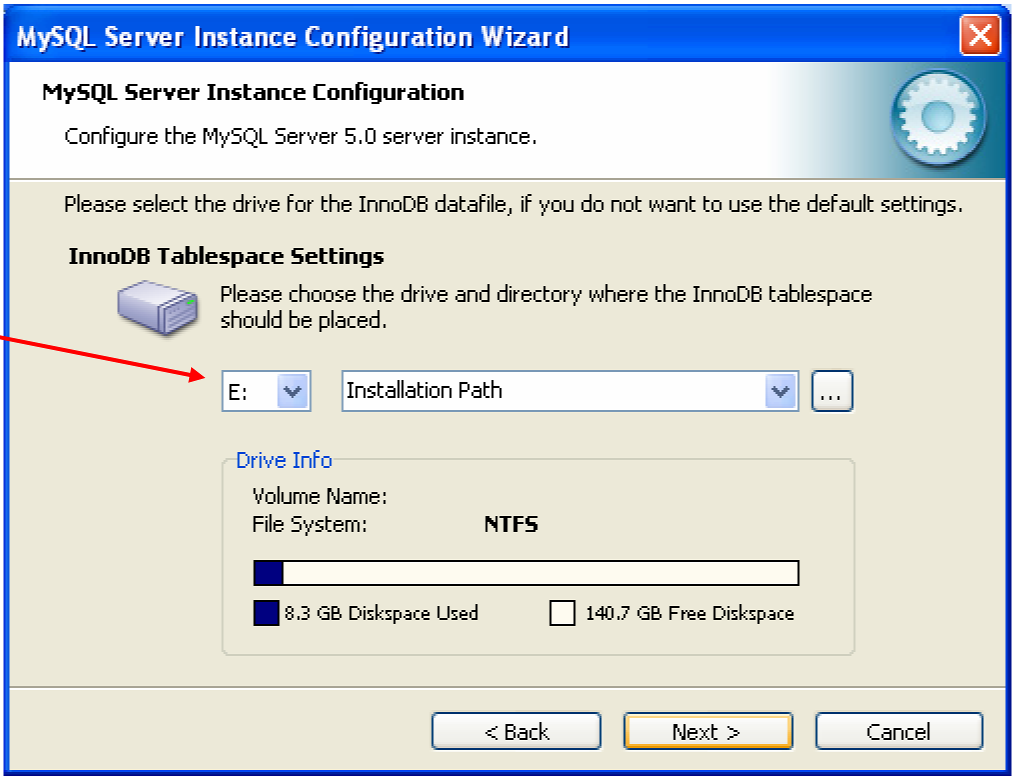
Optimized for application servers and transactional web applications. This will make InnoDB the main storage engine. Note that the MyISAM engine can still be used.

Non-Transactional Database Only

Suited for simple web applications, monitoring or logging applications as well as analysis programs. Only the non-transactional MyISAM storage engine will be activated.

Installing MySQL 5.0 (cont.)

Choose the installation path to keep InnoDB tables in same area as other MySQL files



MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please select the drive for the InnoDB datafile, if you do not want to use the default settings.

InnoDB Tablespace Settings

Please choose the drive and directory where the InnoDB tablespace should be placed.

E: | Installation Path

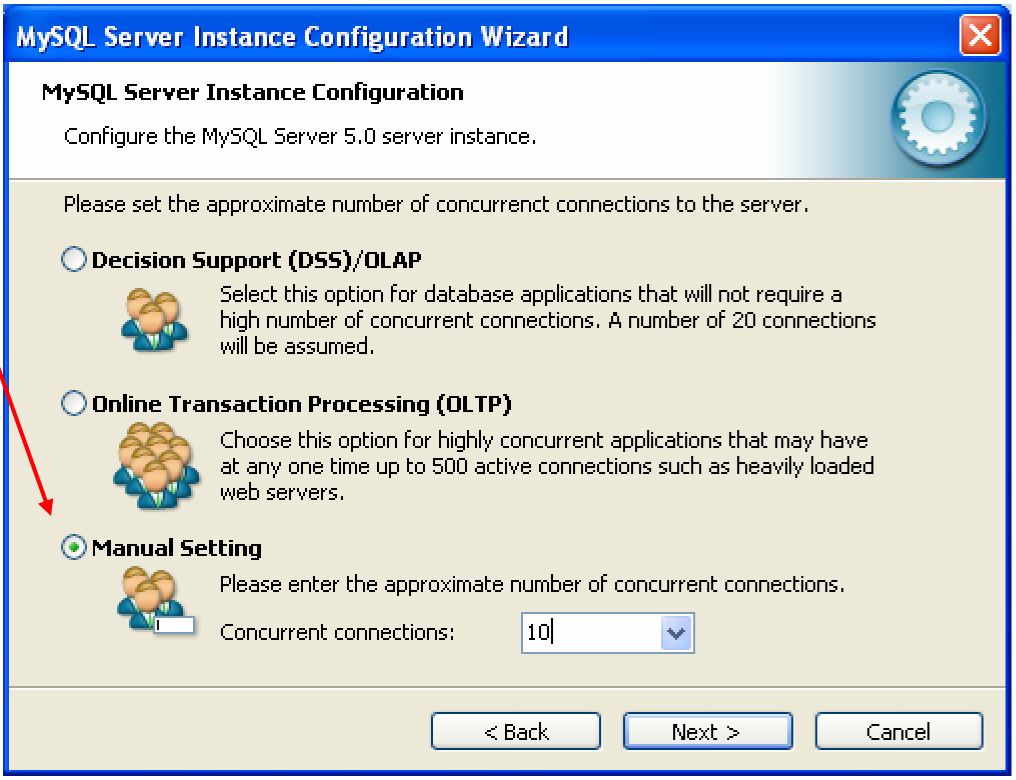
Drive Info

Volume Name:
File System: **NTFS**

■ 8.3 GB Diskspace Used   □ 140.7 GB Free Diskspace

Installing MySQL 5.0 (cont.)

Select manual setting for this option. The default is 15, I set mine to 10, but you can use any number you would like, but pick something greater than 2 or 3.



MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please set the approximate number of concurrenct connections to the server.

( ) **Decision Support (DSS)/OLAP**

Select this option for database applications that will not require a high number of concurrent connections. A number of 20 connections will be assumed.

( ) **Online Transaction Processing (OLTP)**

Choose this option for highly concurrent applications that may have at any one time up to 500 active connections such as heavily loaded web servers.

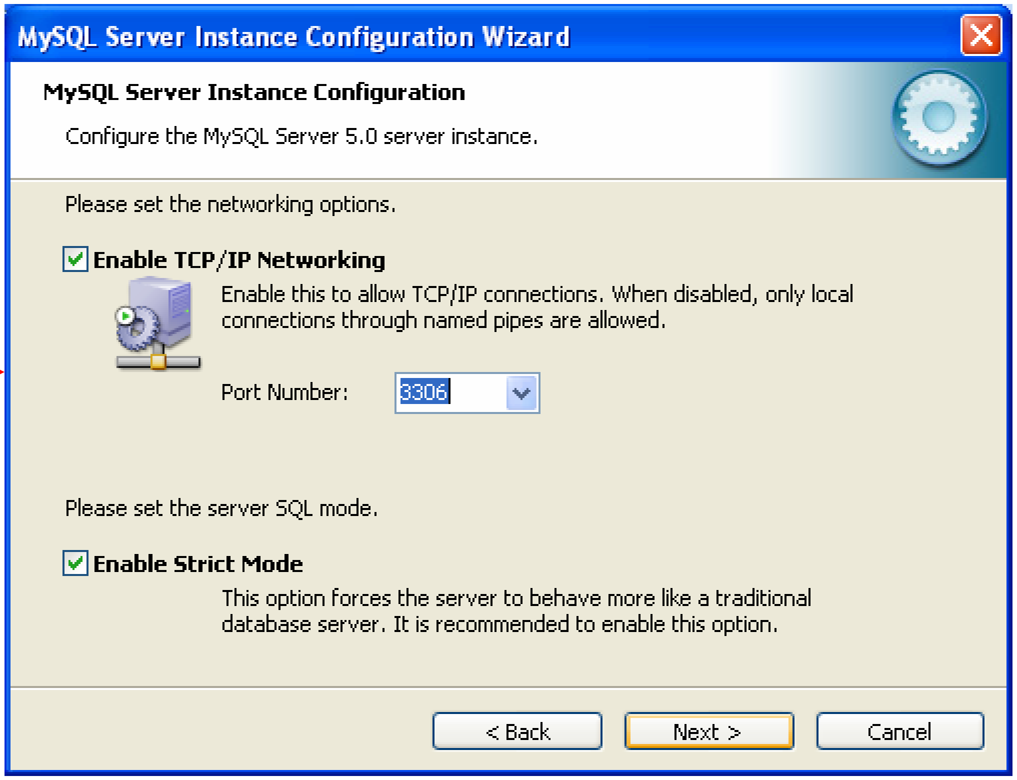
[✓ **Manual Setting**]

Please enter the approximate number of concurrent connections.

Concurrent connections: 10

Installing MySQL 5.0 (cont.)

Accept all defaults in this window



MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please set the networking options.

[✓] **Enable TCP/IP Networking**

Enable this to allow TCP/IP connections. When disabled, only local connections through named pipes are allowed.

Port Number: 3306

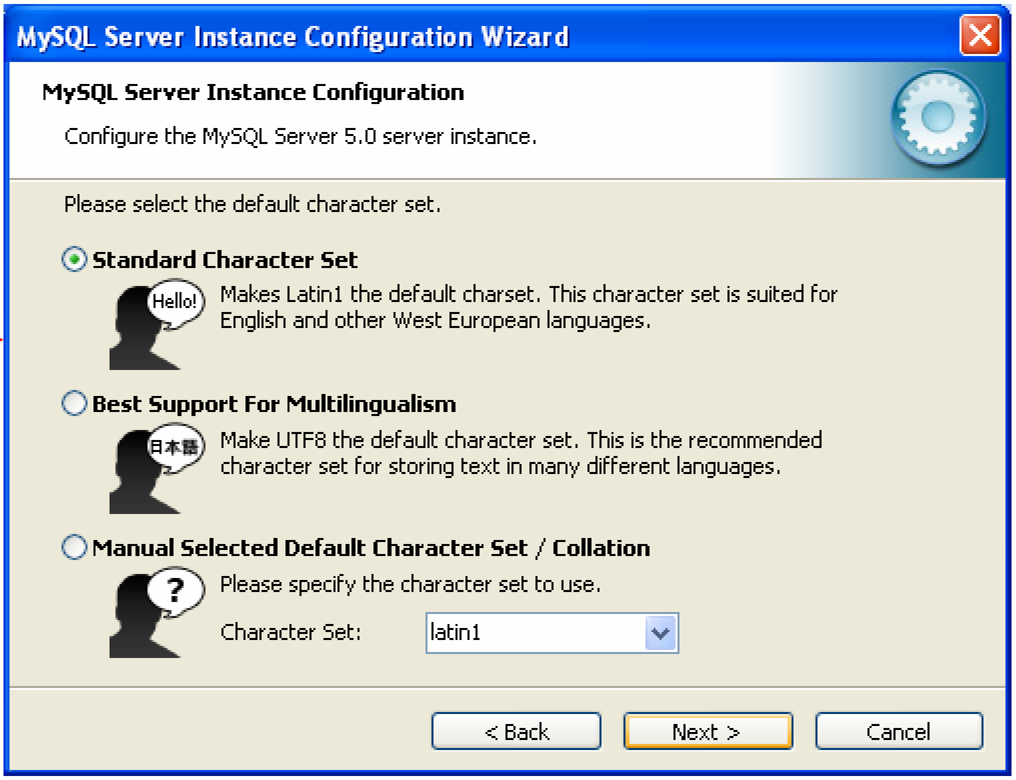
Please set the server SQL mode.

[✓] **Enable Strict Mode**

This option forces the server to behave more like a traditional database server. It is recommended to enable this option.

Installing MySQL 5.0 (cont.)

Your choice again



MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please select the default character set.

[✓ **Standard Character Set**]

Makes Latin1 the default charset. This character set is suited for English and other West European languages.

Best Support For Multilingualism

Make UTF8 the default character set. This is the recommended character set for storing text in many different languages.

Manual Selected Default Character Set / Collation

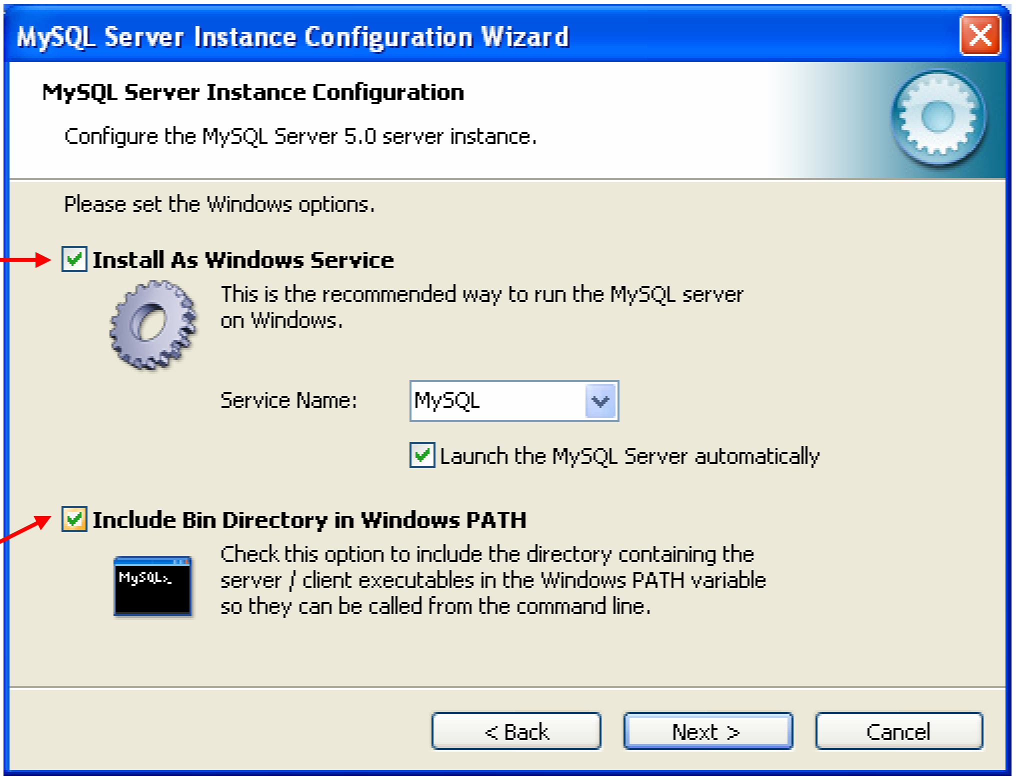
Please specify the character set to use.

Character Set: latin1

Installing MySQL 5.0 (cont.)

Accept default options

This option is not marked by default, but you can mark and accept it if you want to include MySQL file locations in your PATH statement.



MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please set the Windows options.

[✓] Install As Windows Service

This is the recommended way to run the MySQL server on Windows.

Service Name: MySQL

[✓] Launch the MySQL Server automatically

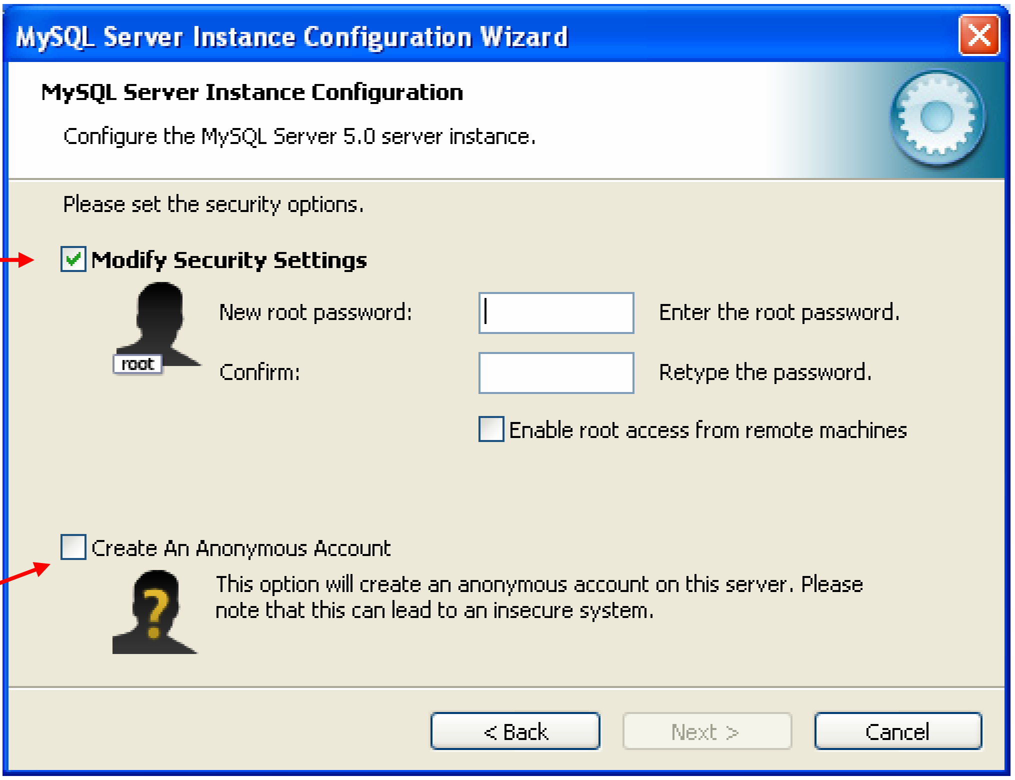
[✓] Include Bin Directory in Windows PATH

Check this option to include the directory containing the server / client executables in the Windows PATH variable so they can be called from the command line.

Installing MySQL 5.0 (cont.)

Accept default setting and enter a password for the root (superuser with all privileges by default). Enabling root access from remote machines is only necessary if you will be accessing the DB as the root user from a remote machine – we will not be doing this in this course.

Do not enable this option



MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Please set the security options.

[✓] **Modify Security Settings**

New root password:		Enter the root password.
Confirm:		Retype the password.

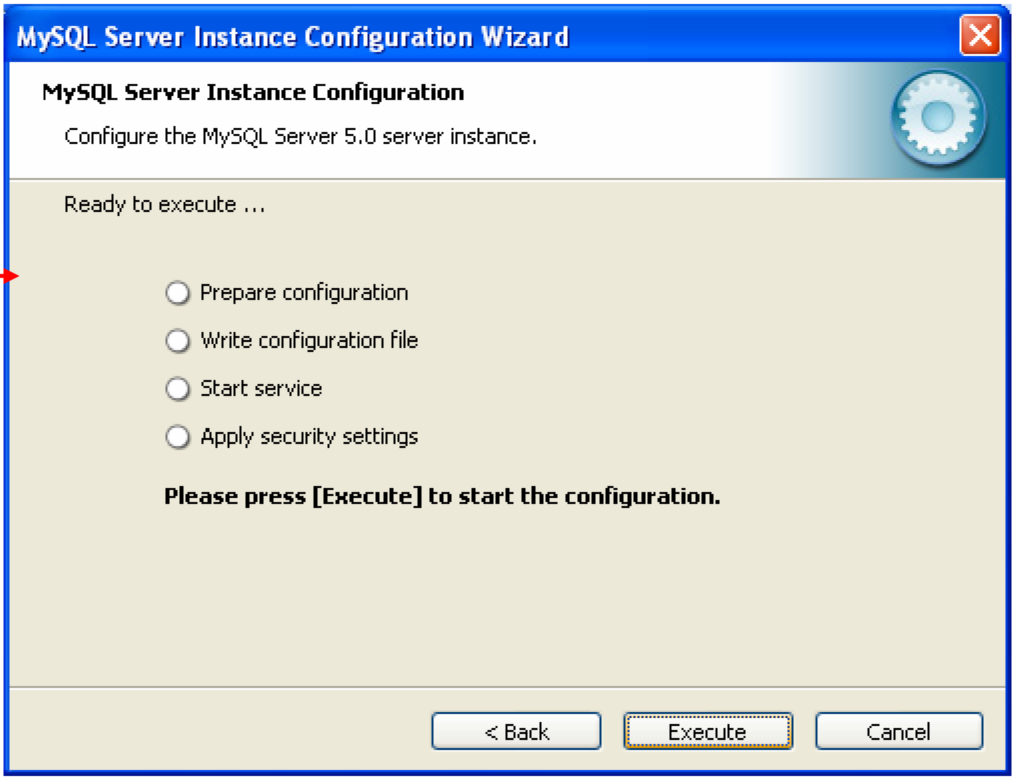
[ ] Enable root access from remote machines

[ ] Create An Anonymous Account

This option will create an anonymous account on this server. Please note that this can lead to an insecure system.

Installing MySQL 5.0 (cont.)

Configuration is about to begin. Now cross your fingers, toes, and anything else you have, take a deep breath, click the Execute button and close your eyes for a few seconds.


MySQL Server Instance Configuration Wizard

MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

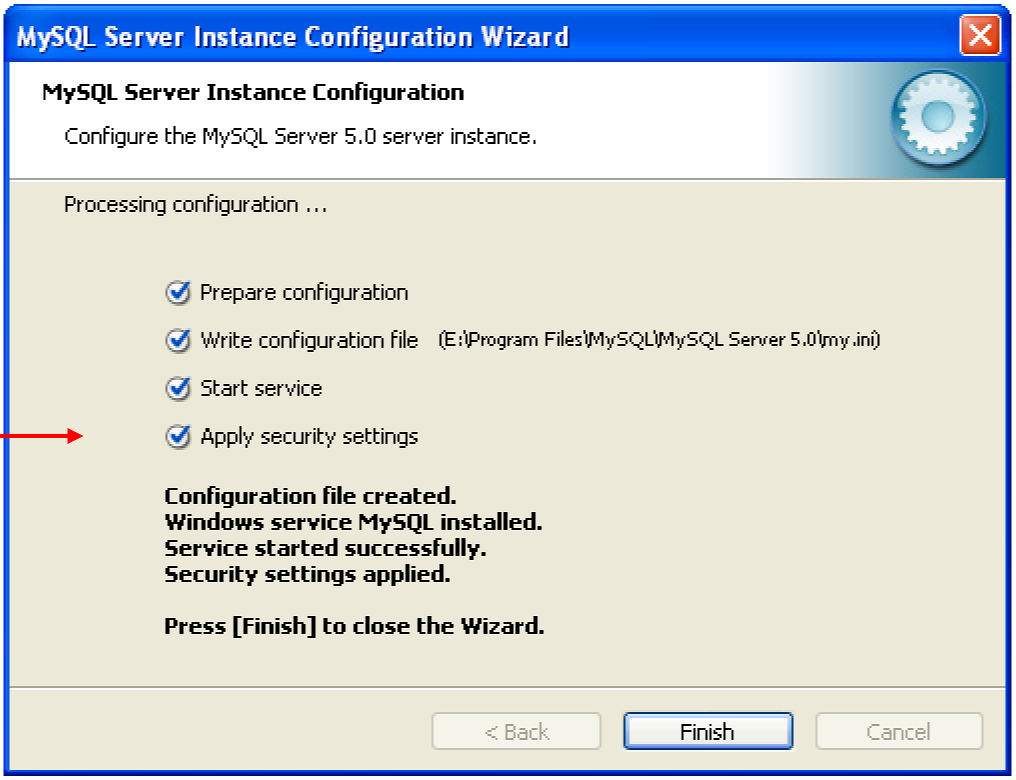
Ready to execute ...

- Prepare configuration
- Write configuration file
- Start service
- Apply security settings

Please press [Execute] to start the configuration.

Installing MySQL 5.0 (cont.)

You've successfully installed MySQL!!



MySQL Server Instance Configuration

Configure the MySQL Server 5.0 server instance.

Processing configuration ...

✓ Prepare configuration

✓ Write configuration file (E:\Program Files\MySQL\MySQL Server 5.0\my.ini)

✓ Start service

✓ Apply security settings

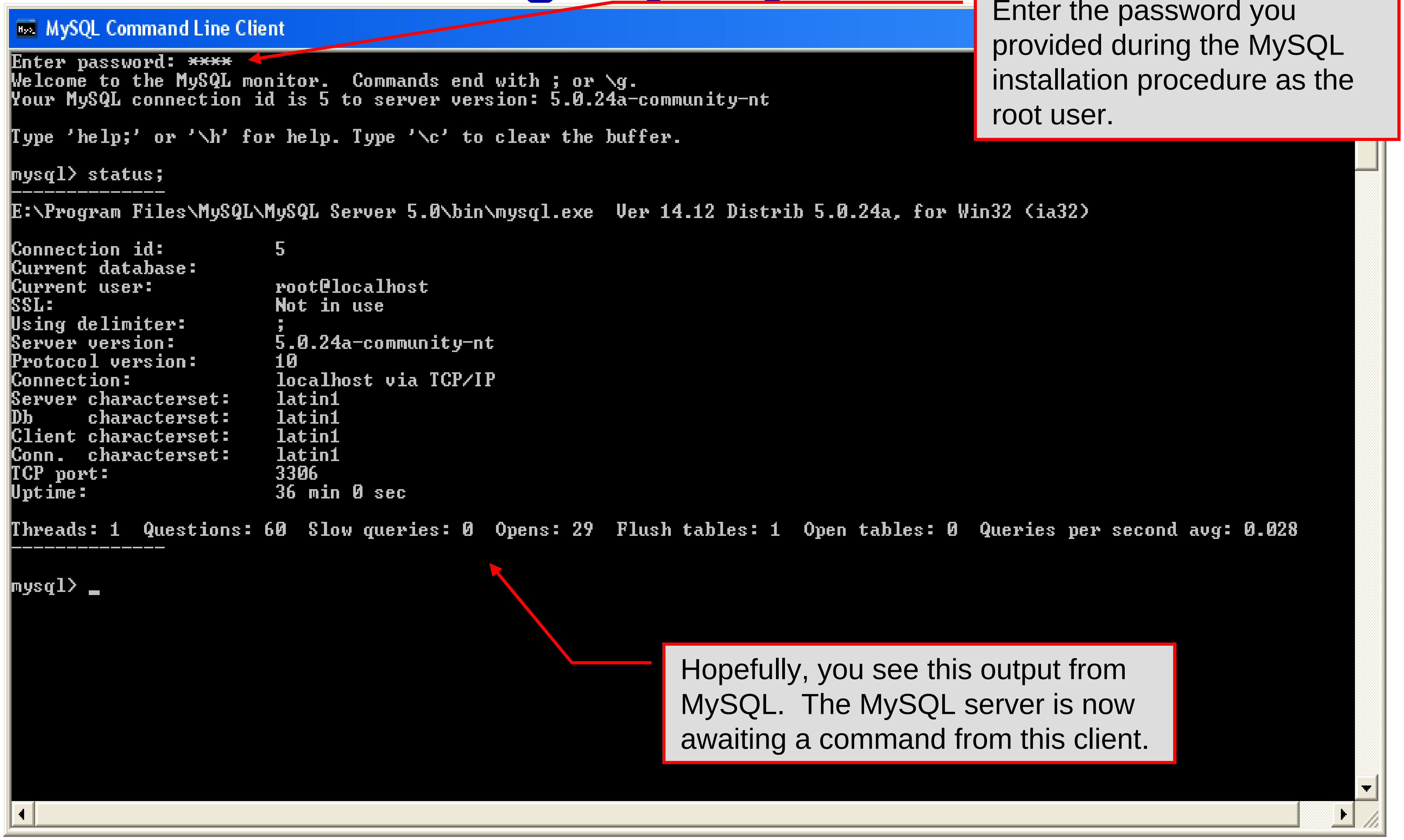
Configuration file created.
Windows service MySQL installed.
Service started successfully.
Security settings applied.

Press [Finish] to close the Wizard.

Running MySQL 5.0

- If you've successfully installed MySQL, it should now be running as a service on your machine. It will start automatically when your machine boots.
- Go into your listing of programs (from the start menu at the bottom: All Programs) and you should see MySQL appear. Since you will be running MySQL clients a lot, it will be easier if you pin the MySQL Client to the start menu.
- To verify that MySQL is running properly as a service you can either check the process window or run a MySQL client.

Running MySQL 5.0 (cont.)

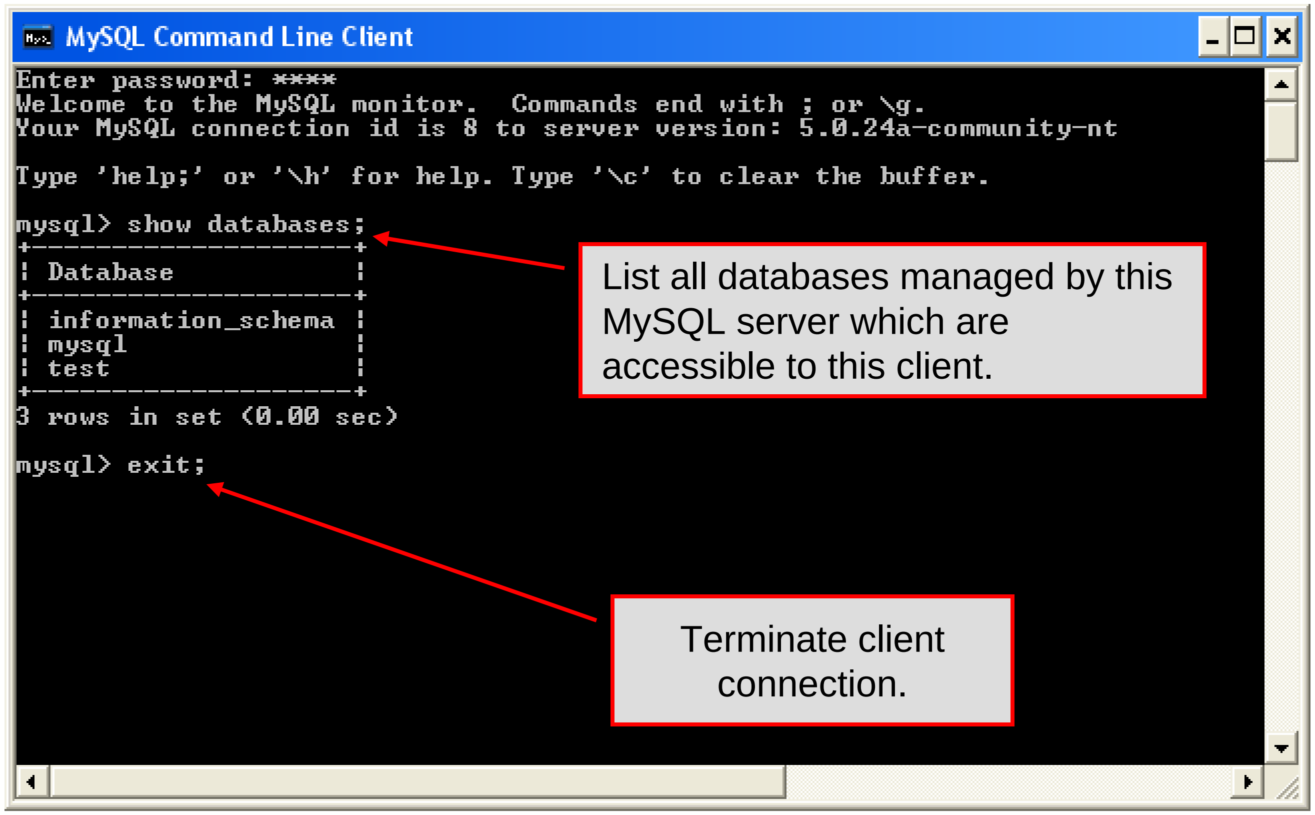


```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5 to server version: 5.0.24a-community-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> status;
--------------
E:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe  Ver 14.12 Distrib 5.0.24a, for Win32 (ia32)

Connection id:          5
Current database:
Current user:           root@localhost
SSL:                    Not in use
Using delimiter:        ;
Server version:         5.0.24a-community-nt
Protocol version:       10
Connection:             localhost via TCP/IP
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    latin1
Conn.  characterset:    latin1
TCP port:               3306
Uptime:                 36 min 0 sec

Threads: 1  Questions: 60  Slow queries: 0  Opens: 29  Flush tables: 1  Open tables: 0  Queries per second avg: 0.028
--------------

mysql> _
```

Enter the password you provided during the MySQL installation procedure as the root user.

Hopefully, you see this output from MySQL. The MySQL server is now awaiting a command from this client.

Running MySQL 5.0 (cont.)



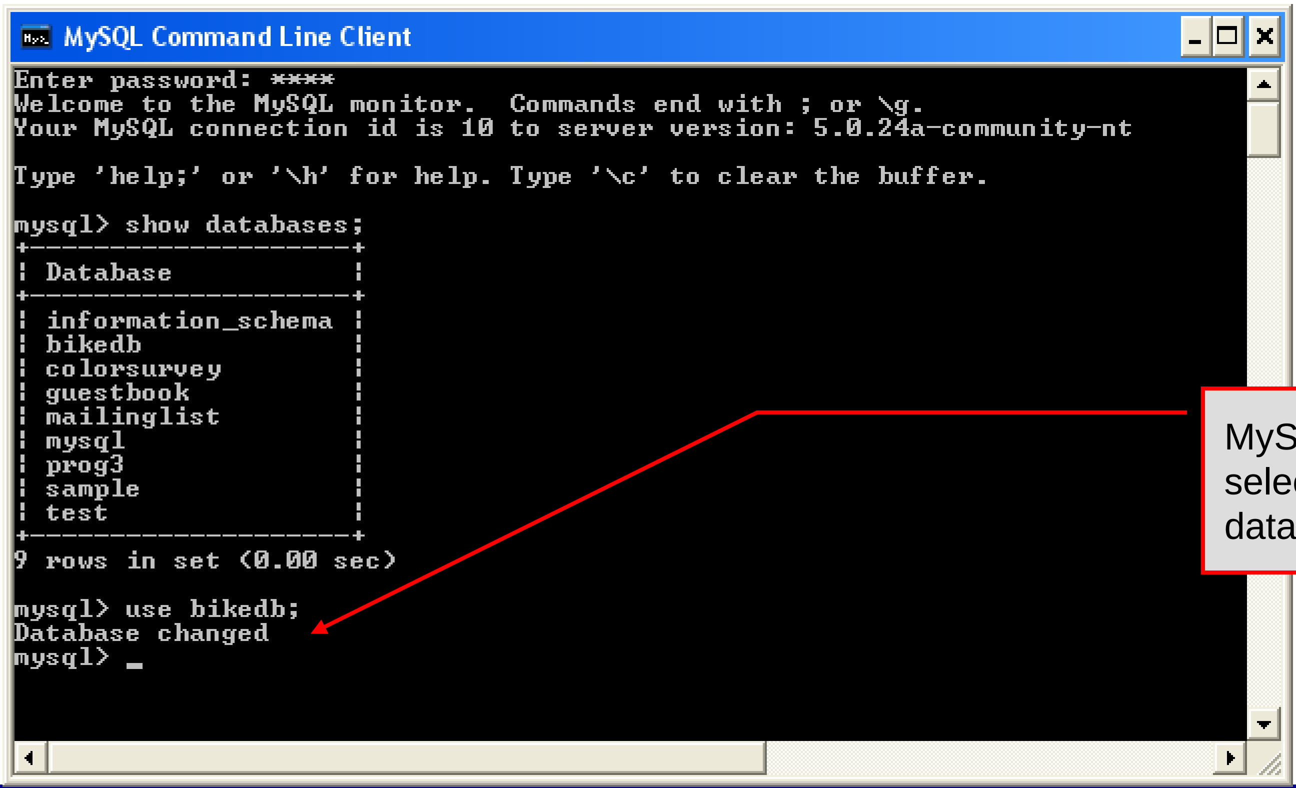
```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8 to server version: 5.0.24a-community-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.00 sec)

mysql> exit;
```

List all databases managed by this MySQL server which are accessible to this client.

Terminate client connection.

Specifying A Database Within MySQL

- Unless, it is specifically stated, in the following slides we'll assume that the user has root-level privileges.
- To select a database for use in MySQL the `use` command must be issued. In the example below, we'll select the `bikedb` database.

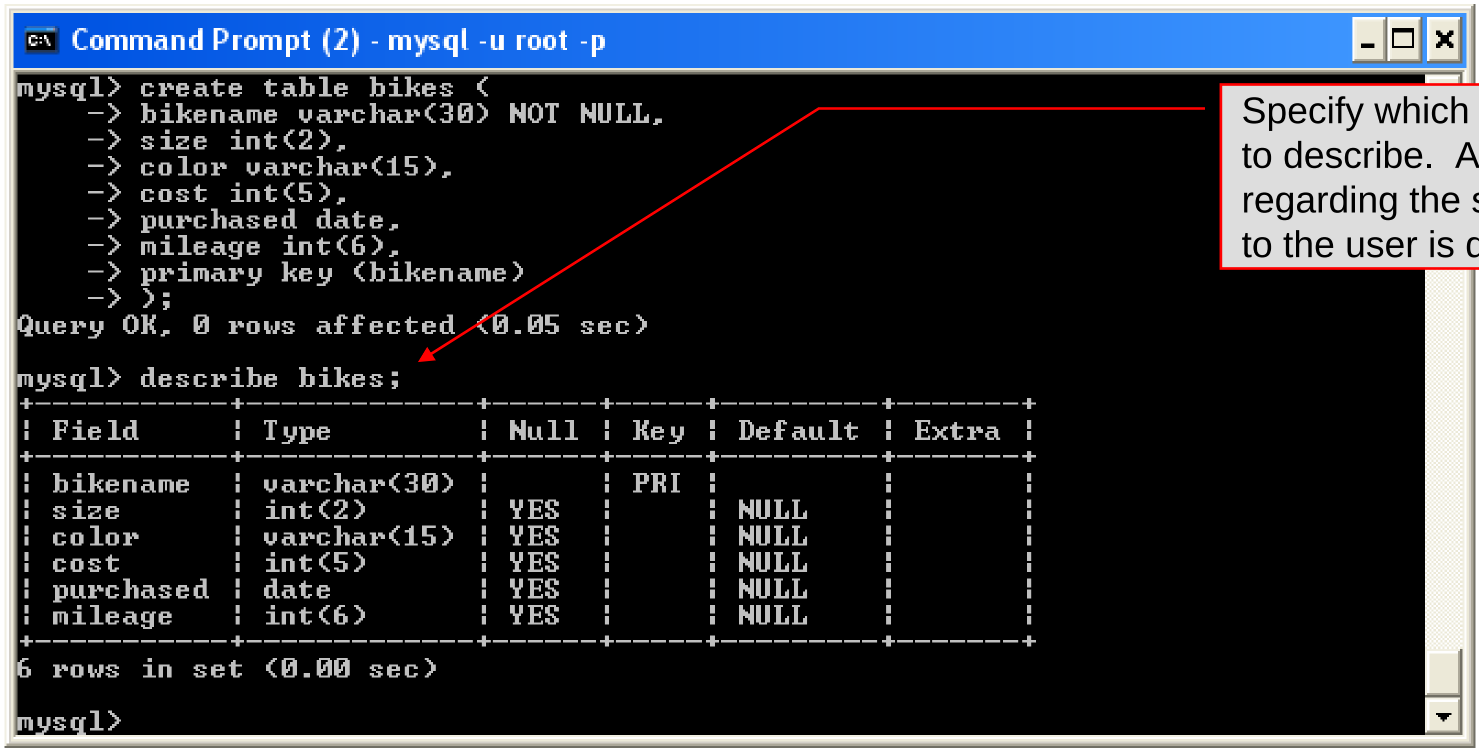


```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10 to server version: 5.0.24a-community-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| colorsurvey        |
| guestbook          |
| mailinglist        |
| mysql              |
| prog3              |
| sample             |
| test               |
+--------------------+
9 rows in set (0.00 sec)

mysql> use bikedb;
Database changed
mysql> _
```

MySQL acknowledges selection of bikedb database.

Viewing the Schema of a Relation

- To see the schema of a relation within a database, use the `describe <tablename>` command as illustrated below.

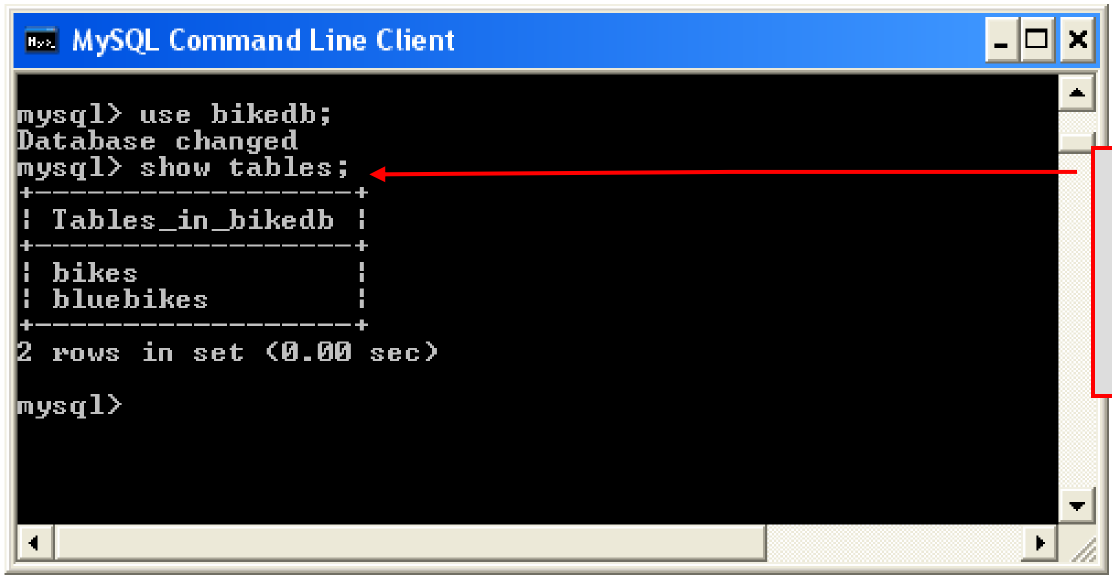


```
mysql> create table bikes (
    -> bikename varchar(30) NOT NULL,
    -> size int(2),
    -> color varchar(15),
    -> cost int(5),
    -> purchased date,
    -> mileage int(6),
    -> primary key (bikename)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> describe bikes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| bikename  | varchar(30) |      | PRI |         |       |
| size      | int(2)      | YES  |     | NULL    |       |
| color     | varchar(15) | YES  |     | NULL    |       |
| cost      | int(5)      | YES  |     | NULL    |       |
| purchased | date        | YES  |     | NULL    |       |
| mileage   | int(6)      | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

Specify which table's schema to describe. All information regarding the schema visible to the user is displayed.

Viewing the Relations of a Database

- Once a database has been selected you can see the relations (tables) within that database with the `show tables` command as illustrated below.



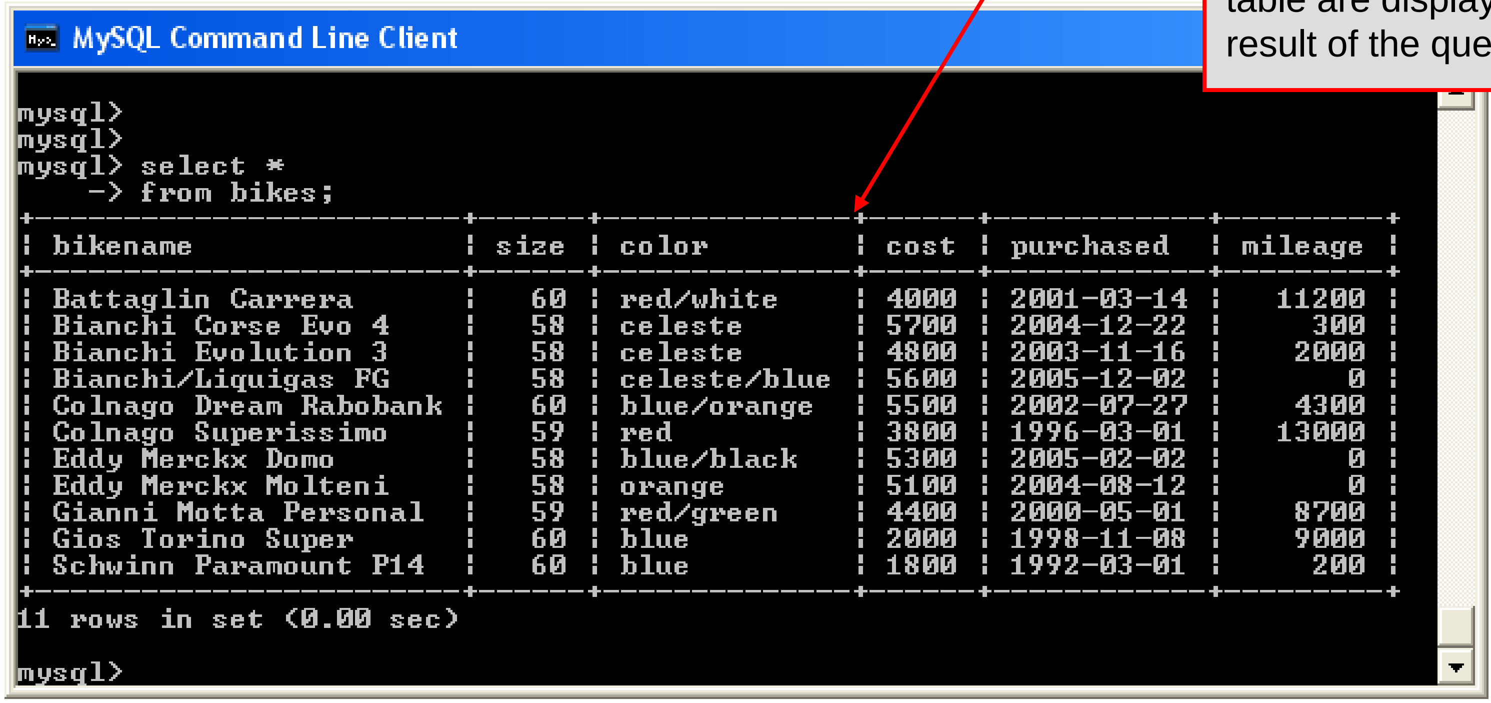
MySQL Command Line Client

```
mysql> use bikedb;
Database changed
mysql> show tables;
+------------------+
| Tables_in_bikedb |
+------------------+
| bikes            |
| bluebikes        |
+------------------+
2 rows in set (0.00 sec)

mysql>
```

Show tables command lists all the relations within a database visible to the user. There are two tables in this database.

Running a Simple Select Query in MySQL

- Within the MySQL monitor, running an SQL query is straight forward. The example below illustrates a simple selection query on the `bikes` table of the `bikedb` database.

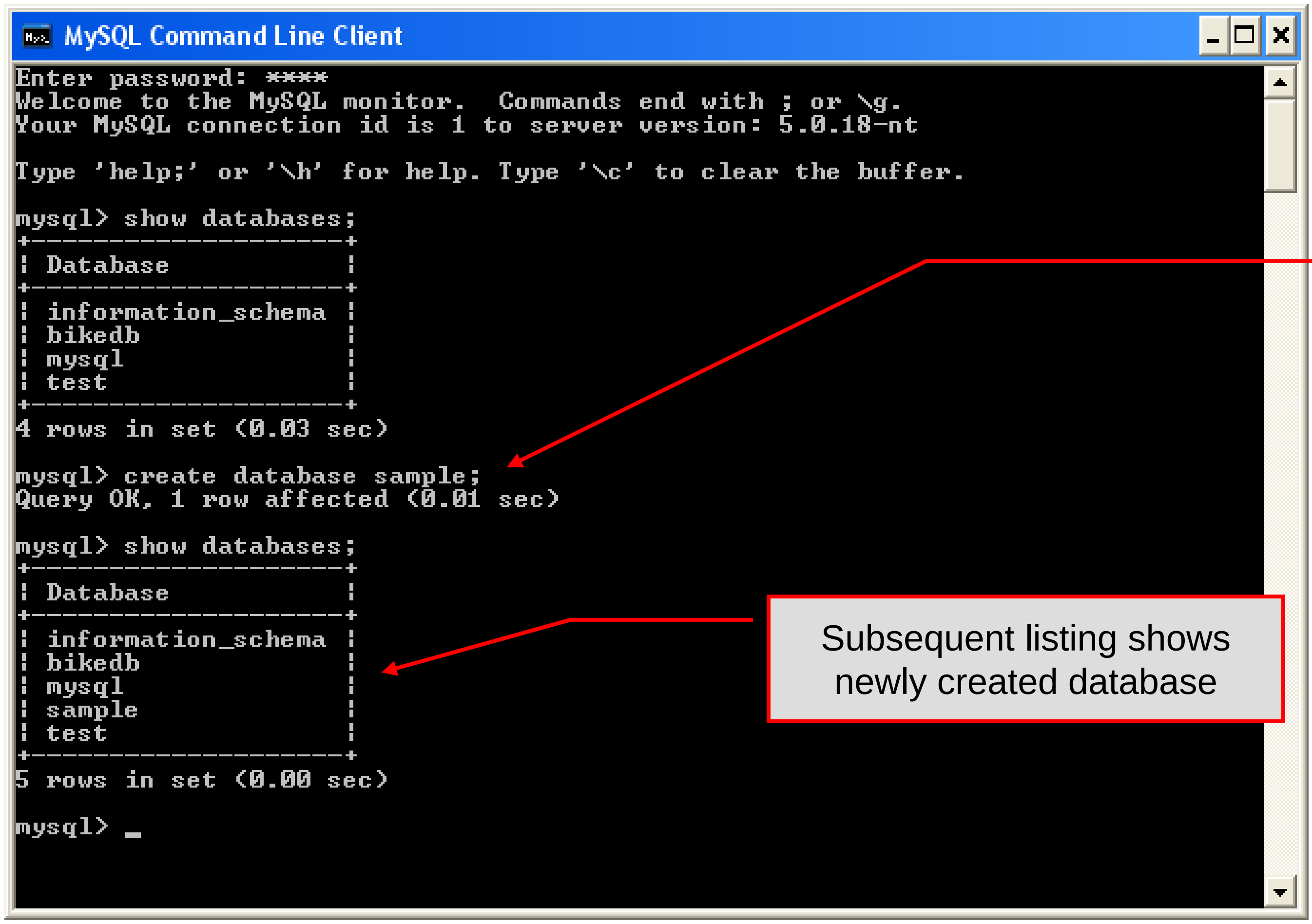


```
mysql>
mysql>
mysql> select *
    -> from bikes;
+------------------------+------+----------------+------+------------+---------+
| bikename               | size | color          | cost | purchased  | mileage |
+------------------------+------+----------------+------+------------+---------+
| Battaglin Carrera      |   60 | red/white      | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4    |   58 | celeste        | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3    |   58 | celeste        | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG    |   58 | celeste/blue   | 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank |   60 | blue/orange    | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo    |   59 | red            | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo       |   58 | blue/black     | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni    |   58 | orange         | 5100 | 2004-08-12 |       0 |
| Gianni Motta Personal  |   59 | red/green      | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super      |   60 | blue           | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14  |   60 | blue           | 1800 | 1992-03-01 |     200 |
+------------------------+------+----------------+------+------------+---------+
11 rows in set (0.00 sec)

mysql>
```

The tuples within the bikes table are displayed as the result of the query.

Creating a Database in MySQL

- From the MySQL monitor enter `create database <db name>`



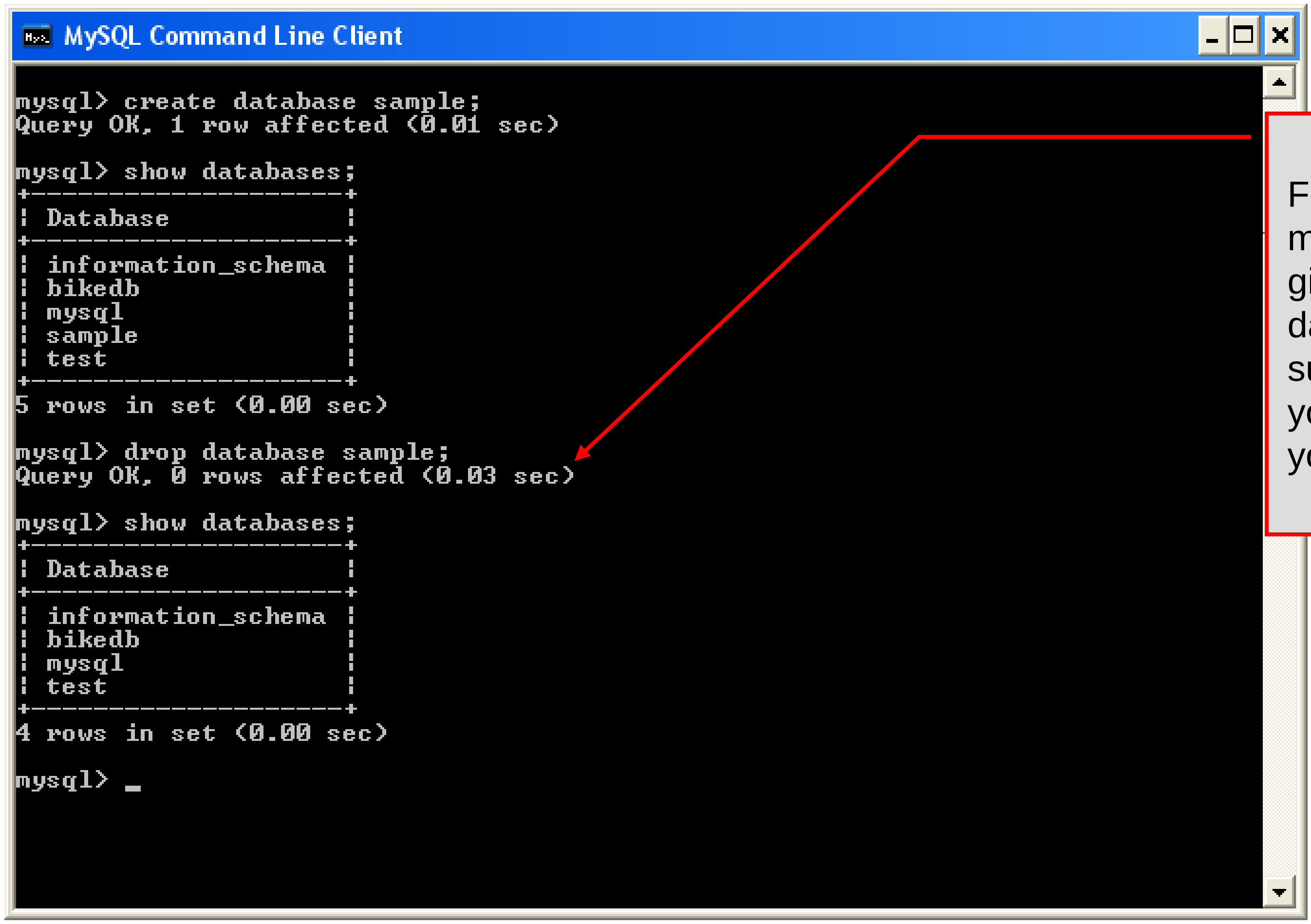
```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 5.0.18-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.03 sec)

mysql> create database sample;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| mysql              |
| sample             |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> _
```

Create new database from within MySQL monitor.

Subsequent listing shows newly created database

Dropping a Database in MySQL

- From the MySQL monitor execute the `drop database <db name>` command.

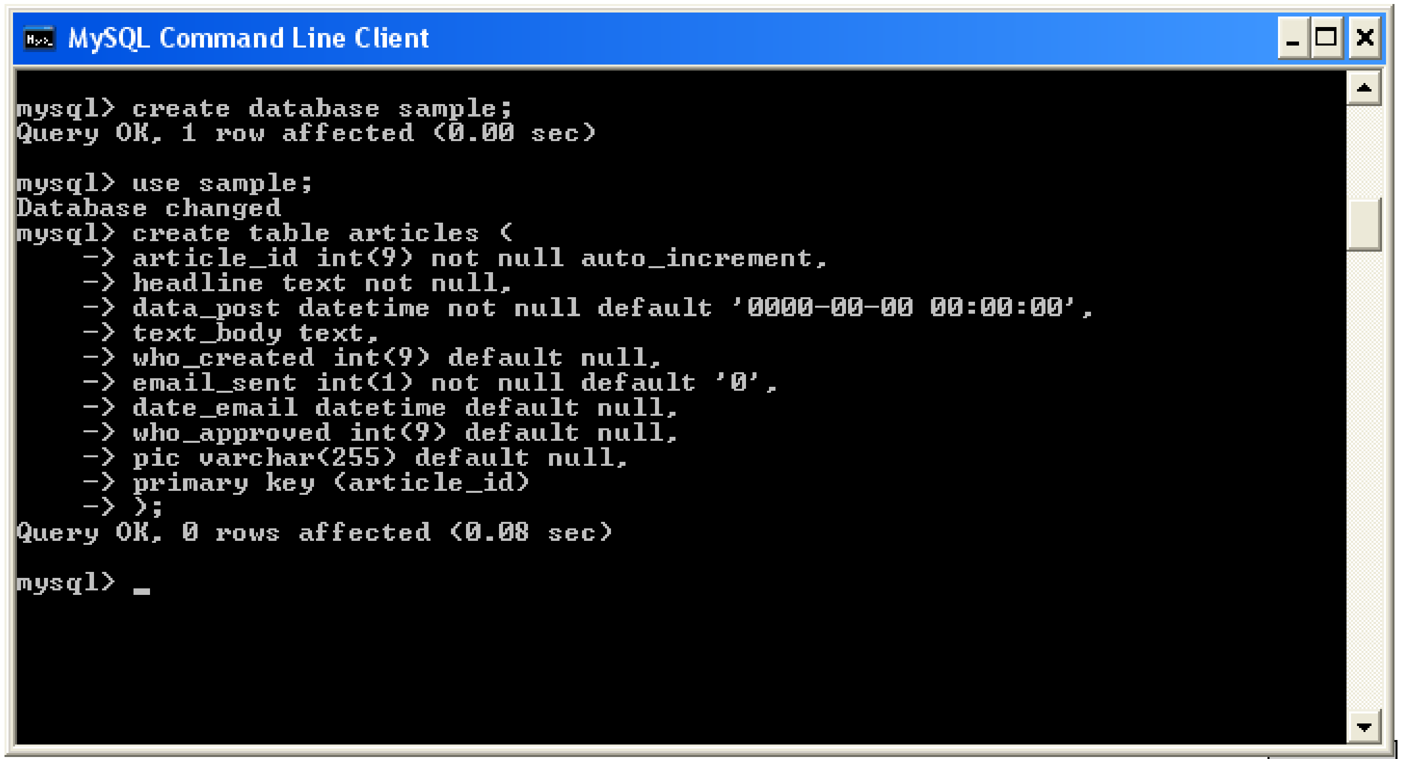


```
mysql> create database sample;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| mysql              |
| sample             |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> drop database sample;
Query OK, 0 rows affected (0.03 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)

mysql> _
```

From within the MySQL monitor, no warning is given when dropping a database. Be very sure that this is what you want to do before you do it.

Manipulating Tables in MySQL

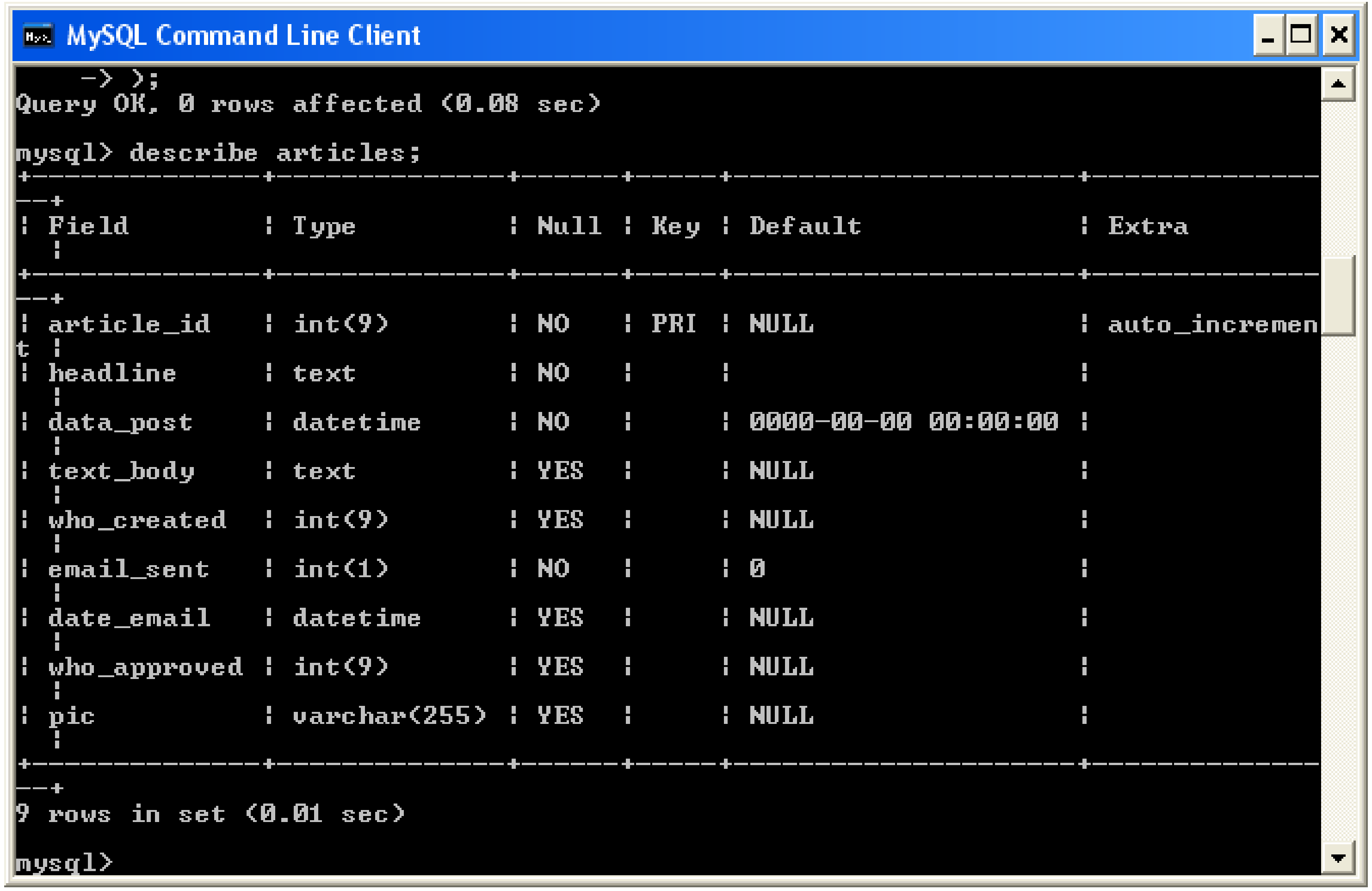
- The creation of a database does not place any relations into the database. Relations must be separately created.
- To create a table within a database, first select the database (or create one if you haven't already done so), then execute the `create table` command.



```
mysql> create database sample;
Query OK, 1 row affected (0.00 sec)

mysql> use sample;
Database changed
mysql> create table articles (
    -> article_id int(9) not null auto_increment,
    -> headline text not null,
    -> data_post datetime not null default '0000-00-00 00:00:00',
    -> text_body text,
    -> who_created int(9) default null,
    -> email_sent int(1) not null default '0',
    -> date_email datetime default null,
    -> who_approved int(9) default null,
    -> pic varchar(255) default null,
    -> primary key (article_id)
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> _
```

Manipulating Tables in MySQL (cont.)

Screen shot showing newly created table.



```
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> describe articles;
+--------------+--------------+------+-----+---------------------+---------------
--+
| Field        | Type         | Null | Key | Default             | Extra
  |
+--------------+--------------+------+-----+---------------------+---------------
--+
| article_id   | int(9)       | NO   | PRI | NULL                | auto_incremen
t |
| headline     | text         | NO   |     |                     |
  |
| data_post    | datetime     | NO   |     | 0000-00-00 00:00:00 |
  |
| text_body    | text         | YES  |     | NULL                |
  |
| who_created  | int(9)       | YES  |     | NULL                |
  |
| email_sent   | int(1)       | NO   |     | 0                   |
  |
| date_email   | datetime     | YES  |     | NULL                |
  |
| who_approved | int(9)       | YES  |     | NULL                |
  |
| pic          | varchar(255) | YES  |     | NULL                |
  |
+--------------+--------------+------+-----+---------------------+---------------
--+
9 rows in set (0.01 sec)

mysql>
```

Manipulating Tables in MySQL (cont.)

- The `create table` command has the following general format:

```
create [temporary] table
[if not exists] tablename
[(create_definition, ...)]
[table_options] [select_statement];
```

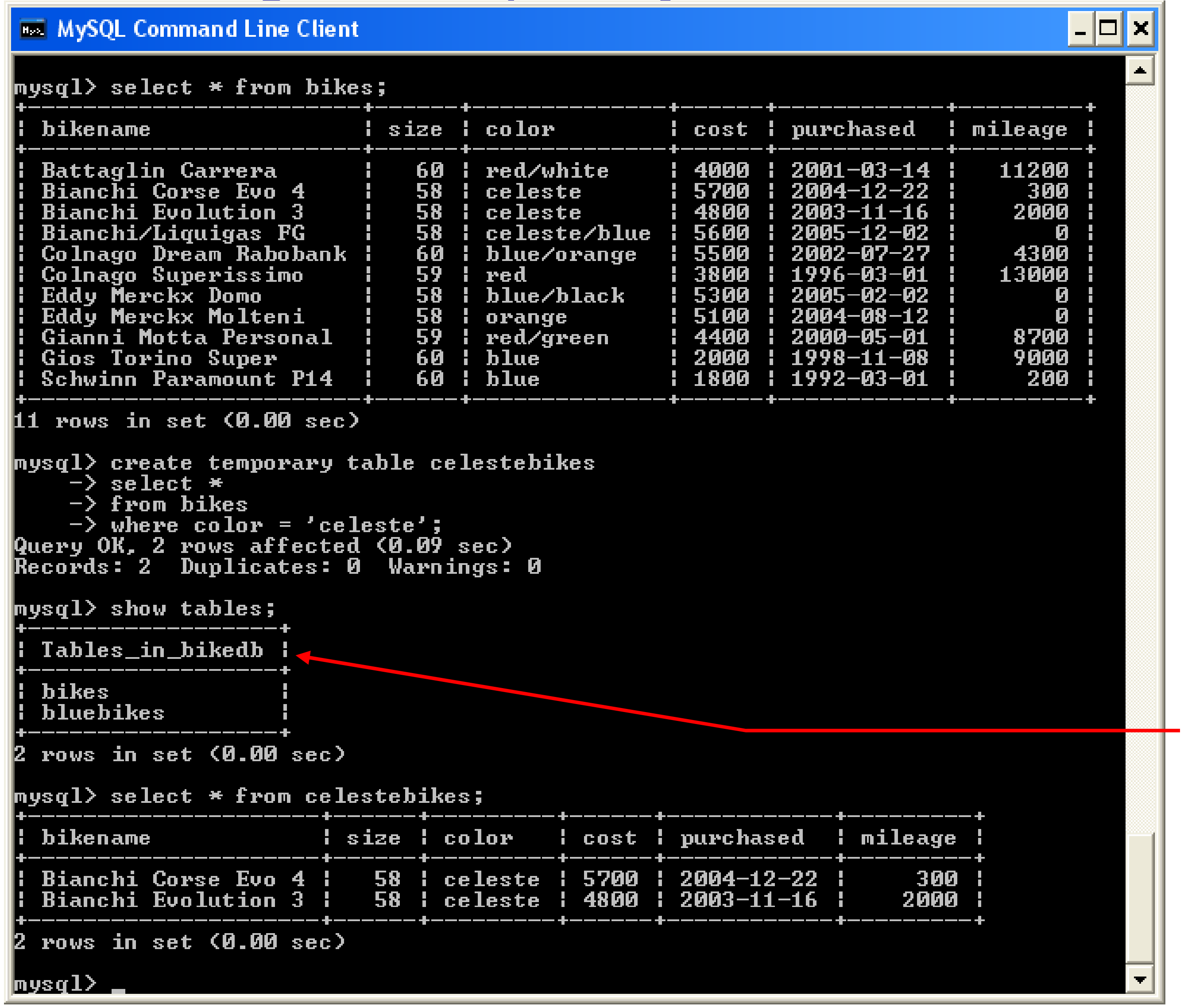
- If the [if not exists] clause is present, MySQL will produce an error message if a table with the specified name already exists in the database, otherwise the table is created.

Manipulating Tables in MySQL (cont.)

- A temporary table exists only for the life of the current database connection. It is automatically destroyed when the connection is closed or dies.
- Two different connections can use the same name for a temporary table without conflicting with one another.
- Temporary tables are most useful when queries get complex and intermediate results become useful. Also, versions of MySQL earlier than version 4.1 do not have subselect capability and temporary tables are a convenient way to simulate subselect query results.

Note: Non-root users require special permission to be able to create temporary tables. These users must have the Create_tmp_tables privilege set in the user grant table. We'll see more on this later.

Creating A Temporary Table From A Select Query



```
mysql> select * from bikes;
+-------------------------+------+-------------+------+------------+---------+
| bikename                | size | color       | cost | purchased  | mileage |
+-------------------------+------+-------------+------+------------+---------+
| Battaglin Carrera       |   60 | red/white   | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4     |   58 | celeste     | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3     |   58 | celeste     | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG     |   58 | celeste/blue| 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank  |   60 | blue/orange | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo     |   59 | red         | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo        |   58 | blue/black  | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni     |   58 | orange      | 5100 | 2004-08-12 |       0 |
| Gianni Motta Personal   |   59 | red/green   | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super       |   60 | blue        | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14   |   60 | blue        | 1800 | 1992-03-01 |     200 |
+-------------------------+------+-------------+------+------------+---------+
11 rows in set (0.00 sec)

mysql> create temporary table celestebikes
    -> select *
    -> from bikes
    -> where color = 'celeste';
Query OK, 2 rows affected (0.09 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> show tables;
+------------------+
| Tables_in_bikedb |
+------------------+
| bikes            |
| bluebikes        |
+------------------+
2 rows in set (0.00 sec)

mysql> select * from celestebikes;
+---------------------+------+---------+------+------------+---------+
| bikename            | size | color   | cost | purchased  | mileage |
+---------------------+------+---------+------+------------+---------+
| Bianchi Corse Evo 4 |   58 | celeste | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3 |   58 | celeste | 4800 | 2003-11-16 |    2000 |
+---------------------+------+---------+------+------------+---------+
2 rows in set (0.00 sec)

mysql> _
```

A SELECT query produces a result set which has been extracted from one or more tables. A table can be created with the results of this data using the create table command.

Notice that temporary tables do not appear in a table listing.

Manipulating Tables in MySQL (cont.)

- Recall that the `create table` command has the following general format:

```
create [temporary] table
[if not exists] tablename
[(create_definition, ...)]
[table_options]
[select_statement];
```

- The table options allow you to specify the MySQL table type. The table type can be anyone of the six types listed in the table on the next slide.

Manipulating Tables in MySQL (cont.)

Table Type	Description
ISAM	MySQL's original table handler
HEAP	The data for this table is only stored in memory
MyISAM	A binary portable table handler that has replaced ISAM
MERGE	A collection of MyISAM tables used as one table
BDB	Transaction-safe tables with page locking
InnoDB	Transaction-safe tables with row locking

MySQL Table Types

ISAM, HEAP, and MyISAM are available for MySQL versions 3.23.6 or later.

MERGE, BDB, and InnoDB are available for MySQL versions 4.0 and later.

Default table type is ISAM.

Altering A Table

- After a table has been created, it is possible to change the specifications of its schema. This is done through the `alter table` command:

```
alter table table_name action_list
```

  – Note: Changing the schema of a table in a database is not something that is done very often once the database has been created. The time for altering the schema is during the design phase. Altering the schema of an operational database is a very dangerous thing.

- Multiple changes to the table can be made at the same time by separating actions with commas in the `action_list`.

- The possible attribute (column) actions that can be used are shown in the table on the following slide.

Altering A Table (cont.)

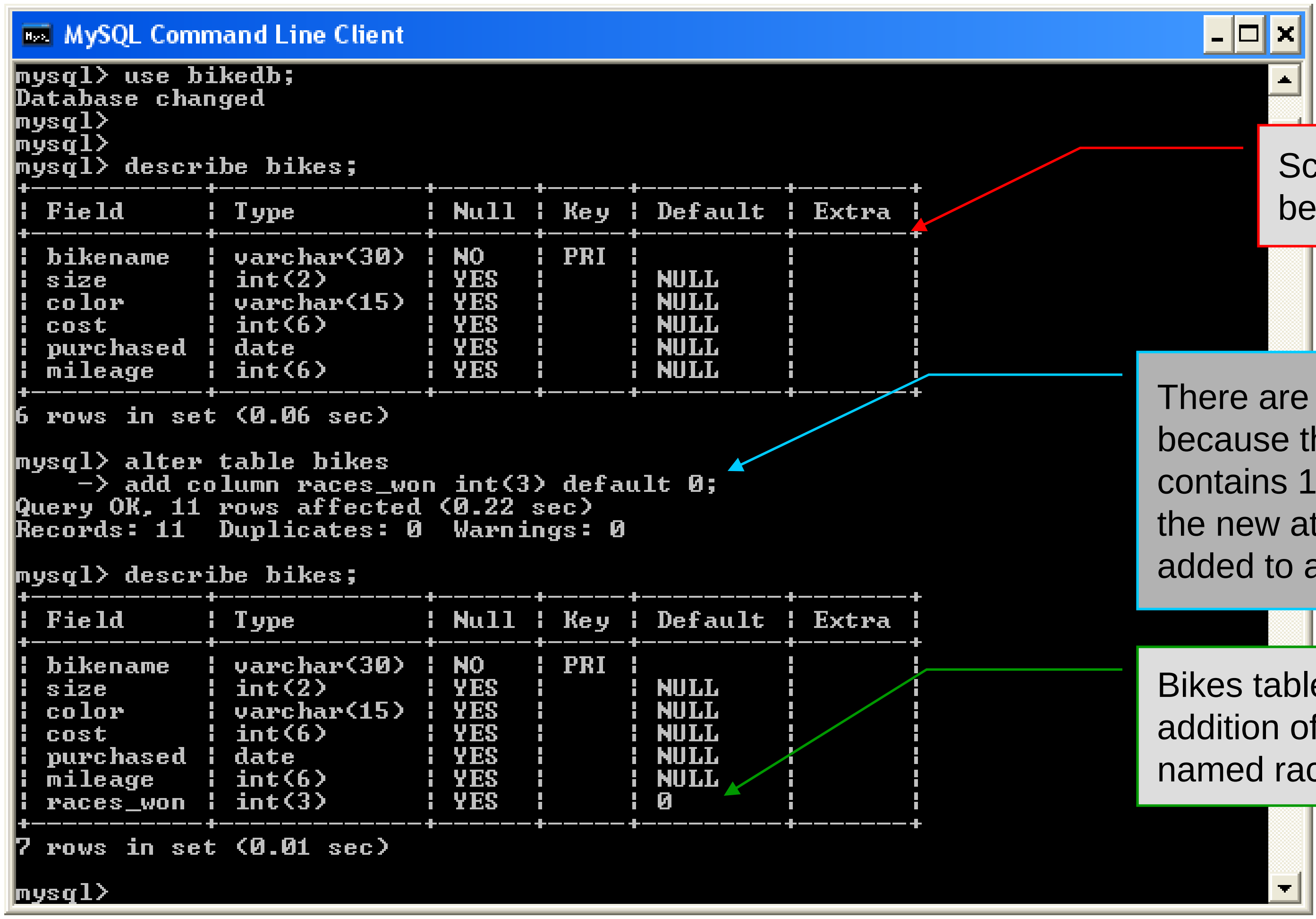
Action Syntax	Action Performed
`add [column]` *`column_declaration`* `[first | after` *`column_name`*`]`	Add a column to the table
`alter [column]` *`column_name`* `{set default` *`literal`* `| drop default}`	Specify new default value for a column or remove old default
`change [column]` *`column_name`* *`column_declaration`*	Modify column declaration with renaming of column
`modify [column]` *`column_declaration`*	Modify column declaration without renaming column
`drop [column]` *`column_name`*	Drop a column and all data contained within it.
`rename [as] new_table_name`	Rename a table
`table_options`	Change the table options

Actions performed by `alter table` (column related) command

`column_name` represents the current name of the column, *`column_declaration`* represents the new declaration, in the same format as if it were in a `create` command.

Altering A Table (cont.)

- The screen shot below shows an example of altering a table.



```
mysql> use bikedb;
Database changed
mysql>
mysql>
mysql> describe bikes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| bikename  | varchar(30) | NO   | PRI |         |       |
| size      | int(2)      | YES  |     | NULL    |       |
| color     | varchar(15) | YES  |     | NULL    |       |
| cost      | int(6)      | YES  |     | NULL    |       |
| purchased | date        | YES  |     | NULL    |       |
| mileage   | int(6)      | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.06 sec)

mysql> alter table bikes
    -> add column races_won int(3) default 0;
Query OK, 11 rows affected (0.22 sec)
Records: 11  Duplicates: 0  Warnings: 0

mysql> describe bikes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| bikename  | varchar(30) | NO   | PRI |         |       |
| size      | int(2)      | YES  |     | NULL    |       |
| color     | varchar(15) | YES  |     | NULL    |       |
| cost      | int(6)      | YES  |     | NULL    |       |
| purchased | date        | YES  |     | NULL    |       |
| mileage   | int(6)      | YES  |     | NULL    |       |
| races_won | int(3)      | YES  |     | 0       |       |
+-----------+-------------+------+-----+---------+-------+
7 rows in set (0.01 sec)

mysql>
```

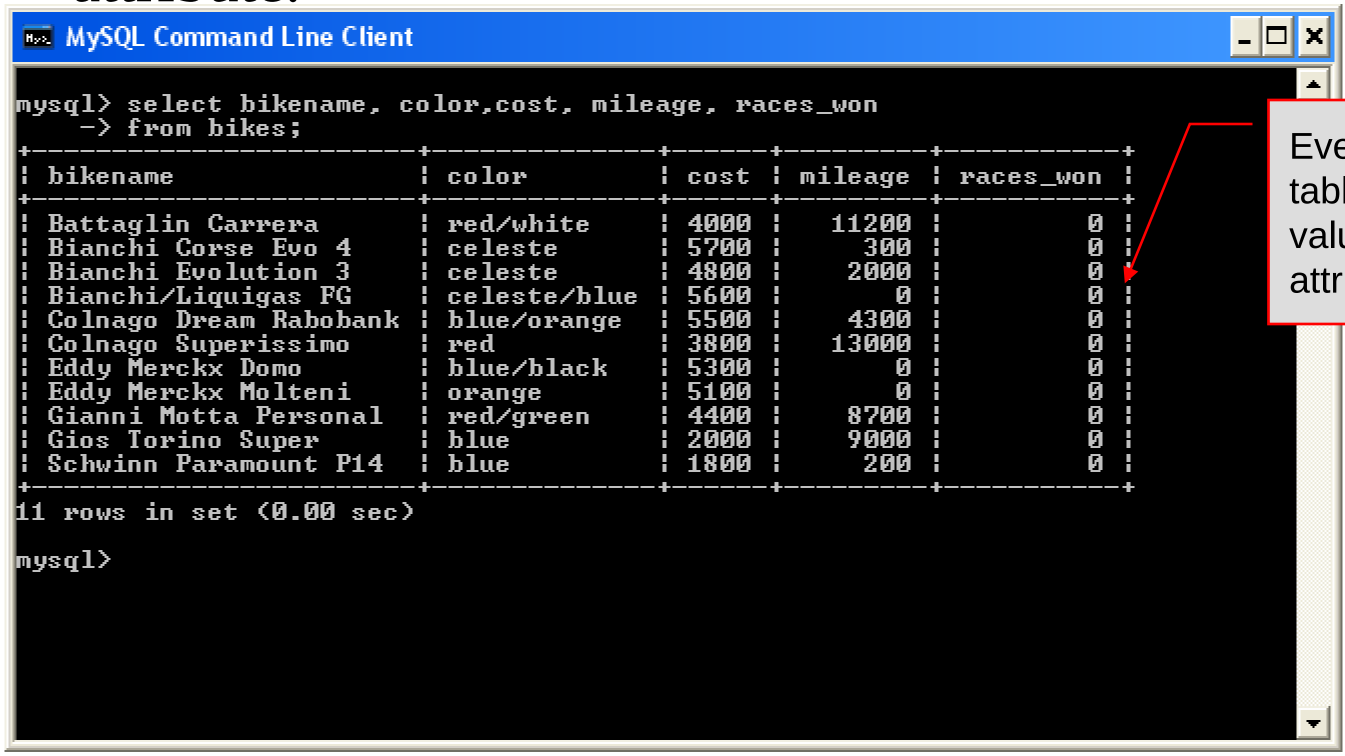
Schema of bikes before alteration

There are 11 rows affected because this table currently contains 11 tuples (rows) and the new attribute has been added to all rows.

Bikes table after the addition of a new column named races_won

Altering A Table (cont.)

- The screen shot below shows the tuples currently in the bikes table after the addition of the new attribute illustrating that all of the tuples have assumed the default value on the new attribute.

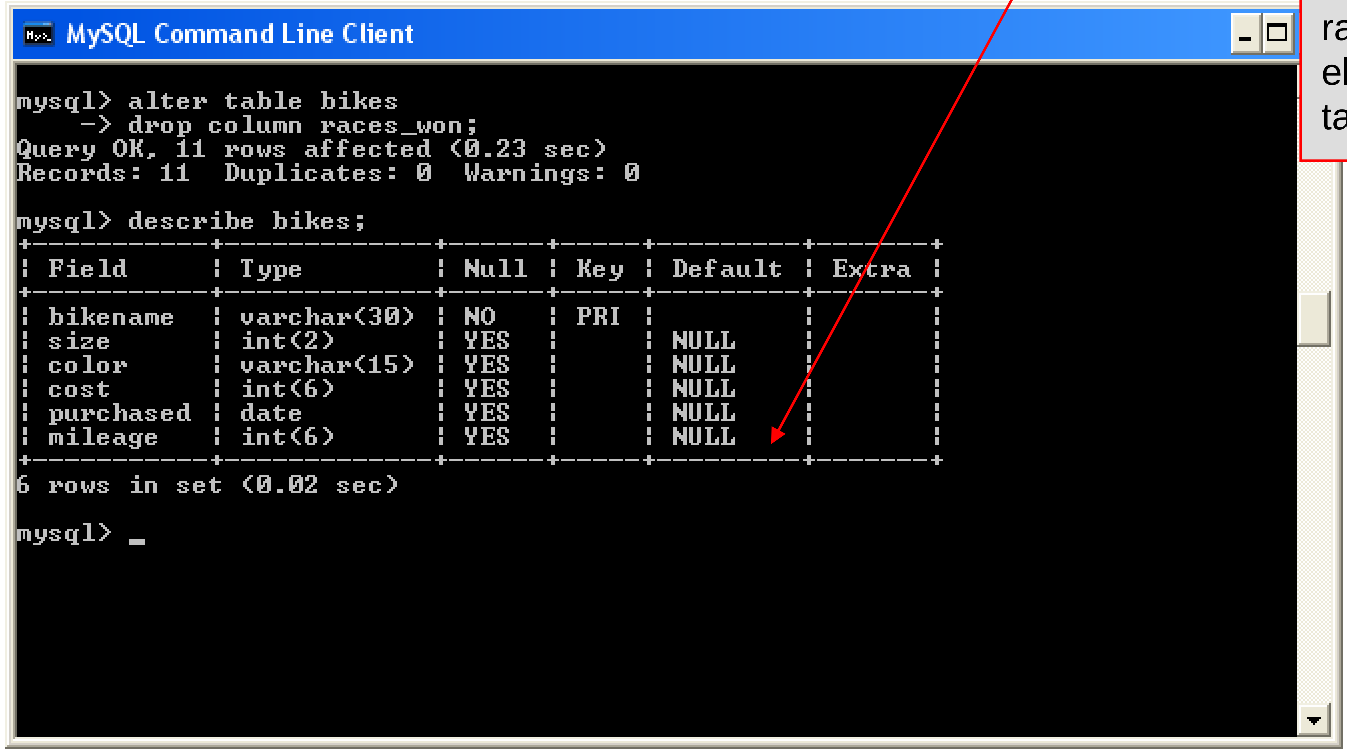


```
mysql> select bikename, color,cost, mileage, races_won
    -> from bikes;
+-------------------------+---------------+------+---------+-----------+
| bikename                | color         | cost | mileage | races_won |
+-------------------------+---------------+------+---------+-----------+
| Battaglin Carrera       | red/white     | 4000 |   11200 |         0 |
| Bianchi Corse Evo 4     | celeste       | 5700 |     300 |         0 |
| Bianchi Evolution 3     | celeste       | 4800 |    2000 |         0 |
| Bianchi/Liquigas FG     | celeste/blue  | 5600 |       0 |         0 |
| Colnago Dream Rabobank  | blue/orange   | 5500 |    4300 |         0 |
| Colnago Superissimo     | red           | 3800 |   13000 |         0 |
| Eddy Merckx Domo        | blue/black    | 5300 |       0 |         0 |
| Eddy Merckx Molteni     | orange        | 5100 |       0 |         0 |
| Gianni Motta Personal   | red/green     | 4400 |    8700 |         0 |
| Gios Torino Super       | blue          | 2000 |    9000 |         0 |
| Schwinn Paramount P14   | blue          | 1800 |     200 |         0 |
+-------------------------+---------------+------+---------+-----------+
11 rows in set (0.00 sec)

mysql>
```

Every tuple in the table has the default value for the new attribute.

Altering A Table (cont.)

- The screen shot below illustrates dropping a column from a table.
- Note that in general, this type of operation may not always be allowed due to constraint violations.

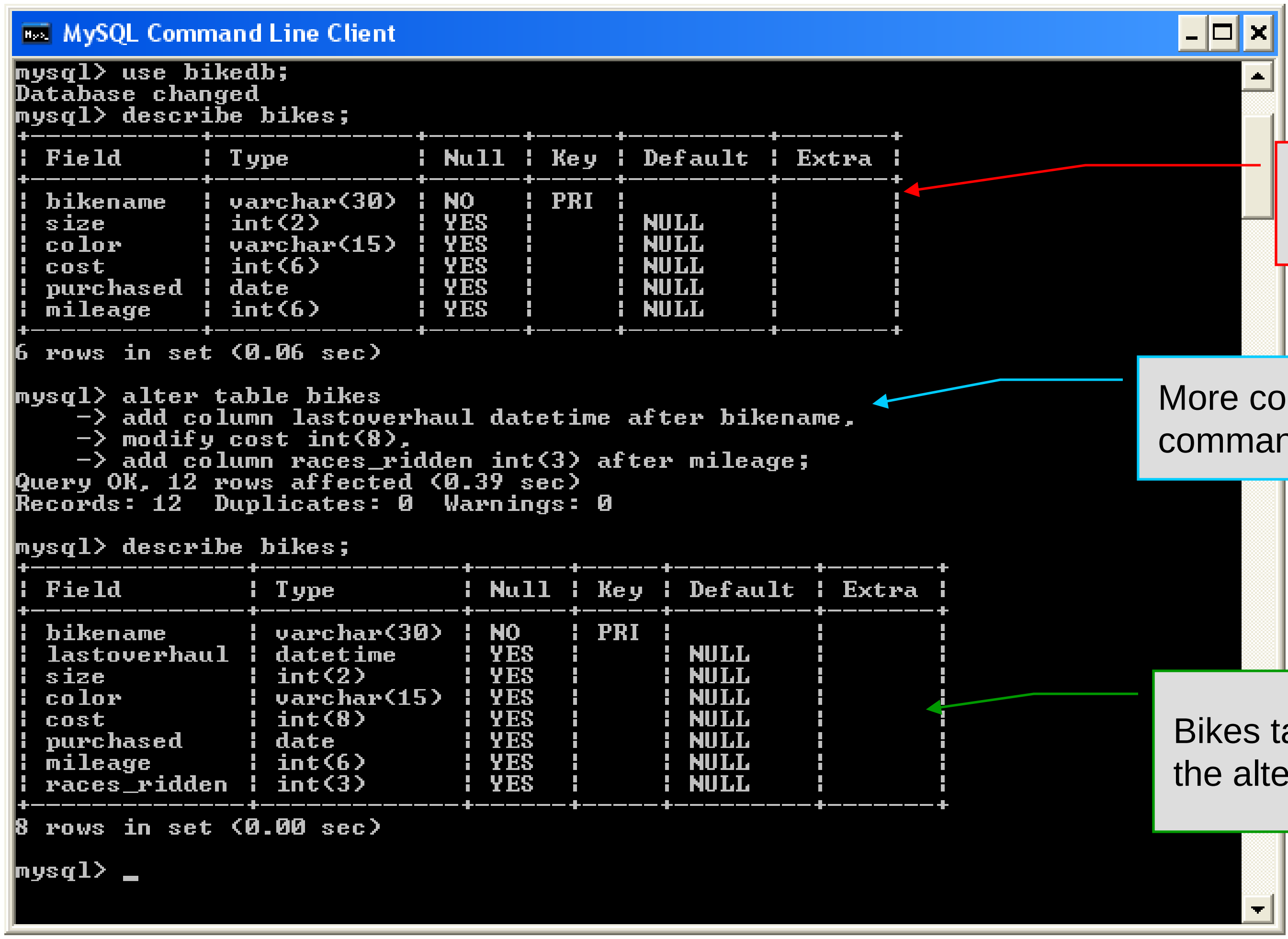


```
mysql> alter table bikes
    -> drop column races_won;
Query OK, 11 rows affected (0.23 sec)
Records: 11  Duplicates: 0  Warnings: 0

mysql> describe bikes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| bikename  | varchar(30) | NO   | PRI |         |       |
| size      | int(2)      | YES  |     | NULL    |       |
| color     | varchar(15) | YES  |     | NULL    |       |
| cost      | int(6)      | YES  |     | NULL    |       |
| purchased | date        | YES  |     | NULL    |       |
| mileage   | int(6)      | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)

mysql> _
```

The attribute races_won has been eliminated from the table.

Altering A Table (cont.)

- The screen shot below shows a more complicated example of altering a table.



```
mysql> use bikedb;
Database changed
mysql> describe bikes;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| bikename  | varchar(30) | NO   | PRI |         |       |
| size      | int(2)      | YES  |     | NULL    |       |
| color     | varchar(15) | YES  |     | NULL    |       |
| cost      | int(6)      | YES  |     | NULL    |       |
| purchased | date        | YES  |     | NULL    |       |
| mileage   | int(6)      | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.06 sec)

mysql> alter table bikes
    -> add column lastoverhaul datetime after bikename,
    -> modify cost int(8),
    -> add column races_ridden int(3) after mileage;
Query OK, 12 rows affected (0.39 sec)
Records: 12  Duplicates: 0  Warnings: 0

mysql> describe bikes;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| bikename     | varchar(30) | NO   | PRI |         |       |
| lastoverhaul | datetime    | YES  |     | NULL    |       |
| size         | int(2)      | YES  |     | NULL    |       |
| color        | varchar(15) | YES  |     | NULL    |       |
| cost         | int(8)      | YES  |     | NULL    |       |
| purchased    | date        | YES  |     | NULL    |       |
| mileage      | int(6)      | YES  |     | NULL    |       |
| races_ridden | int(3)      | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

mysql> _
```

Schema of bikes before alteration

More complicated alter table command.

Bikes table after the alteration

Inserting Data Into A Table

- Data can be entered into a MySQL table using either the `insert` or `replace` commands.
- The `insert` statement is the primary way of getting data into the database and has the following form:

Form 1

```
insert [low priority | delayed] [ignore] [into]table_name
        [set] column_name1 = expression1,
              column_name2 = expression2, …
```

Form 2

```
insert [low priority | delayed] [ignore] [into]table_name
        [(column_name,…)]values (expression,…), (…)…
```

Form 3

```
insert [low priority | delayed] [ignore] [into]table_name
        [(column_name,…)] select…
```

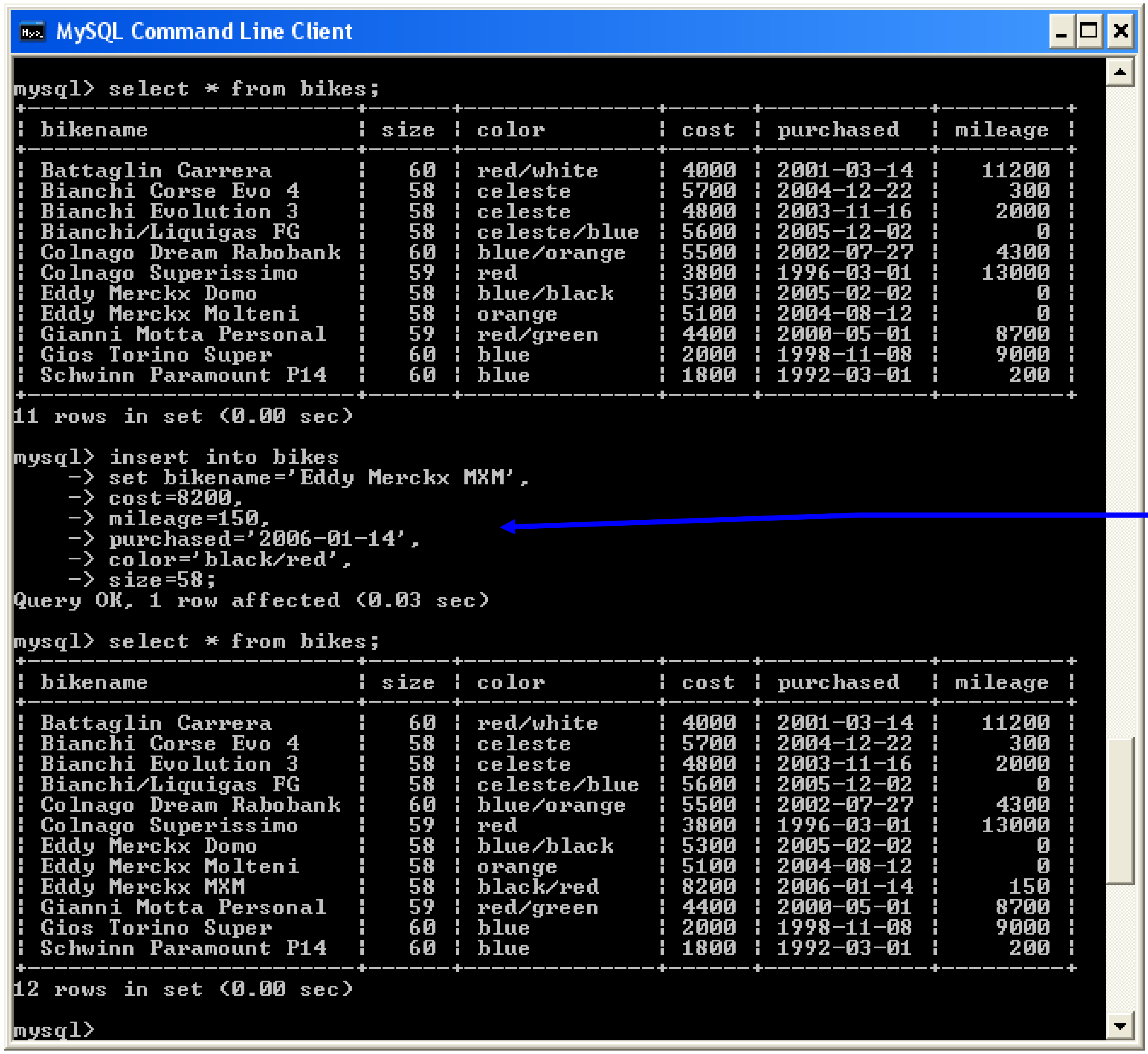
Inserting Data Into A Table (cont.)

- Form 1 of the insert statement is the most verbose, but also the most common. The `set` clause explicitly names each column and states what value (evaluated from each `expression`) should be put into the table.

- Form 2 (insert values) requires just a comma separated list of the data. For each row inserted, each data value must correspond with a column. In other words, the number of values listed must match the number of columns and the order of the value list must be the same as the columns. (In form 1, the order is not critical since each column is named.)

- Form 3 is used to insert data into a table which is the result set of a `select` statement. This is similar to the temporary table example seen earlier in the notes.

- The following couple of pages give some examples of the different forms of the `insert` command.

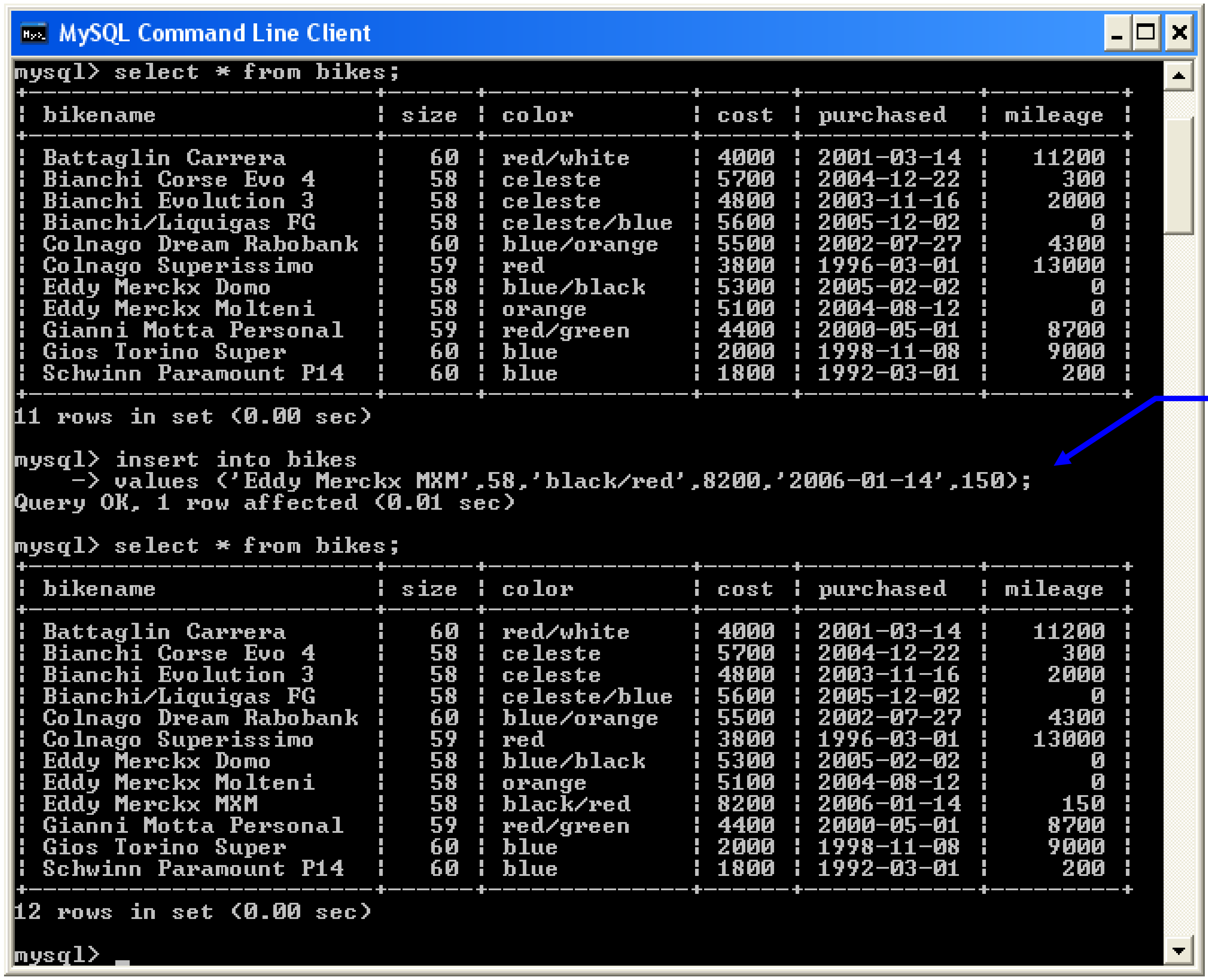


```
mysql> select * from bikes;
+-----------------------+------+--------------+------+------------+---------+
| bikename              | size | color        | cost | purchased  | mileage |
+-----------------------+------+--------------+------+------------+---------+
| Battaglin Carrera     |   60 | red/white    | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4   |   58 | celeste      | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3   |   58 | celeste      | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG   |   58 | celeste/blue | 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank |  60 | blue/orange  | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo   |   59 | red          | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo      |   58 | blue/black   | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni   |   58 | orange       | 5100 | 2004-08-12 |       0 |
| Gianni Motta Personal |   59 | red/green    | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super     |   60 | blue         | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14 |   60 | blue         | 1800 | 1992-03-01 |     200 |
+-----------------------+------+--------------+------+------------+---------+
11 rows in set (0.00 sec)

mysql> insert into bikes
    -> set bikename='Eddy Merckx MXM',
    -> cost=8200,
    -> mileage=150,
    -> purchased='2006-01-14',
    -> color='black/red',
    -> size=58;
Query OK, 1 row affected (0.03 sec)

mysql> select * from bikes;
+-----------------------+------+--------------+------+------------+---------+
| bikename              | size | color        | cost | purchased  | mileage |
+-----------------------+------+--------------+------+------------+---------+
| Battaglin Carrera     |   60 | red/white    | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4   |   58 | celeste      | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3   |   58 | celeste      | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG   |   58 | celeste/blue | 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank |  60 | blue/orange  | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo   |   59 | red          | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo      |   58 | blue/black   | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni   |   58 | orange       | 5100 | 2004-08-12 |       0 |
| Eddy Merckx MXM       |   58 | black/red    | 8200 | 2006-01-14 |     150 |
| Gianni Motta Personal |   59 | red/green    | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super     |   60 | blue         | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14 |   60 | blue         | 1800 | 1992-03-01 |     200 |
+-----------------------+------+--------------+------+------------+---------+
12 rows in set (0.00 sec)

mysql>
```

Examples: Inserting Data Into A Table

Using Form 1 for insertion – attribute order is not important.

Examples: Inserting Data Into A Table

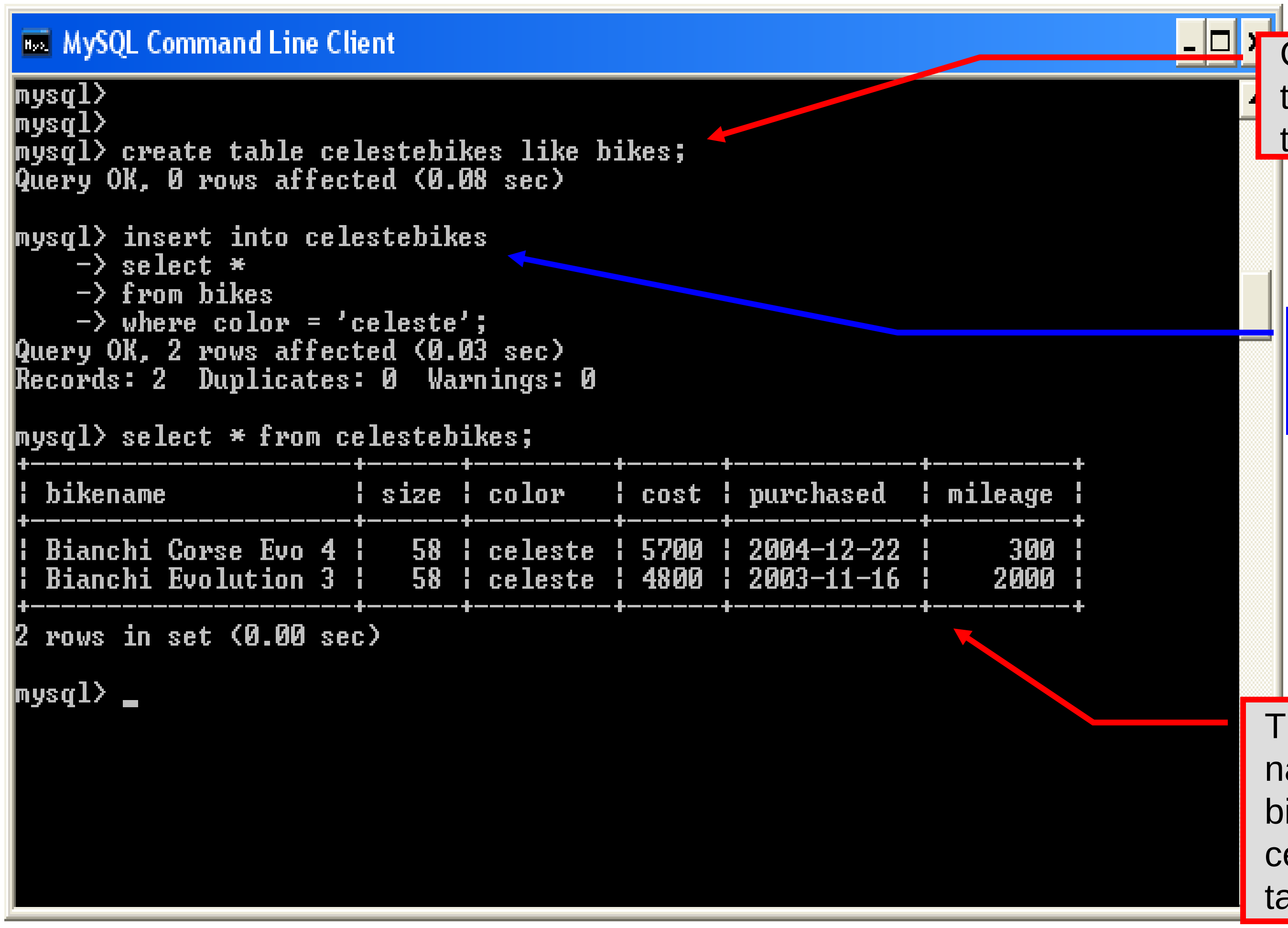


MySQL Command Line Client

```
mysql> select * from bikes;
+--------------------------+------+----------------+------+------------+---------+
| bikename                 | size | color          | cost | purchased  | mileage |
+--------------------------+------+----------------+------+------------+---------+
| Battaglin Carrera        |   60 | red/white      | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4      |   58 | celeste        | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3      |   58 | celeste        | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG      |   58 | celeste/blue   | 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank   |   60 | blue/orange    | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo      |   59 | red            | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo         |   58 | blue/black     | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni      |   58 | orange         | 5100 | 2004-08-12 |       0 |
| Gianni Motta Personal    |   59 | red/green      | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super        |   60 | blue           | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14    |   60 | blue           | 1800 | 1992-03-01 |     200 |
+--------------------------+------+----------------+------+------------+---------+
11 rows in set (0.00 sec)

mysql> insert into bikes
    -> values ('Eddy Merckx MXM',58,'black/red',8200,'2006-01-14',150);
Query OK, 1 row affected (0.01 sec)

mysql> select * from bikes;
+--------------------------+------+----------------+------+------------+---------+
| bikename                 | size | color          | cost | purchased  | mileage |
+--------------------------+------+----------------+------+------------+---------+
| Battaglin Carrera        |   60 | red/white      | 4000 | 2001-03-14 |   11200 |
| Bianchi Corse Evo 4      |   58 | celeste        | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3      |   58 | celeste        | 4800 | 2003-11-16 |    2000 |
| Bianchi/Liquigas FG      |   58 | celeste/blue   | 5600 | 2005-12-02 |       0 |
| Colnago Dream Rabobank   |   60 | blue/orange    | 5500 | 2002-07-27 |    4300 |
| Colnago Superissimo      |   59 | red            | 3800 | 1996-03-01 |   13000 |
| Eddy Merckx Domo         |   58 | blue/black     | 5300 | 2005-02-02 |       0 |
| Eddy Merckx Molteni      |   58 | orange         | 5100 | 2004-08-12 |       0 |
| Eddy Merckx MXM          |   58 | black/red      | 8200 | 2006-01-14 |     150 |
| Gianni Motta Personal    |   59 | red/green      | 4400 | 2000-05-01 |    8700 |
| Gios Torino Super        |   60 | blue           | 2000 | 1998-11-08 |    9000 |
| Schwinn Paramount P14    |   60 | blue           | 1800 | 1992-03-01 |     200 |
+--------------------------+------+----------------+------+------------+---------+
12 rows in set (0.00 sec)

mysql> _
```

Using Form 2 for insertion – attribute order is important.

Examples: Inserting Data Into A Table



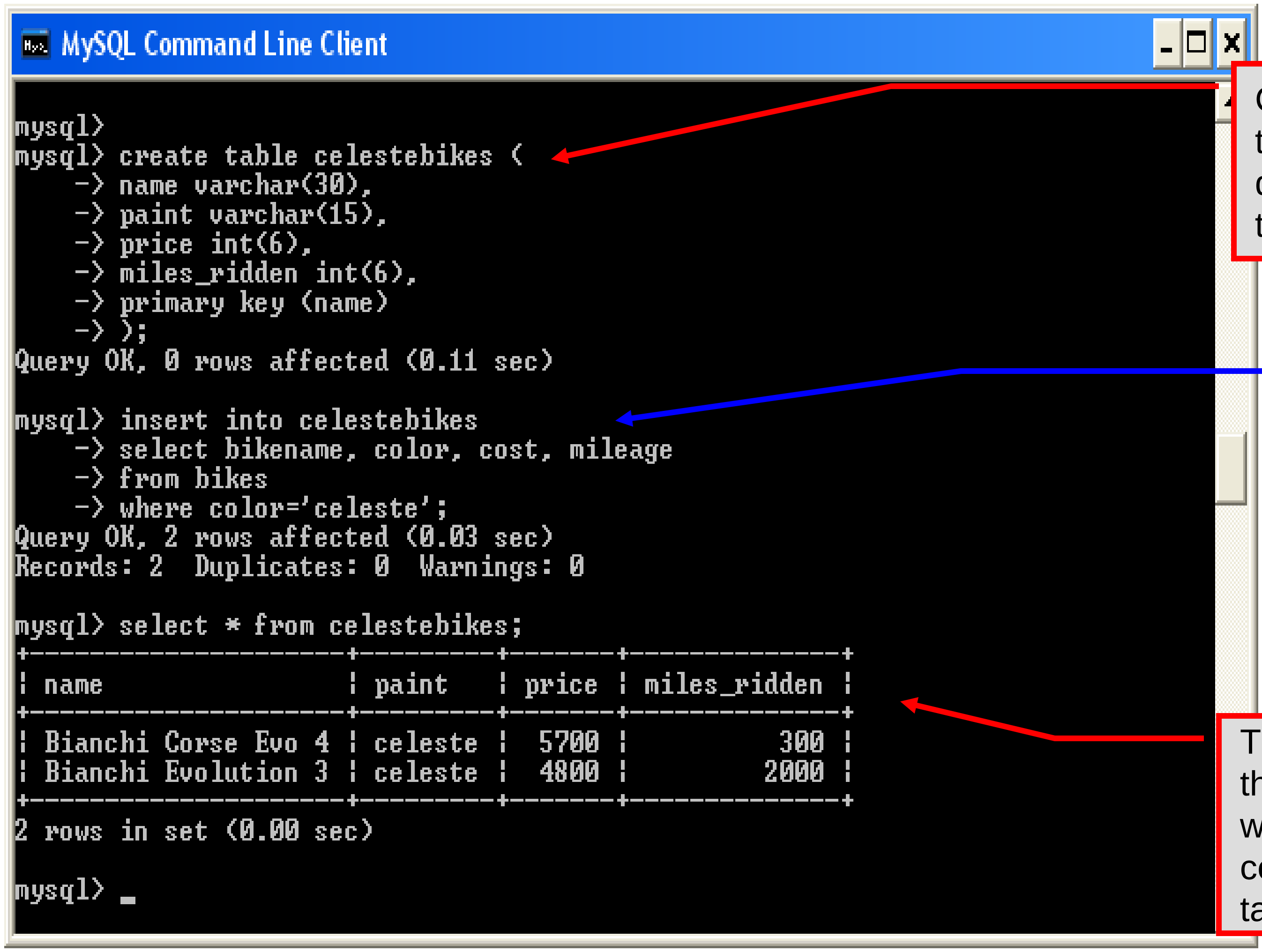
```
mysql>
mysql>
mysql> create table celestebikes like bikes;
Query OK, 0 rows affected (0.08 sec)

mysql> insert into celestebikes
    -> select *
    -> from bikes
    -> where color = 'celeste';
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from celestebikes;
+----------------------+------+---------+------+------------+---------+
| bikename             | size | color   | cost | purchased  | mileage |
+----------------------+------+---------+------+------------+---------+
| Bianchi Corse Evo 4  |   58 | celeste | 5700 | 2004-12-22 |     300 |
| Bianchi Evolution 3  |   58 | celeste | 4800 | 2003-11-16 |    2000 |
+----------------------+------+---------+------+------------+---------+
2 rows in set (0.00 sec)

mysql> _
```

Creates an initially empty table just like the bikes table

Using Form 3 for insertion

This table contains the name and cost of those bikes whose color was celeste from the source table.

Examples: Inserting Data Into A Table



```
mysql>
mysql> create table celestebikes (
    -> name varchar(30),
    -> paint varchar(15),
    -> price int(6),
    -> miles_ridden int(6),
    -> primary key (name)
    -> );
Query OK, 0 rows affected (0.11 sec)

mysql> insert into celestebikes
    -> select bikename, color, cost, mileage
    -> from bikes
    -> where color='celeste';
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from celestebikes;
+---------------------+---------+-------+--------------+
| name                | paint   | price | miles_ridden |
+---------------------+---------+-------+--------------+
| Bianchi Corse Evo 4 | celeste |  5700 |          300 |
| Bianchi Evolution 3 | celeste |  4800 |         2000 |
+---------------------+---------+-------+--------------+
2 rows in set (0.00 sec)

mysql> _
```

Create an initially empty table with a schema different from the base table.

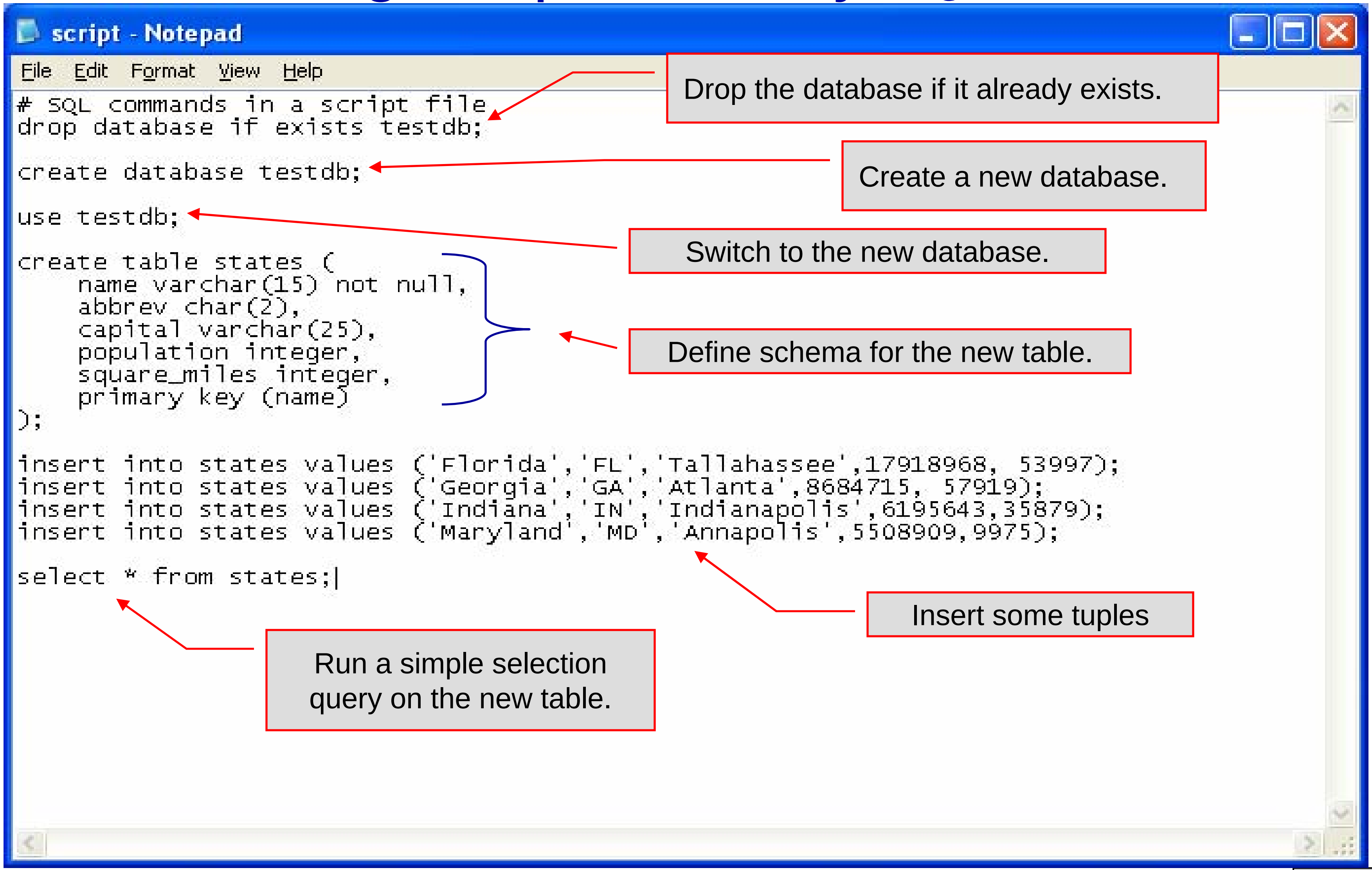
Using Form 3 for insertion

This table contains the those bike tuples whose color was celeste from the source table.

Using Scripts with MySQL

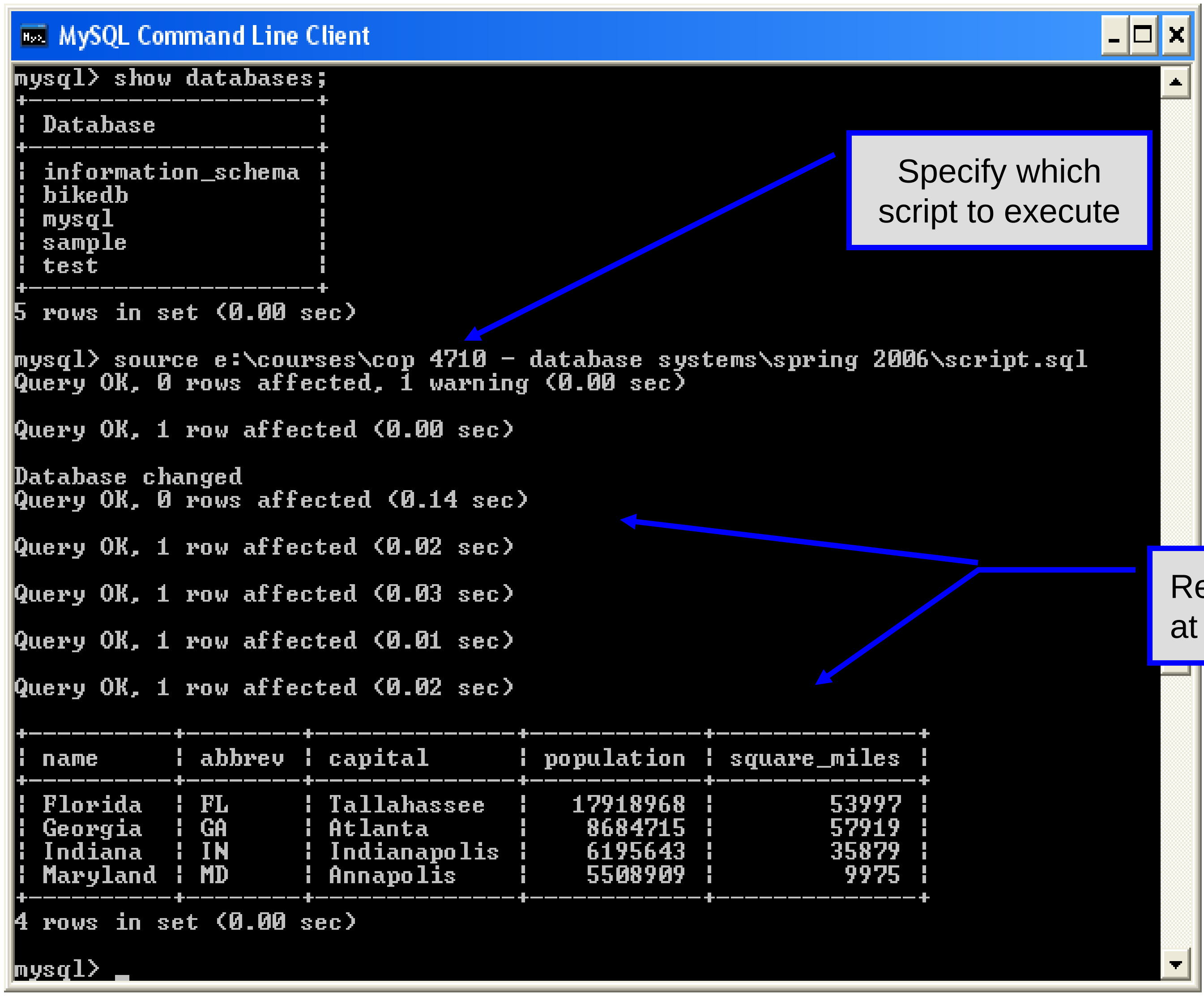
- Entering data to create sample databases using conventional SQL commands is tedious and prone to errors. A much simpler technique is to use scripts. The following illustrates two techniques for invoking scripts in MySQL.
- Create your script file using the text editor of your choice.
- Comments in the SQL script files begin with a # symbol.
- In the script file example shown on the next slide, I drop the database in the first SQL command. Without the if exists clause, this will generate an error if the database does not exist. The first time the script executes (or subsequent executions if the database is dropped independently) the error will be generated…simply ignore the error.

Using Scripts with MySQL (cont.)


script - Notepad
File Edit Format View Help
Drop the database if it already exists.
Create a new database.
Switch to the new database.
Define schema for the new table.
Insert some tuples
Run a simple selection query on the new table.

```
# SQL commands in a script file
drop database if exists testdb;

create database testdb;

use testdb;

create table states (
    name varchar(15) not null,
    abbrev char(2),
    capital varchar(25),
    population integer,
    square_miles integer,
    primary key (name)
);

insert into states values ('Florida','FL','Tallahassee',17918968, 53997);
insert into states values ('Georgia','GA','Atlanta',8684715, 57919);
insert into states values ('Indiana','IN','Indianapolis',6195643,35879);
insert into states values ('Maryland','MD','Annapolis',5508909,9975);

select * from states;
```

Using Scripts with MySQL (cont.)



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bikedb             |
| mysql              |
| sample             |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> source e:\courses\cop 4710 - database systems\spring 2006\script.sql
Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Database changed
Query OK, 0 rows affected (0.14 sec)

Query OK, 1 row affected (0.02 sec)

Query OK, 1 row affected (0.03 sec)

Query OK, 1 row affected (0.01 sec)

Query OK, 1 row affected (0.02 sec)

+----------+--------+--------------+------------+--------------+
| name     | abbrev | capital      | population | square_miles |
+----------+--------+--------------+------------+--------------+
| Florida  | FL     | Tallahassee  |   17918968 |        53997 |
| Georgia  | GA     | Atlanta      |    8684715 |        57919 |
| Indiana  | IN     | Indianapolis |    6195643 |        35879 |
| Maryland | MD     | Annapolis    |    5508909 |         9975 |
+----------+--------+--------------+------------+--------------+
4 rows in set (0.00 sec)

mysql> _
```

Specify which script to execute

Results of select query at end of script.

Importing Data Using the `mysqlimport` Utility

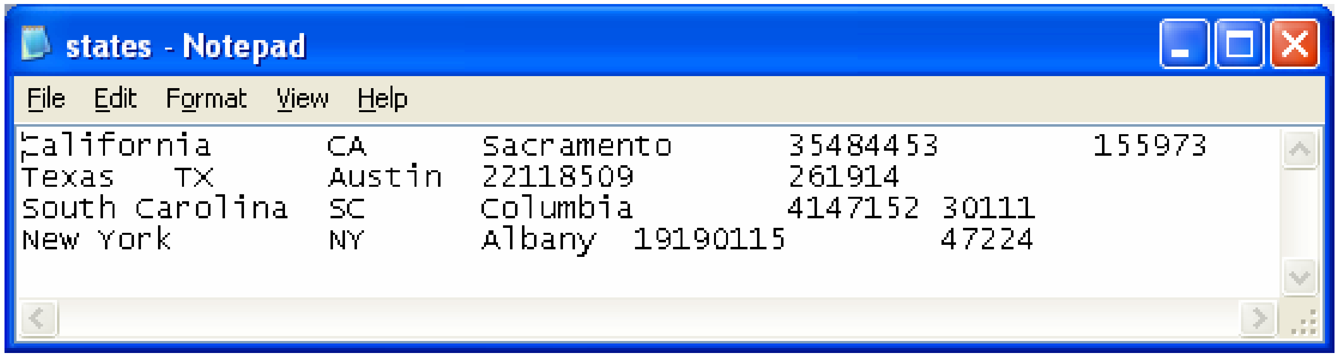
- As with many things in MySQL there are several ways to accomplish a specific task. For getting data into tables, the `mysqlimport` utility is also useful.
- The `mysqlimport` utility reads a range of data formats, including comma- and tab- delimited, and inserts the data into a specified database table. The syntax for `mysqlimport` is:

```
mysqlimport [options] database_name file1 file2 …
```

- This utility is designed to be invoked from the command line.
- The name of the file (excluding the extension) must match the name of the database table into which the data import will occur. Failure to match names will result in an error.

Importing Data Using the `mysqlimportUtility` (cont.)

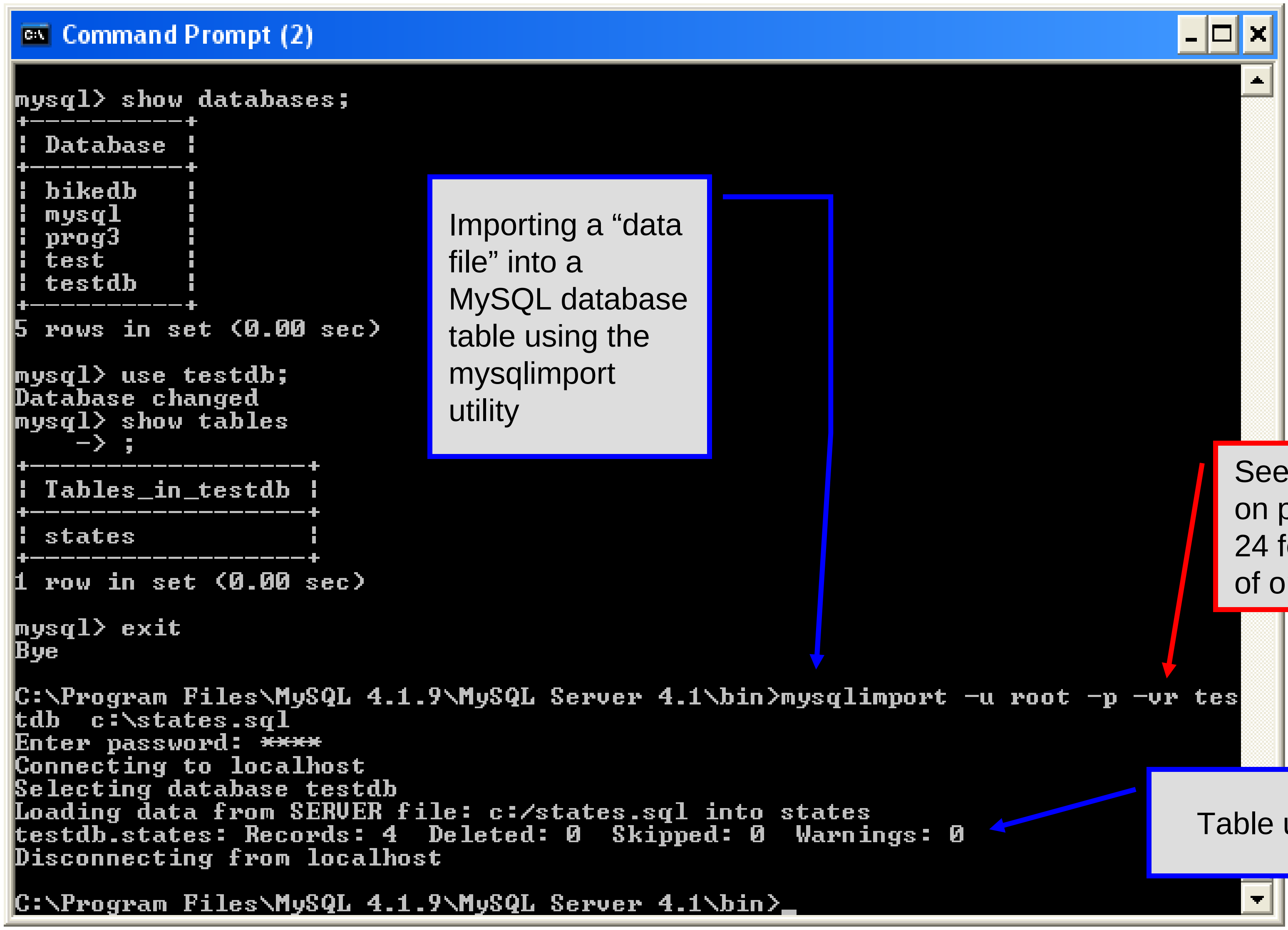
- The file shown below was created to import additional data into the states table within the testdb database used in the previous example.



```
California      CA      Sacramento      35484453        155973
Texas   TX      Austin  22118509        261914
South Carolina  SC      Columbia        4147152 30111
New York        NY      Albany  19190115        47224
```

- In this case, the default field delimiter (tab), default field enclosure (nothing), and the default line delimiter (\n) were used. Many options are available and are illustrated in the table on pages 60-61.

Importing Data Using the `mysqlimportUtility`



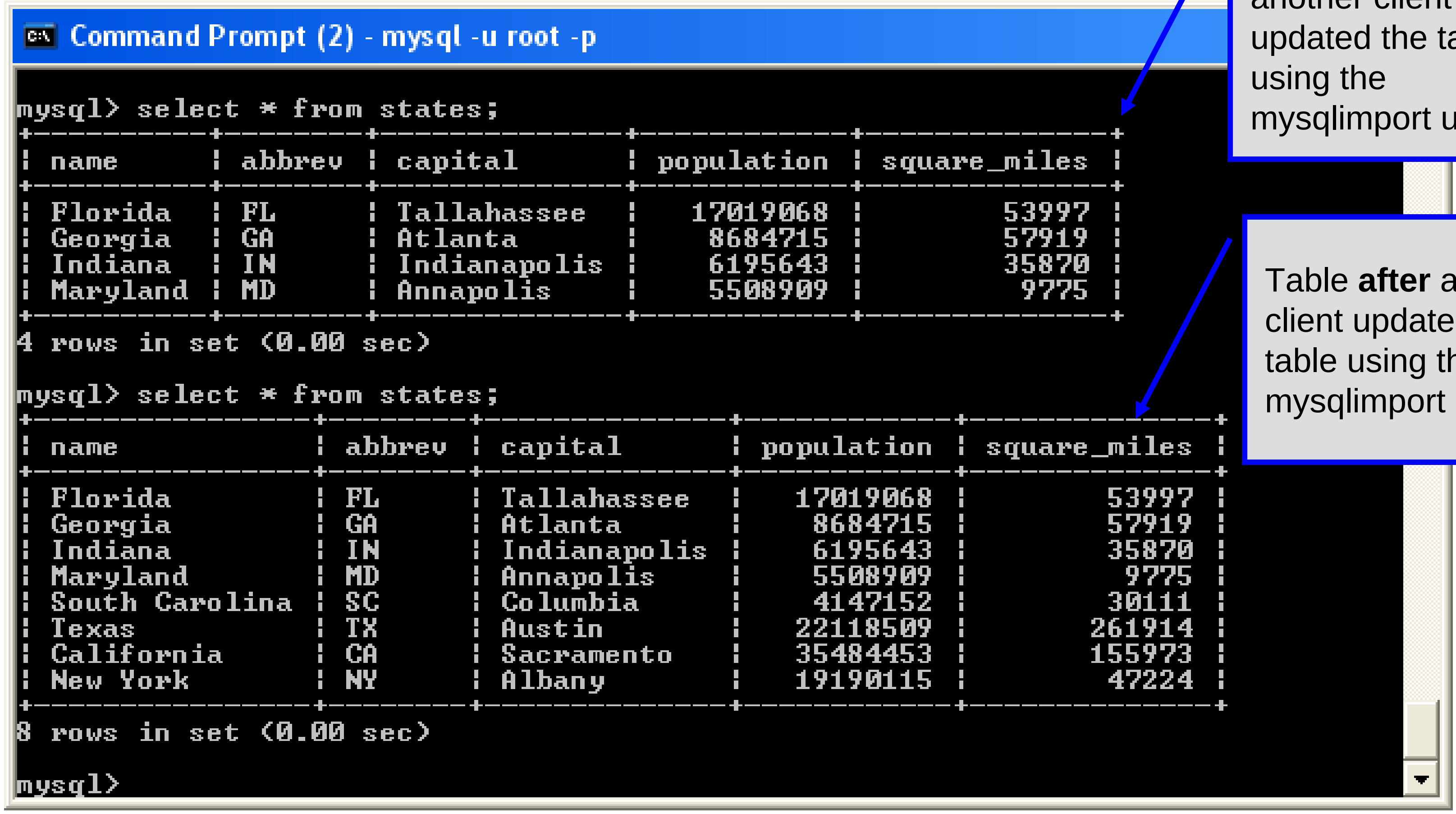
```
mysql> show databases;
+----------+
| Database |
+----------+
| bikedb   |
| mysql    |
| prog3    |
| test     |
| testdb   |
+----------+
5 rows in set (0.00 sec)

mysql> use testdb;
Database changed
mysql> show tables
    -> ;
+------------------+
| Tables_in_testdb |
+------------------+
| states           |
+------------------+
1 row in set (0.00 sec)

mysql> exit
Bye

C:\Program Files\MySQL 4.1.9\MySQL Server 4.1\bin>mysqlimport -u root -p -vr tes
tdb  c:\states.sql
Enter password: *****
Connecting to localhost
Selecting database testdb
Loading data from SERVER file: c:/states.sql into states
testdb.states: Records: 4  Deleted: 0  Skipped: 0  Warnings: 0
Disconnecting from localhost

C:\Program Files\MySQL 4.1.9\MySQL Server 4.1\bin>_
```

Importing a “data file” into a MySQL database table using the mysqlimport utility

See tables on pages 23-24 for listing of options.

Table updated

Importing Data Using the `mysqlimportUtility`



```
mysql> select * from states;
+----------+--------+--------------+------------+--------------+
| name     | abbrev | capital      | population | square_miles |
+----------+--------+--------------+------------+--------------+
| Florida  | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia  | GA     | Atlanta      |    8684715 |        57919 |
| Indiana  | IN     | Indianapolis |    6195643 |        35870 |
| Maryland | MD     | Annapolis    |    5508909 |         9775 |
+----------+--------+--------------+------------+--------------+
4 rows in set (0.00 sec)

mysql> select * from states;
+----------------+--------+--------------+------------+--------------+
| name           | abbrev | capital      | population | square_miles |
+----------------+--------+--------------+------------+--------------+
| Florida        | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia        | GA     | Atlanta      |    8684715 |        57919 |
| Indiana        | IN     | Indianapolis |    6195643 |        35870 |
| Maryland       | MD     | Annapolis    |    5508909 |         9775 |
| South Carolina | SC     | Columbia     |    4147152 |        30111 |
| Texas          | TX     | Austin       |   22118509 |       261914 |
| California     | CA     | Sacramento   |   35484453 |       155973 |
| New York       | NY     | Albany       |   19190115 |        47224 |
+----------------+--------+--------------+------------+--------------+
8 rows in set (0.00 sec)

mysql>
```

Table **before** another client updated the table using the mysqlimport utility.

Table **after** another client updated the table using the mysqlimport utility.

`mysqlimportUtility` Options

Option	Action
-r or –replace	Causes imported rows to overwrite existing rows if they have the same unique key value.
-i or –ignore	Ignores rows that have the same unique key value as existing rows.
-f or –force	Forces mysqlimport to continue inserting data even if errors are encountered.
-l or –lock	Lock each table before importing (a good idea in general and especially on a busy server).
-d or –delete	Empty the table before inserting data.
--fields-terminated-by=‘char’	Specify the separator used between values of the same row, default \t (tab).
--fields-enclosed-by=‘char’	Specify the delimiter that encloses each field, default is none.

`mysqlimport` Utility Options (cont.)

Option	Action
--fields-optionally-enclosed-by=‘char’	Same as –fields-enclosed-by, but delimiter is used only to enclosed string-type columns, default is none.
--fields-escaped-by=‘char’	Specify the escape character placed before special characters; default is \.
--lines-terminated-by=‘char’	Specify the separator used to terminate each row of data, default is \n (newline).
-u or –user	Specify your username
-p or –password	Specify your password
-h or –host	Import into MySQL on the named host; default is localhost.
-s or –silent	Silent mode, output appears only when errors occur.
-v or –verbose	Verbose mode, print more commentary on action.
-? or –help	Print help message and exit

Importing Data From A File With SQL Statement `Load Data Infile`

- Using the utility `mysqlimport` to load data into a table from an external file works well if the user has access to a command window or command line.
- If you have access via a connection to only the MySQL database, or you are importing data from within an executing application, you will need to use the SQL statement `Load Data Infile`.
- The Load Data Infile statement also provides a bit more flexibility since the file name does not need to match the table name. Other than that the options are basically the same and the same results are accomplished.
- The example on page 65 illustrates this SQL command which is available in MySQL.

Importing Data From A File With SQL Statement `Load Data Infile` (cont.)

- The basic form of the Load Data Infile statement is:

```
LOAD DATA [LOW_PRIORITY | CONCURRENT] [LOCAL] INFILE 'filename'
[REPLACE | IGNORE]
INTO TABLE tablename
[FIELDS
     [TERMINATED BY 'char']
     [ [OPTIONALLY] ENCLOSED BY 'char']
     [ESCAPED BY '\char']  ]
[LINES
     [STARTING BY 'char']
     [TERMINATED BY 'char']  ]
[IGNORE number LINES]
[(column_name, … )]
```

Either allow concurrent update or block until no other clients are reading from the specified table. See page 69.

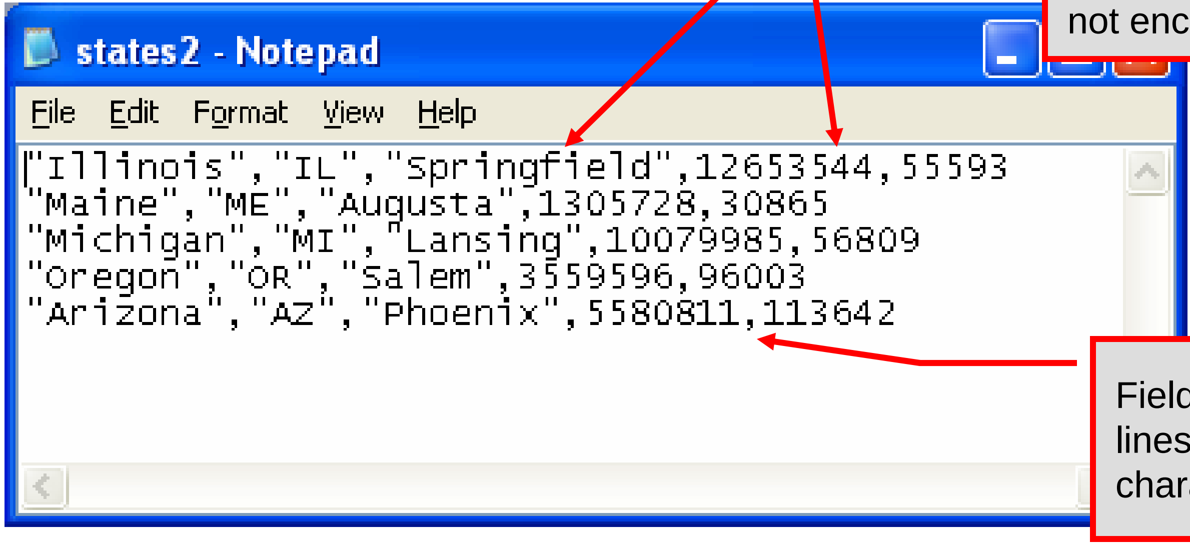
Same as –r and –i options in mysqlimport utility – either replace or ignore rows with duplicate keys.

Sets the characters that delimit and enclose the fields and lines in the data file. Similar to mysqlimport syntax.

Ignores lines at the start of the file (miss header info)

Used to load only certain columns (not entire rows)

Load Data Infile Example

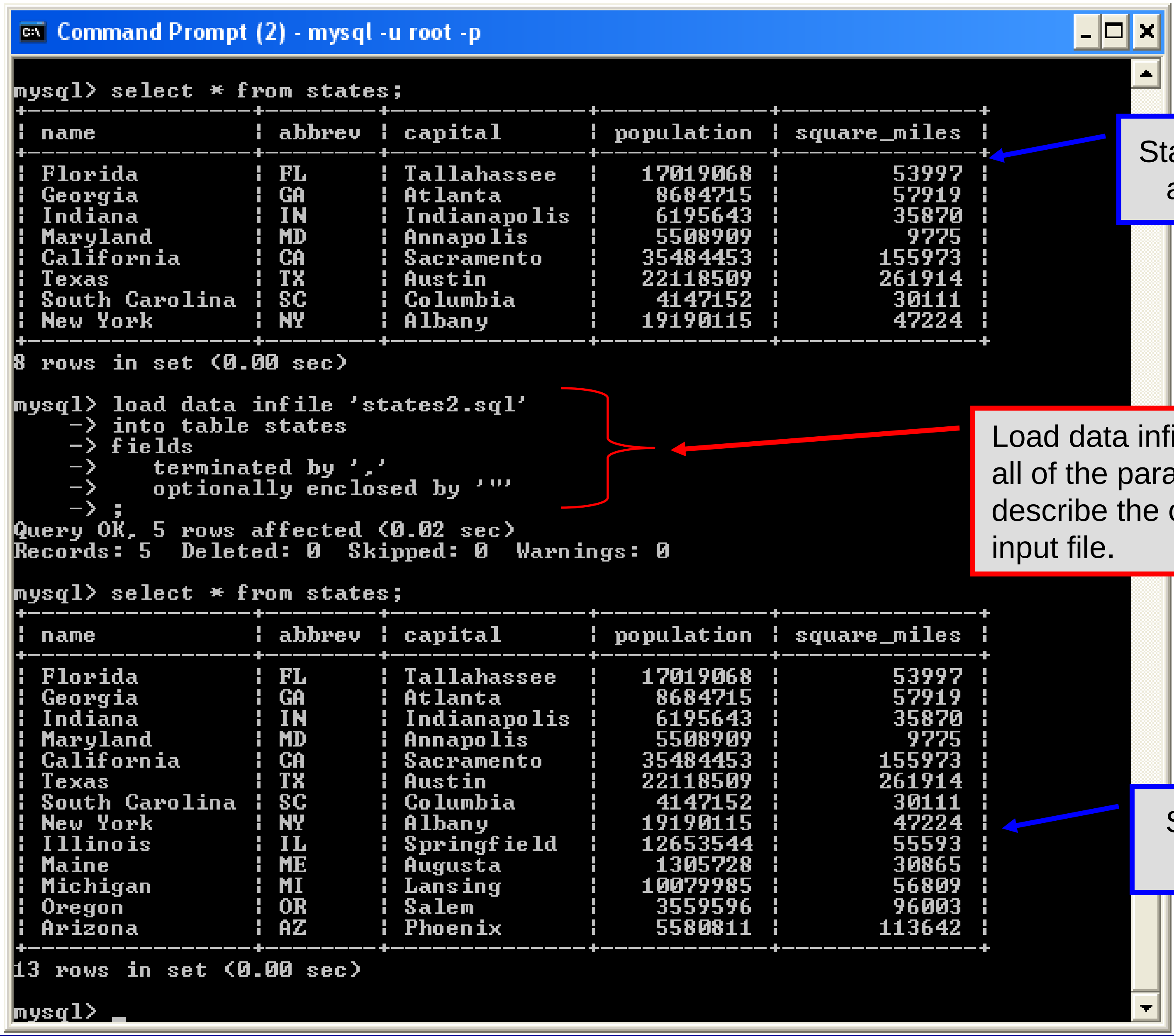


```
"Illinois","IL","Springfield",12653544,55593
"Maine","ME","Augusta",1305728,30865
"Michigan","MI","Lansing",10079985,56809
"Oregon","OR","Salem",3559596,96003
"Arizona","AZ","Phoenix",5580811,113642
```

String fields are enclosed by double quotes in this file. Numeric values are not enclosed in quotes.

Fields are delimited by commas and lines are terminated by newline characters (an invisible \n)

Text file containing the data to be loaded into the database table.



```
mysql> select * from states;
+-----------------+--------+--------------+------------+--------------+
| name            | abbrev | capital      | population | square_miles |
+-----------------+--------+--------------+------------+--------------+
| Florida         | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia         | GA     | Atlanta      |    8684715 |        57919 |
| Indiana         | IN     | Indianapolis |    6195643 |        35870 |
| Maryland        | MD     | Annapolis    |    5508909 |         9775 |
| California      | CA     | Sacramento   |   35484453 |       155973 |
| Texas           | TX     | Austin       |   22118509 |       261914 |
| South Carolina  | SC     | Columbia     |    4147152 |        30111 |
| New York        | NY     | Albany       |   19190115 |        47224 |
+-----------------+--------+--------------+------------+--------------+
8 rows in set (0.00 sec)

mysql> load data infile 'states2.sql'
    -> into table states
    -> fields
    ->     terminated by ','
    ->     optionally enclosed by '"'
    -> ;
Query OK, 5 rows affected (0.02 sec)
Records: 5  Deleted: 0  Skipped: 0  Warnings: 0

mysql> select * from states;
+-----------------+--------+--------------+------------+--------------+
| name            | abbrev | capital      | population | square_miles |
+-----------------+--------+--------------+------------+--------------+
| Florida         | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia         | GA     | Atlanta      |    8684715 |        57919 |
| Indiana         | IN     | Indianapolis |    6195643 |        35870 |
| Maryland        | MD     | Annapolis    |    5508909 |         9775 |
| California      | CA     | Sacramento   |   35484453 |       155973 |
| Texas           | TX     | Austin       |   22118509 |       261914 |
| South Carolina  | SC     | Columbia     |    4147152 |        30111 |
| New York        | NY     | Albany       |   19190115 |        47224 |
| Illinois        | IL     | Springfield  |   12653544 |        55593 |
| Maine           | ME     | Augusta      |    1305728 |        30865 |
| Michigan        | MI     | Lansing      |   10079985 |        56809 |
| Oregon          | OR     | Salem        |    3559596 |        96003 |
| Arizona         | AZ     | Phoenix      |    5580811 |       113642 |
+-----------------+--------+--------------+------------+--------------+
13 rows in set (0.00 sec)

mysql> _
```

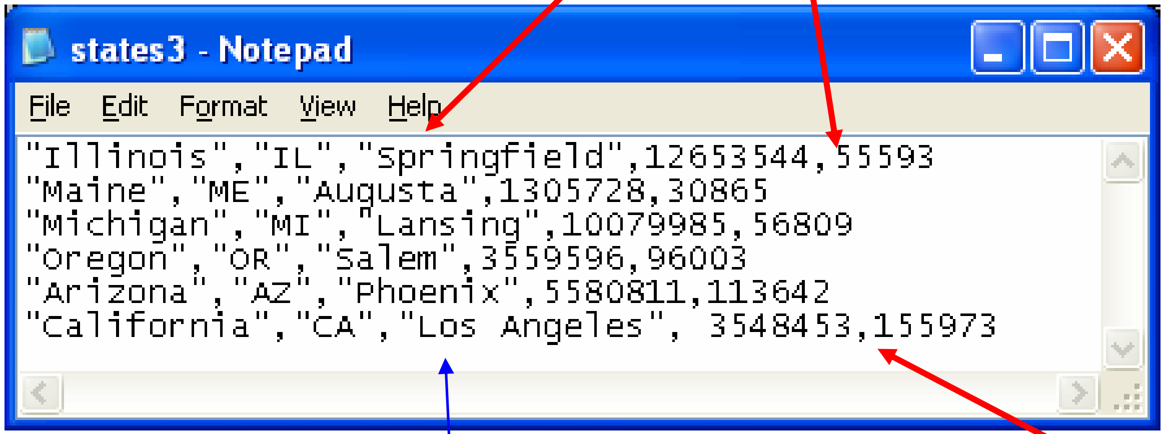
States table **before** addition of data

Load data infile statement indicating all of the parameters which describe the configuration of the input file.

States table **after** addition of data

Load Data Infile Example 2

String fields are enclosed by double quotes in this file. Numeric values are not enclosed in quotes.

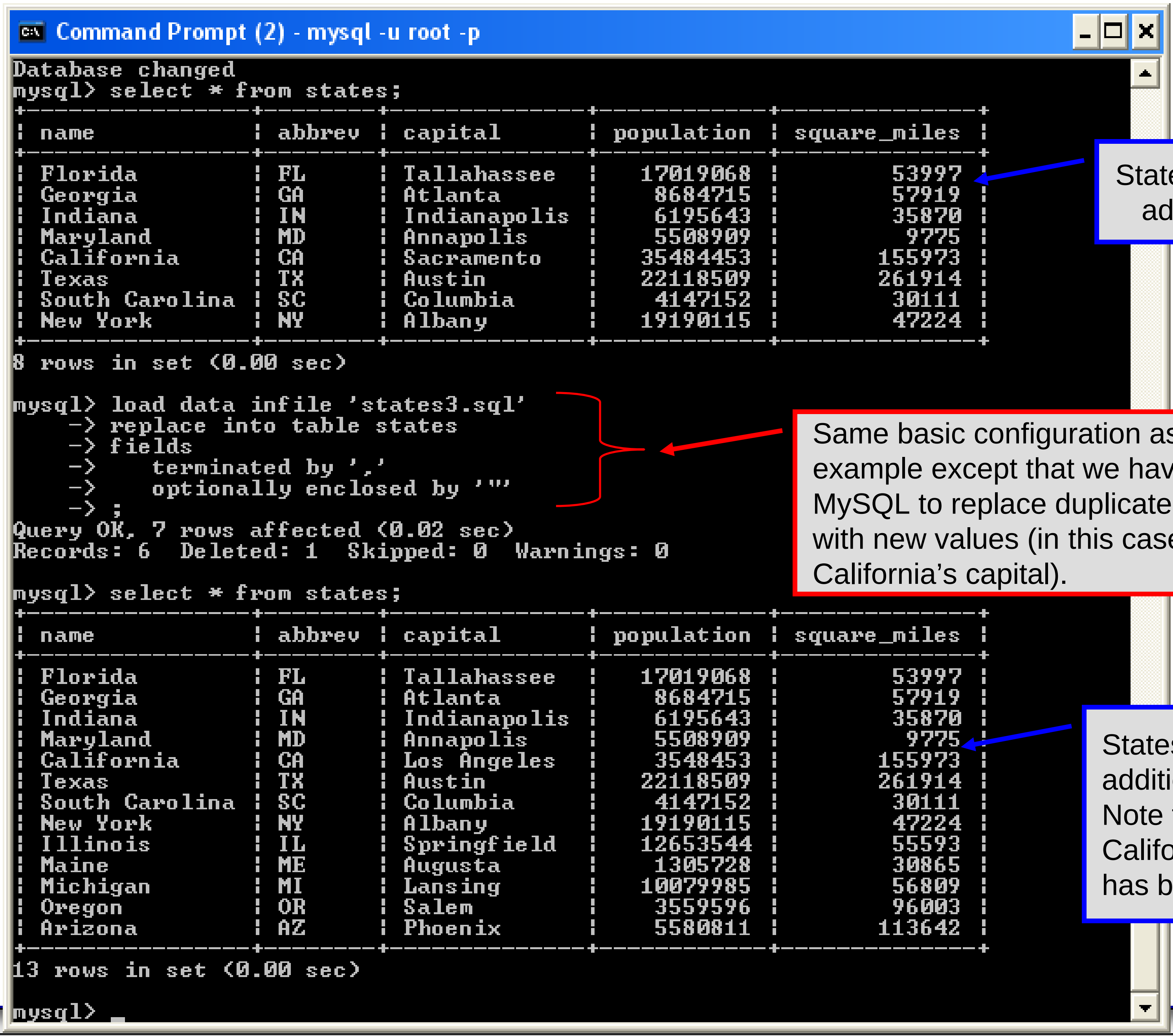


```
"Illinois","IL","Springfield",12653544,55593
"Maine","ME","Augusta",1305728,30865
"Michigan","MI","Lansing",10079985,56809
"Oregon","OR","Salem",3559596,96003
"Arizona","AZ","Phoenix",5580811,113642
"California","CA","Los Angeles", 3548453,155973
```

Text file containing the data to be loaded into the database table.

Fields are delimited by commas and lines are terminated by newline characters (an invisible \n)

California already exists in the states table – this one will replace the value of the capital with a different value.

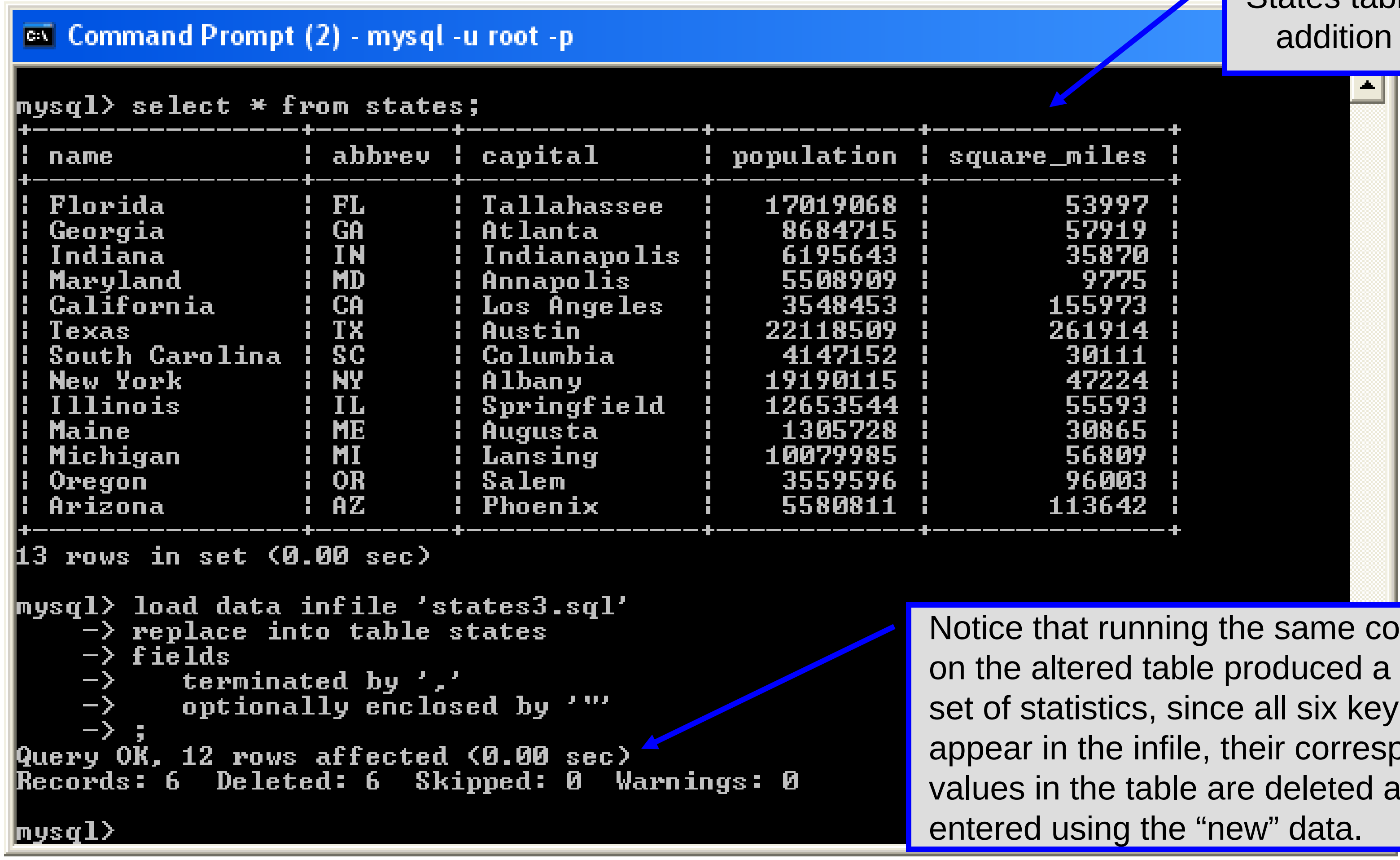


```
Database changed
mysql> select * from states;
+-----------------+--------+--------------+------------+--------------+
| name            | abbrev | capital      | population | square_miles |
+-----------------+--------+--------------+------------+--------------+
| Florida         | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia         | GA     | Atlanta      |    8684715 |        57919 |
| Indiana         | IN     | Indianapolis |    6195643 |        35870 |
| Maryland        | MD     | Annapolis    |    5508909 |         9775 |
| California      | CA     | Sacramento   |   35484453 |       155973 |
| Texas           | TX     | Austin       |   22118509 |       261914 |
| South Carolina  | SC     | Columbia     |    4147152 |        30111 |
| New York        | NY     | Albany       |   19190115 |        47224 |
+-----------------+--------+--------------+------------+--------------+
8 rows in set (0.00 sec)

mysql> load data infile 'states3.sql'
    -> replace into table states
    -> fields
    ->     terminated by ','
    ->     optionally enclosed by '"'
    -> ;
Query OK, 7 rows affected (0.02 sec)
Records: 6  Deleted: 1  Skipped: 0  Warnings: 0

mysql> select * from states;
+-----------------+--------+--------------+------------+--------------+
| name            | abbrev | capital      | population | square_miles |
+-----------------+--------+--------------+------------+--------------+
| Florida         | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia         | GA     | Atlanta      |    8684715 |        57919 |
| Indiana         | IN     | Indianapolis |    6195643 |        35870 |
| Maryland        | MD     | Annapolis    |    5508909 |         9775 |
| California      | CA     | Los Angeles  |    3548453 |       155973 |
| Texas           | TX     | Austin       |   22118509 |       261914 |
| South Carolina  | SC     | Columbia     |    4147152 |        30111 |
| New York        | NY     | Albany       |   19190115 |        47224 |
| Illinois        | IL     | Springfield  |   12653544 |        55593 |
| Maine           | ME     | Augusta      |    1305728 |        30865 |
| Michigan        | MI     | Lansing      |   10079985 |        56809 |
| Oregon          | OR     | Salem        |    3559596 |        96003 |
| Arizona         | AZ     | Phoenix      |    5580811 |       113642 |
+-----------------+--------+--------------+------------+--------------+
13 rows in set (0.00 sec)

mysql> _
```

States table **before** addition of data

Same basic configuration as in previous example except that we have instructed MySQL to replace duplicate key value rows with new values (in this case replacing California's capital).

States table **after** addition of data. Note that California's capital has been changed!



Command Prompt (2) - mysql -u root -p

```
mysql> select * from states;
+----------------+--------+--------------+------------+--------------+
| name           | abbrev | capital      | population | square_miles |
+----------------+--------+--------------+------------+--------------+
| Florida        | FL     | Tallahassee  |   17019068 |        53997 |
| Georgia        | GA     | Atlanta      |    8684715 |        57919 |
| Indiana        | IN     | Indianapolis |    6195643 |        35870 |
| Maryland       | MD     | Annapolis    |    5508909 |         9775 |
| California     | CA     | Los Angeles  |    3548453 |       155973 |
| Texas          | TX     | Austin       |   22118509 |       261914 |
| South Carolina | SC     | Columbia     |    4147152 |        30111 |
| New York       | NY     | Albany       |   19190115 |        47224 |
| Illinois       | IL     | Springfield  |   12653544 |        55593 |
| Maine          | ME     | Augusta      |    1305728 |        30865 |
| Michigan       | MI     | Lansing      |   10079985 |        56809 |
| Oregon         | OR     | Salem        |    3559596 |        96003 |
| Arizona        | AZ     | Phoenix      |    5580811 |       113642 |
+----------------+--------+--------------+------------+--------------+
13 rows in set (0.00 sec)

mysql> load data infile 'states3.sql'
    -> replace into table states
    -> fields
    ->    terminated by ','
    ->    optionally enclosed by '"'
    -> ;
Query OK, 12 rows affected (0.00 sec)
Records: 6  Deleted: 6  Skipped: 0  Warnings: 0

mysql>
```

States table **before** addition of data

Notice that running the same command on the altered table produced a different set of statistics, since all six key values appear in the infile, their corresponding values in the table are deleted and re-entered using the "new" data.

The `Ignore` Clause of the Insert Command

- While the normal issues of data type compatibility are always of concern, there are other issues to deal with when inserting data into tables.
- There is the possibility that a duplicate of a key may be entered. If so, you will see an error like this:

```
ERROR 1062: Duplicate entry ‘2’ for key 1
```

- It is possible to subdue errors by using the keyword `ignore` in the `insert` statement. By using `ignore` any duplicate rows will simply be ignored. They won’t be imported, and the data at the related row of the target table will be left untouched.
  - In your application, you would be wise to check how many rows were affected (imported) whenever using `ignore` because ignoring a record may constitute a failure condition in your application that needs to be handled.

Low Priority and Delayed Inserts

- If you specify `insert low-priority`, the insert waits until all other clients have finished reading from the table before the insert is executed.
- If you specify `insert delayed`, the client performing the action gets and instant acknowledgement that the insert has been performed, although in fact the data will only be inserted when the table is not in use by another thread.
  - This may be useful if you have an application that needs to complete its process in minimum time, or simply where there is no need for it to wait for the effect of an insert to take place. For example, when you're adding data to a log or audit trail.
  - This feature applies only to ISAM or MyISAM type files.

Inserting/Replacing Data Using `replace`

- Data can also be entered into a MySQL table using the `replace` command.
- The `replace` statement has forms similar to the `insert` statement:

Form 1

```
replace [low priority | delayed] [ignore] [into]table_name
          [set] column_name1 = expression1,
                   column_name2 = expression2, …
```

Form 2

```
replace [low priority | delayed] [ignore] [into]table_name
          [(column_name,…)]values (expression,…), (…)…
```

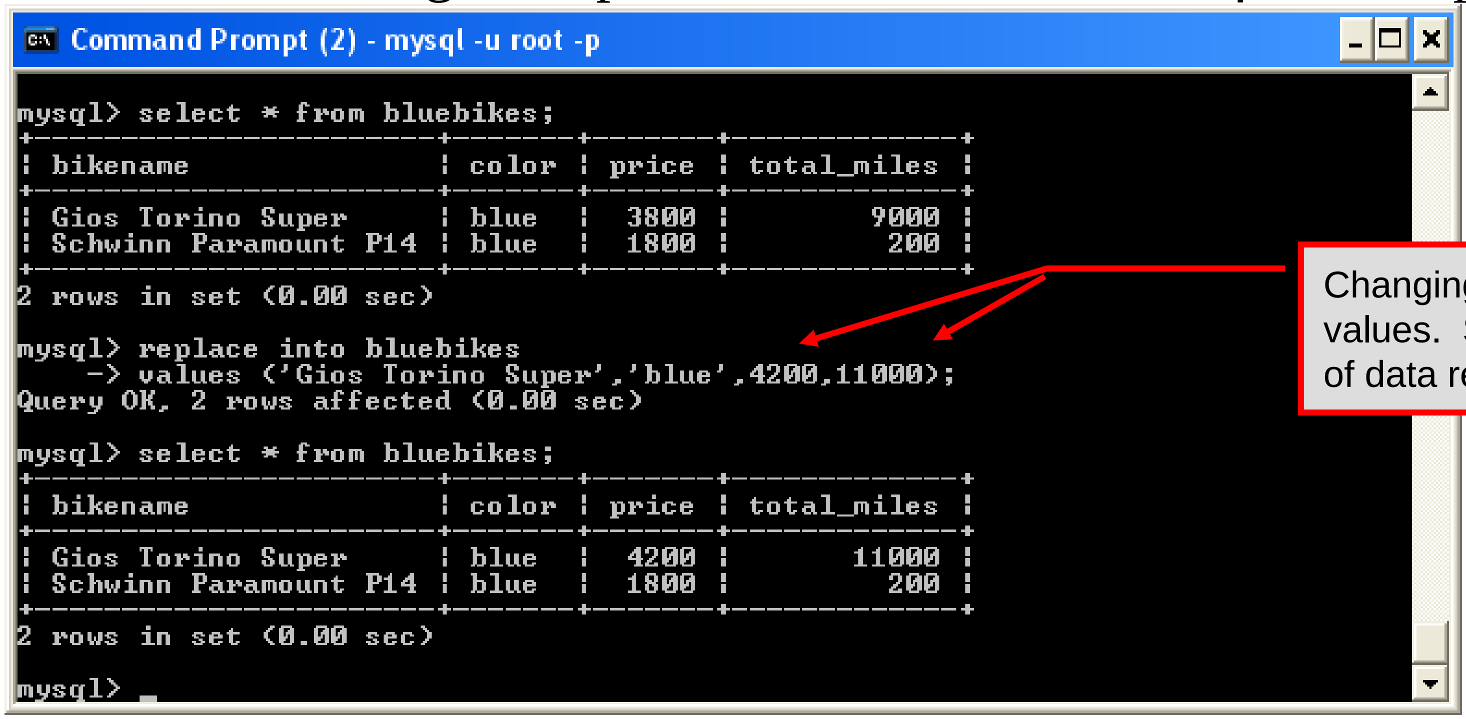
Form 3

```
replace [low priority | delayed] [ignore] [into]table_name
          [(column_name,…)] select…
```

Using `replace`

- The `replace` statement works similar to `insert`. It always tries to insert the new data, but when it tries to insert a new row with the same primary or unique key as an existing row, it deletes the old row and replaces it with the new values.

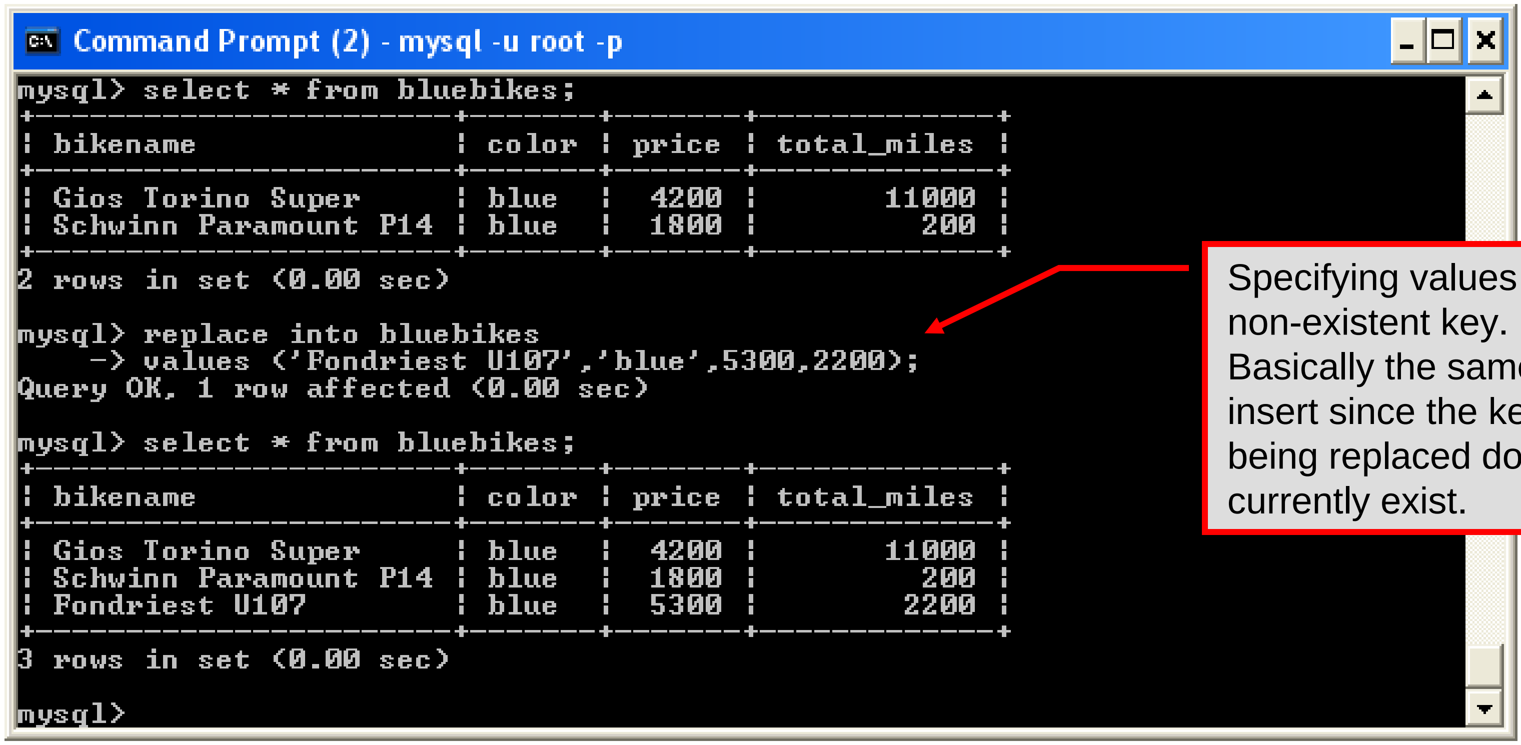
- The following examples will illustrate how `replace` operates.



```
mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  3800 |        9000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
+------------------------+-------+-------+-------------+
2 rows in set (0.00 sec)

mysql> replace into bluebikes
    -> values ('Gios Torino Super','blue',4200,11000);
Query OK, 2 rows affected (0.00 sec)

mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  4200 |       11000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
+------------------------+-------+-------+-------------+
2 rows in set (0.00 sec)

mysql> _
```

Changing non-key values. Simplest form of data replacement.

Using Replace (cont.)



Command Prompt (2) - mysql -u root -p

```
mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  4200 |       11000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
+------------------------+-------+-------+-------------+
2 rows in set (0.00 sec)

mysql> replace into bluebikes
    -> values ('Fondriest U107','blue',5300,2200);
Query OK, 1 row affected (0.00 sec)

mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  4200 |       11000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
| Fondriest U107         | blue  |  5300 |        2200 |
+------------------------+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql>
```

Specifying values for a non-existent key. Basically the same as an insert since the key value being replaced does not currently exist.

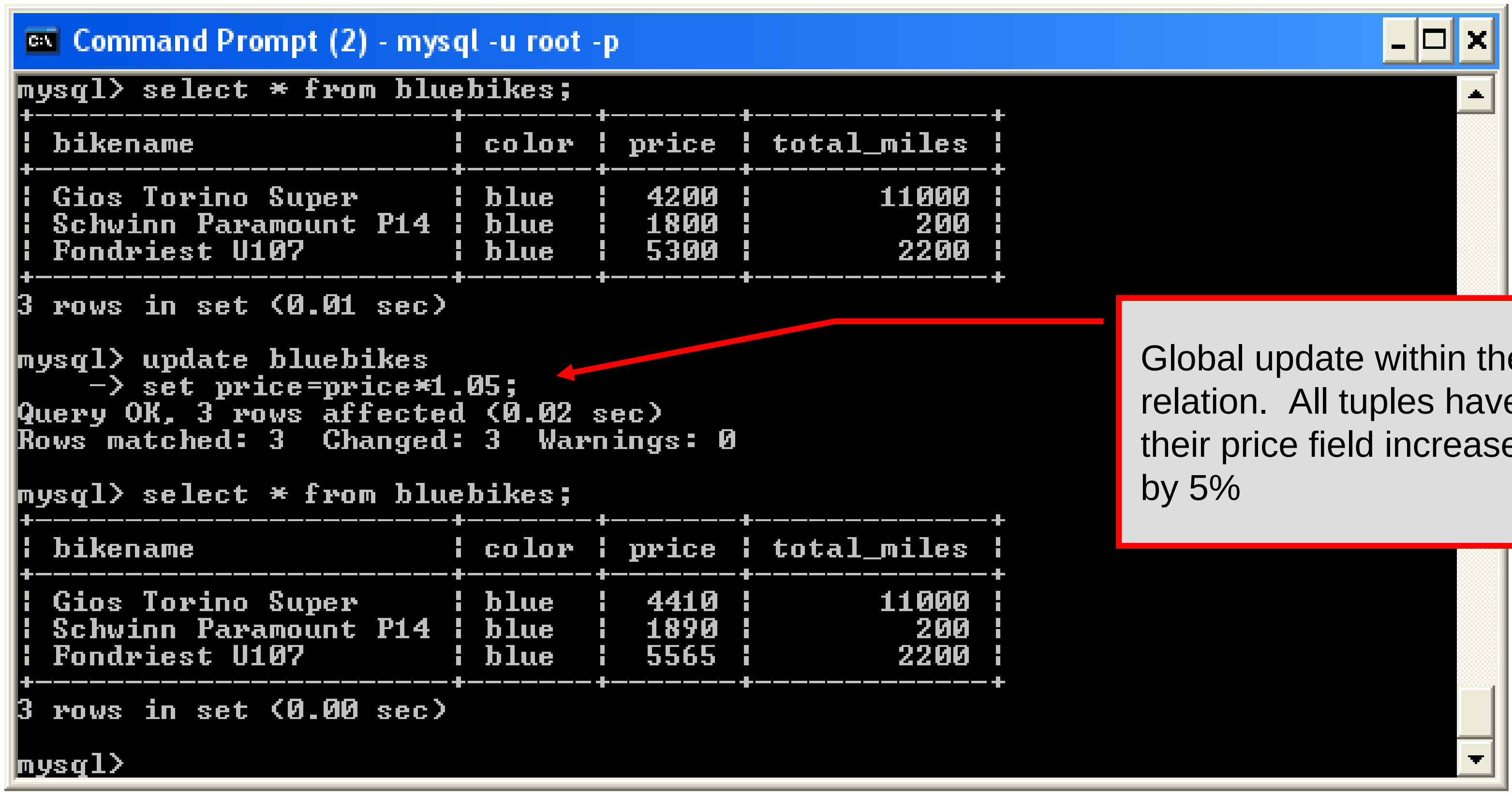
Performing Updates on Tables

- The `update` command allows you to modify the values of the existing data in a table. The basic format of the statement is:

```
update [low priority] [ignore] table_name
    set column_name1 = expression1,
        column_name2 = expression2, …
    [where where_definition]
    [limit num];
```

- There are basically two parts to the statement: the `set` portion to declare which column to set to what value; and the `where` portion, which defines which rows are to be affected.
- `Limit` restricts the number of rows affected to `num`.

Using `update` (cont.)

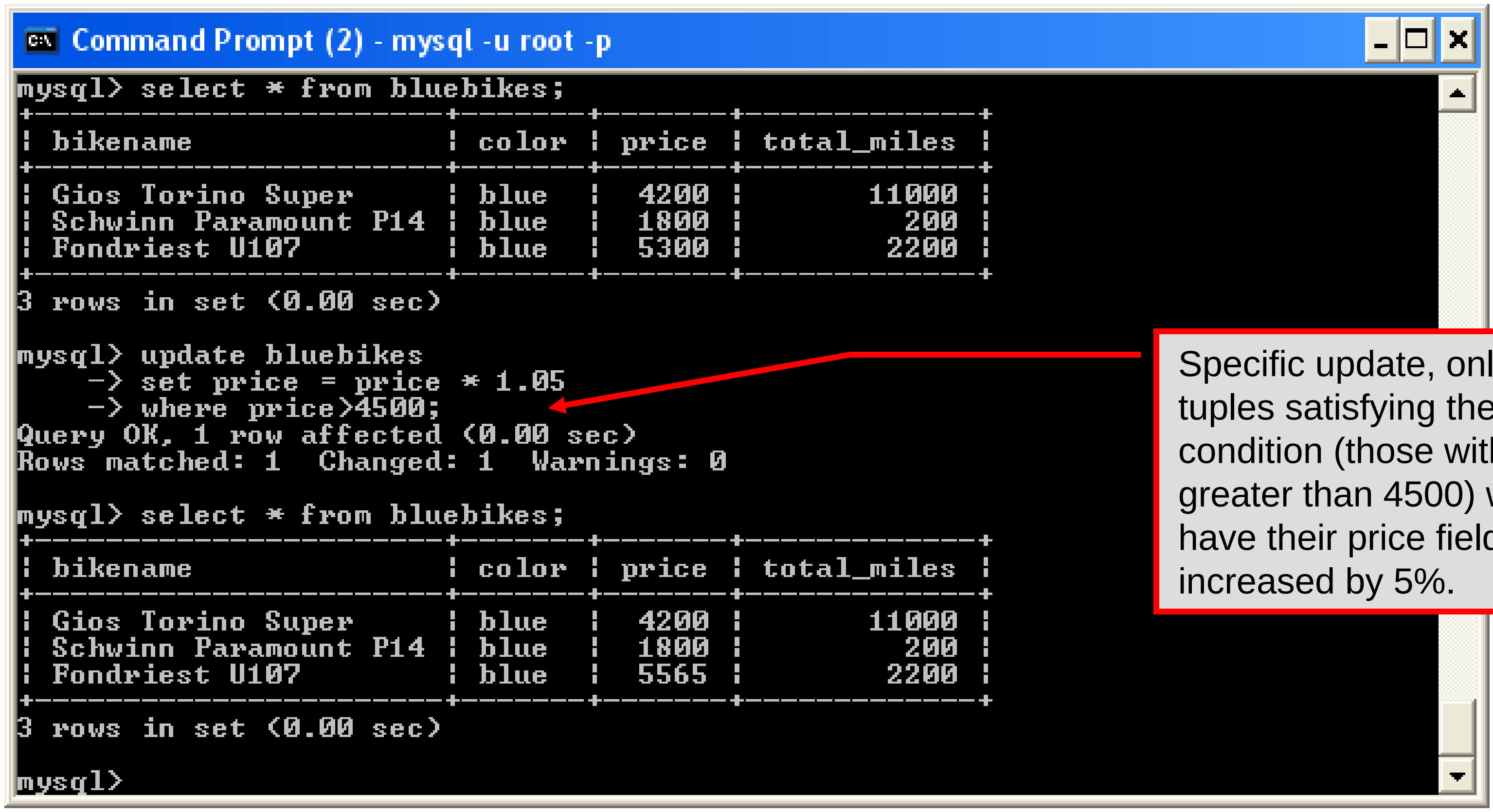


Command Prompt (2) - mysql -u root -p

```
mysql> select * from bluebikes;
+-------------------------+-------+-------+-------------+
| bikename                | color | price | total_miles |
+-------------------------+-------+-------+-------------+
| Gios Torino Super       | blue  |  4200 |       11000 |
| Schwinn Paramount P14   | blue  |  1800 |         200 |
| Fondriest U107          | blue  |  5300 |        2200 |
+-------------------------+-------+-------+-------------+
3 rows in set (0.01 sec)

mysql> update bluebikes
    -> set price=price*1.05;
Query OK, 3 rows affected (0.02 sec)
Rows matched: 3  Changed: 3  Warnings: 0

mysql> select * from bluebikes;
+-------------------------+-------+-------+-------------+
| bikename                | color | price | total_miles |
+-------------------------+-------+-------+-------------+
| Gios Torino Super       | blue  |  4410 |       11000 |
| Schwinn Paramount P14   | blue  |  1890 |         200 |
| Fondriest U107          | blue  |  5565 |        2200 |
+-------------------------+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql>
```

Global update within the relation. All tuples have their price field increased by 5%

Using `update` (cont.)



```
mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  4200 |       11000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
| Fondriest U107         | blue  |  5300 |        2200 |
+------------------------+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql> update bluebikes
    -> set price = price * 1.05
    -> where price>4500;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from bluebikes;
+------------------------+-------+-------+-------------+
| bikename               | color | price | total_miles |
+------------------------+-------+-------+-------------+
| Gios Torino Super      | blue  |  4200 |       11000 |
| Schwinn Paramount P14  | blue  |  1800 |         200 |
| Fondriest U107         | blue  |  5565 |        2200 |
+------------------------+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql>
```

Specific update, only tuples satisfying the select condition (those with price greater than 4500) will have their price field increased by 5%.

Select Queries in MySQL

- The `select` command in MySQL is basically the same as in the standard SQL, however, it does have some additional features. The basic format of the statement is (not all options are shown – for complete details see the SQL Manual):

```
SELECT [ALL | DISTINCT | DISTINCTROW][HIGH_PRIORITY]
       [STRAIGHT JOIN] [SQL_SMALL_RESULT][SQL_BIG_RESULT]
       [SQL_BUFFER_RESULT][SQ_CACHE | SQL_NO_CACHE]
       select_expression, …
   [INTO {OUTFILE | DUMPFILE} 'path/to/filename' export_options]
   [FROM table_references
        WHERE where_definition]
        [GROUP BY {col_name | col_alias | col_pos | formula}
                [asc |desc], …]
        [HAVING where_definition]
        [ORDER BY {col_name | col_alias | col_pos | formula}
                [asc | desc], …]
        [LIMIT [offset, ] num_rows]
        [PROCEDURE procedure_name];
```

MySQL RDBMS (cont.)

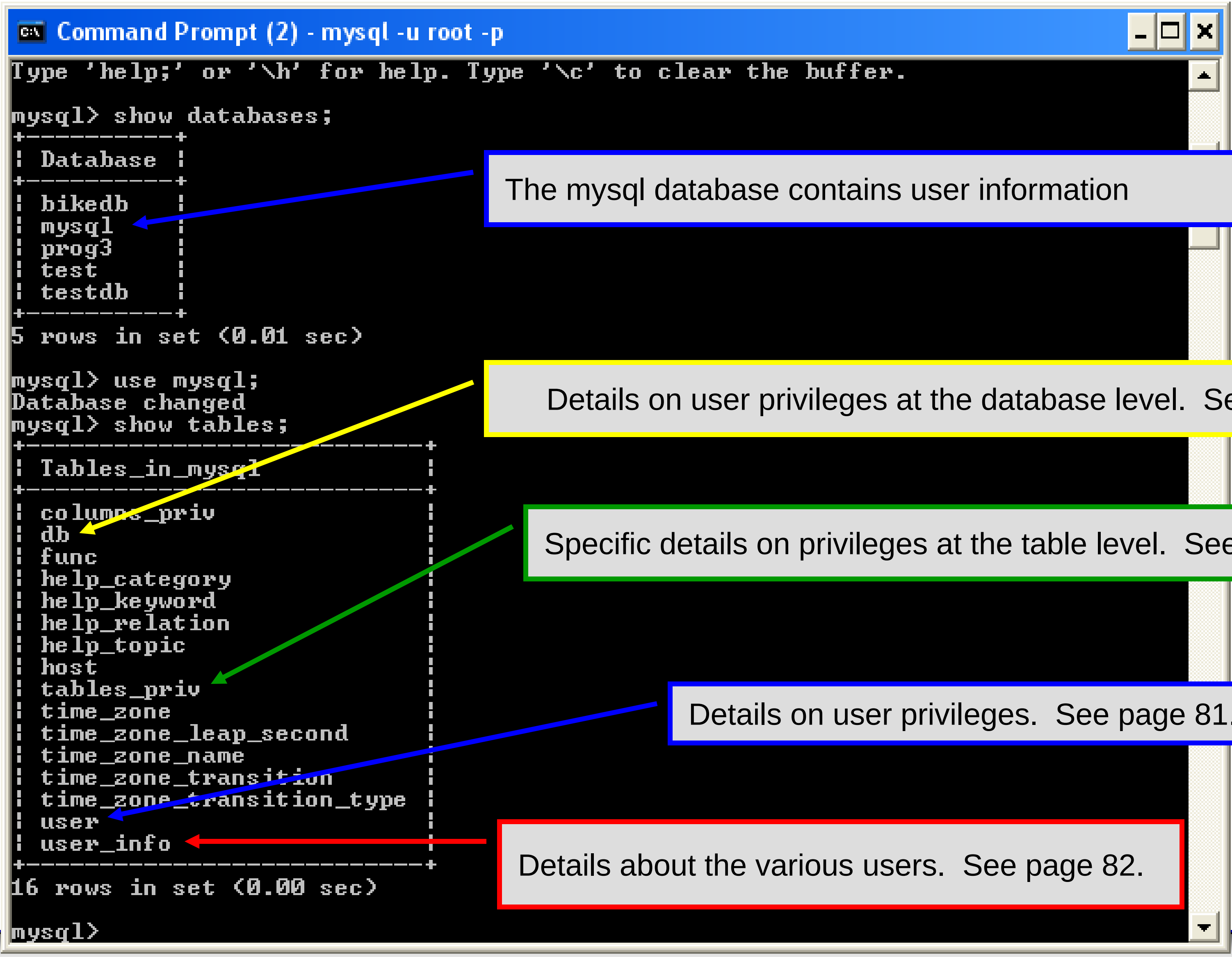
- MySQL features a user permissions system, which allows control over user's access to the databases under MySQL control.
- There are very few competitors of MySQL (Oracle, Sybase, DB2, and SQL Server) that can match the level of sophistication provided by MySQL's permissions system in terms of granularity and level of security provided.

Note that I did not include Microsoft Access in the list above. There are a couple of reasons for this; Access concentrates on the client front-end, although available in shareable versions, it lacks the management system that is a key part of any RDBMS. Access provides virtually no user authentication capabilities nor does it have multithreading processing capabilities, in its normal form.

Authorization in MySQL

- `mysql` and the various utility programs such as `mysqladmin`, `mysqlshow`, and `mysqlimport` can only be invoked by a valid MySQL user.
- Permissions for various users are recorded in grant tables maintained by MySQL.
- As the root user, you have access to all the databases and tables maintained by the MySQL Server.
- One of these databases is named `mysql`.and contains the various information on the users who have access to this installation of MySQL. Some of the tables which comprise this database are shown on the next few pages.

Tables in the `mysql` Database



```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+----------+
| Database |
+----------+
| bikedb   |
| mysql    |
| prog3    |
| test     |
| testdb   |
+----------+
5 rows in set (0.01 sec)

mysql> use mysql;
Database changed
mysql> show tables;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| columns_priv              |
| db                        |
| func                      |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| host                      |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
| user_info                 |
+---------------------------+
16 rows in set (0.00 sec)

mysql>
```

The mysql database contains user information

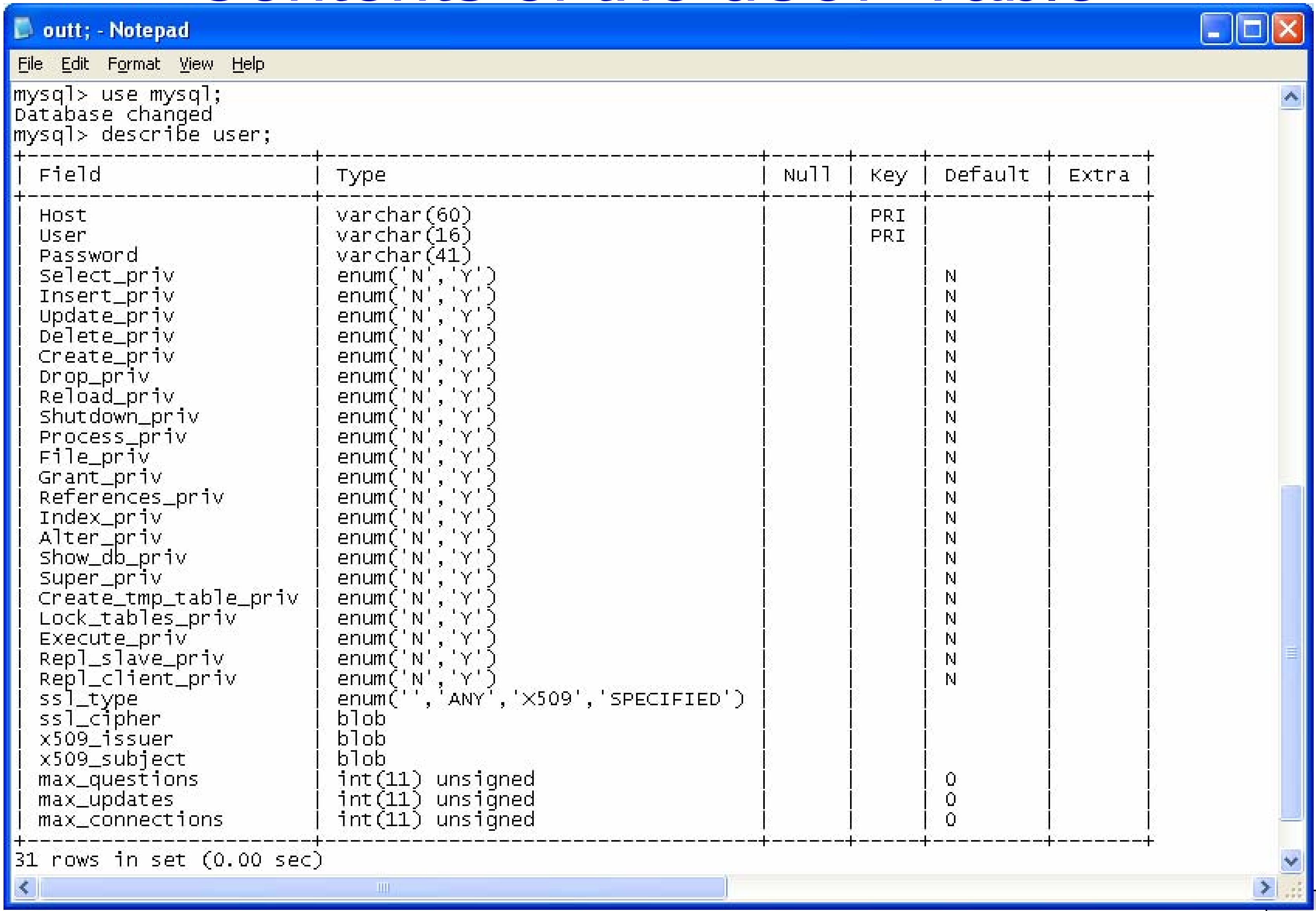
Details on user privileges at the database level. See page 84.

Specific details on privileges at the table level. See page 83

Details on user privileges. See page 81.

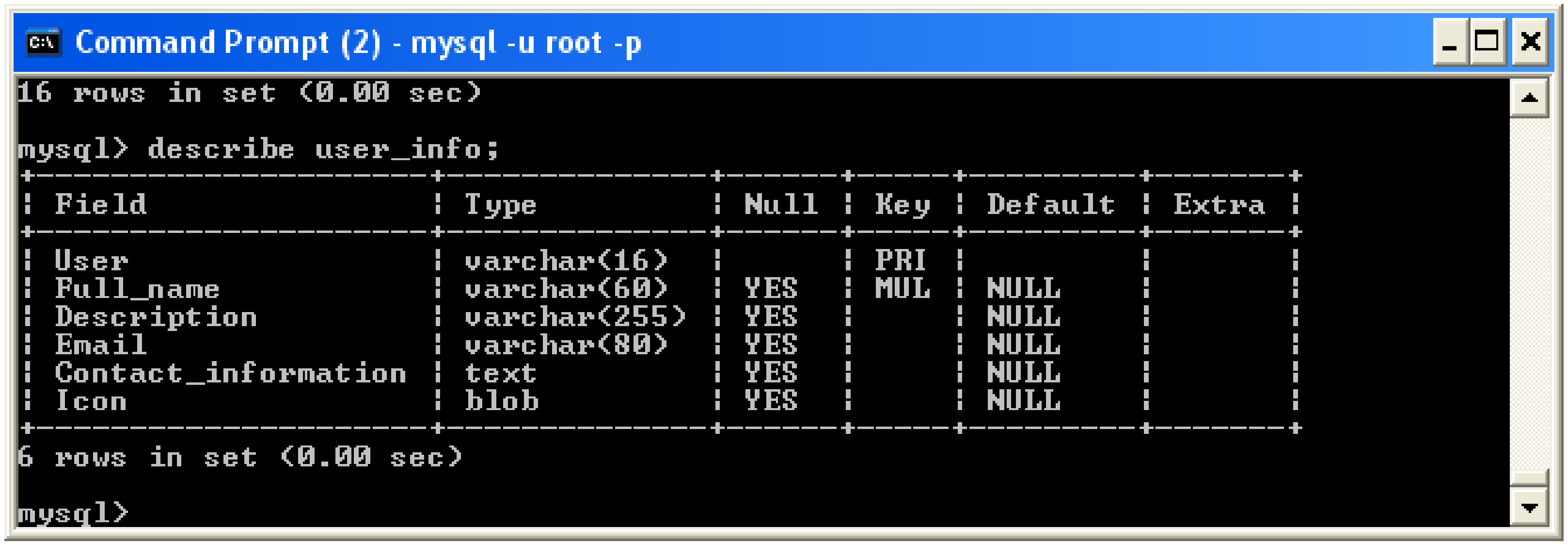
Details about the various users. See page 82.

Contents of the user Table



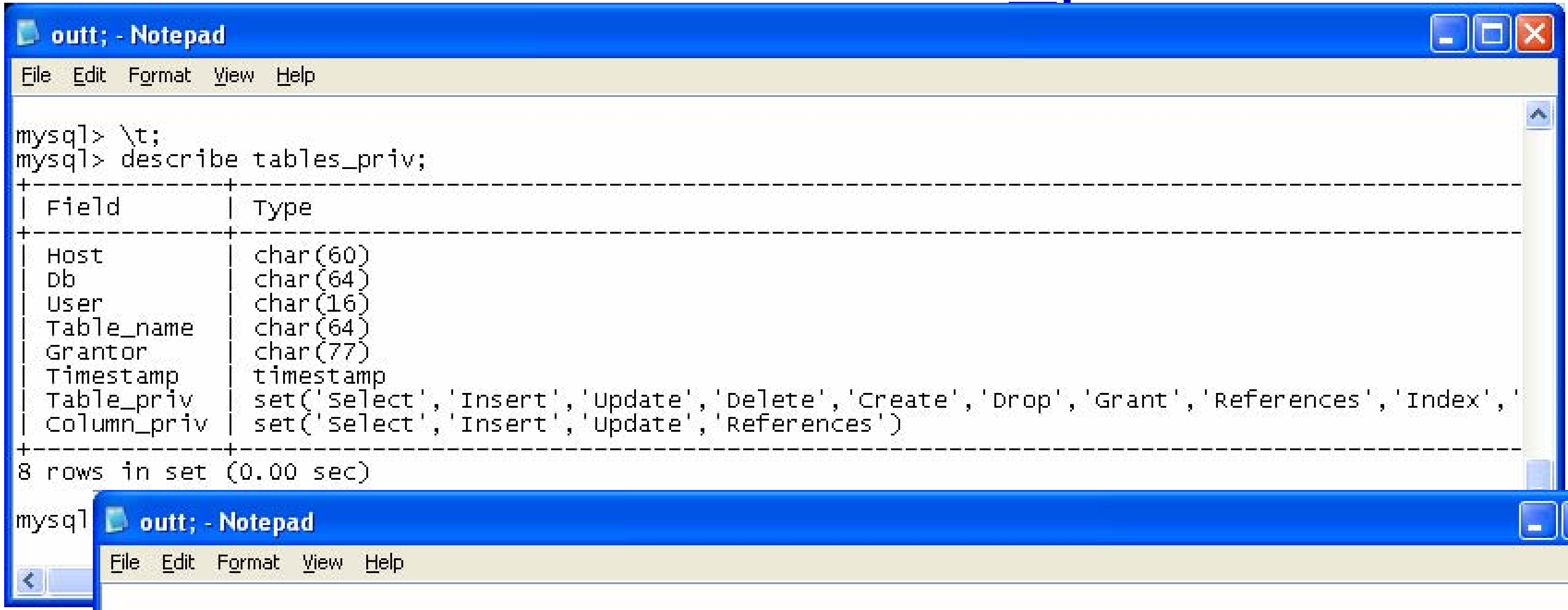
```
mysql> use mysql;
Database changed
mysql> describe user;
+-----------------------+-----------------------------------+------+-----+---------+-------+
| Field                 | Type                              | Null | Key | Default | Extra |
+-----------------------+-----------------------------------+------+-----+---------+-------+
| Host                  | varchar(60)                       |      | PRI |         |       |
| User                  | varchar(16)                       |      | PRI |         |       |
| Password              | varchar(41)                       |      |     |         |       |
| Select_priv           | enum('N','Y')                     |      |     | N       |       |
| Insert_priv           | enum('N','Y')                     |      |     | N       |       |
| Update_priv           | enum('N','Y')                     |      |     | N       |       |
| Delete_priv           | enum('N','Y')                     |      |     | N       |       |
| Create_priv           | enum('N','Y')                     |      |     | N       |       |
| Drop_priv             | enum('N','Y')                     |      |     | N       |       |
| Reload_priv           | enum('N','Y')                     |      |     | N       |       |
| Shutdown_priv         | enum('N','Y')                     |      |     | N       |       |
| Process_priv          | enum('N','Y')                     |      |     | N       |       |
| File_priv             | enum('N','Y')                     |      |     | N       |       |
| Grant_priv            | enum('N','Y')                     |      |     | N       |       |
| References_priv       | enum('N','Y')                     |      |     | N       |       |
| Index_priv            | enum('N','Y')                     |      |     | N       |       |
| Alter_priv            | enum('N','Y')                     |      |     | N       |       |
| Show_db_priv          | enum('N','Y')                     |      |     | N       |       |
| Super_priv            | enum('N','Y')                     |      |     | N       |       |
| Create_tmp_table_priv | enum('N','Y')                     |      |     | N       |       |
| Lock_tables_priv      | enum('N','Y')                     |      |     | N       |       |
| Execute_priv          | enum('N','Y')                     |      |     | N       |       |
| Repl_slave_priv       | enum('N','Y')                     |      |     | N       |       |
| Repl_client_priv      | enum('N','Y')                     |      |     | N       |       |
| ssl_type              | enum('','ANY','X509','SPECIFIED') |      |     |         |       |
| ssl_cipher            | blob                              |      |     |         |       |
| x509_issuer           | blob                              |      |     |         |       |
| x509_subject          | blob                              |      |     |         |       |
| max_questions         | int(11) unsigned                  |      |     | 0       |       |
| max_updates           | int(11) unsigned                  |      |     | 0       |       |
| max_connections       | int(11) unsigned                  |      |     | 0       |       |
+-----------------------+-----------------------------------+------+-----+---------+-------+
31 rows in set (0.00 sec)
```

Contents of the `user_info` Table

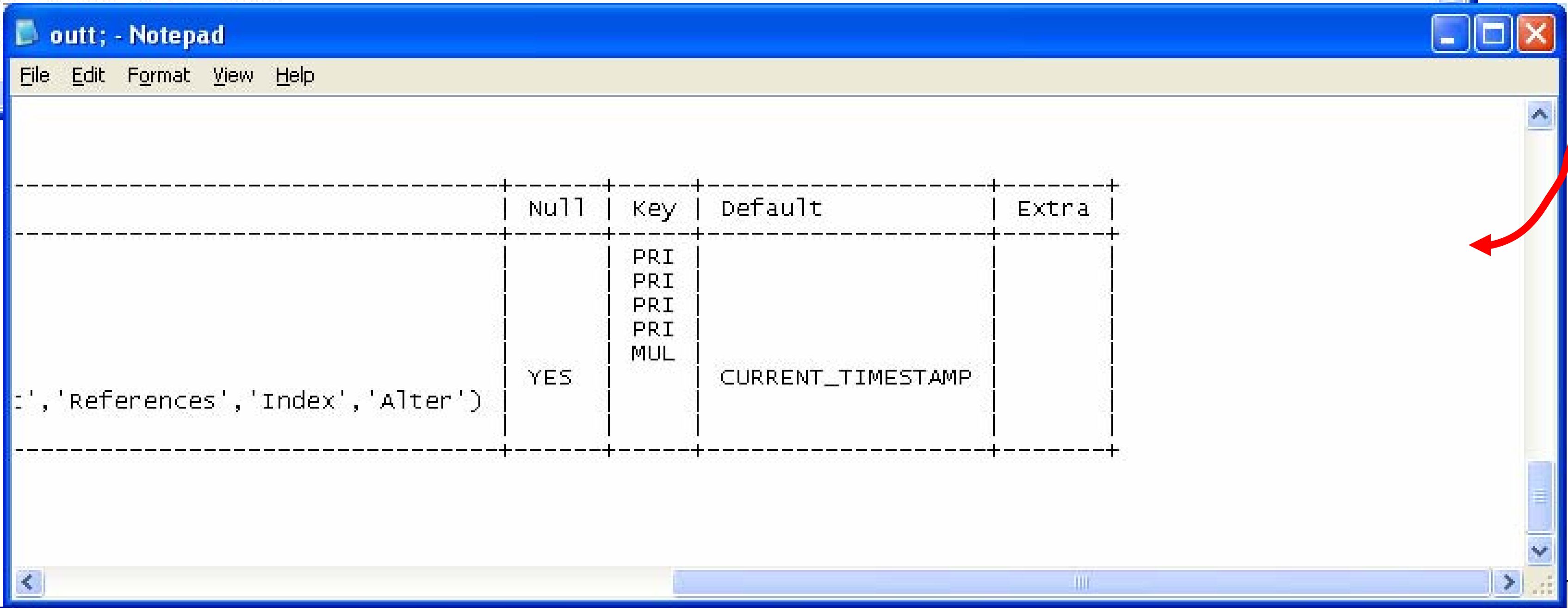

Command Prompt (2) - mysql -u root -p

```
16 rows in set (0.00 sec)

mysql> describe user_info;
+---------------------+--------------+------+-----+---------+-------+
| Field               | Type         | Null | Key | Default | Extra |
+---------------------+--------------+------+-----+---------+-------+
| User                | varchar(16)  |      | PRI |         |       |
| Full_name           | varchar(60)  | YES  | MUL | NULL    |       |
| Description         | varchar(255) | YES  |     | NULL    |       |
| Email               | varchar(80)  | YES  |     | NULL    |       |
| Contact_information | text         | YES  |     | NULL    |       |
| Icon                | blob         | YES  |     | NULL    |       |
+---------------------+--------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

Contents of the `tables_priv` Table

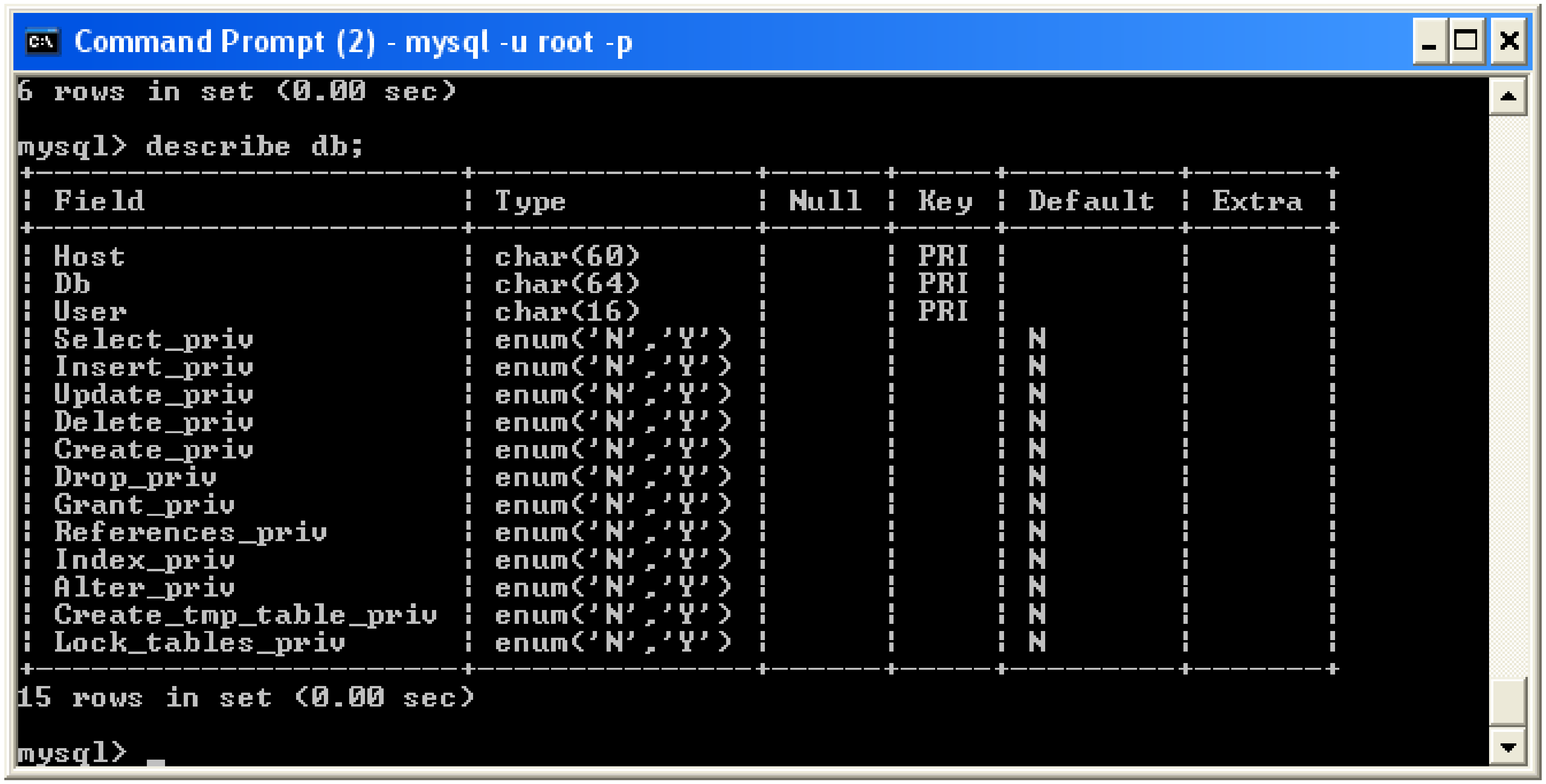


```
mysql> \t;
mysql> describe tables_priv;
+-------------+--------------------------------------------------------------------------------------------------
| Field       | Type
+-------------+--------------------------------------------------------------------------------------------------
| Host        | char(60)
| Db          | char(64)
| User        | char(16)
| Table_name  | char(64)
| Grantor     | char(77)
| Timestamp   | timestamp
| Table_priv  | set('Select','Insert','Update','Delete','Create','Drop','Grant','References','Index','
| Column_priv | set('Select','Insert','Update','References')
+-------------+--------------------------------------------------------------------------------------------------
8 rows in set (0.00 sec)

mysql
```



```
---------------------------------+------+-----+-------------------+-------+
                                 | Null | Key | Default           | Extra |
---------------------------------+------+-----+-------------------+-------+
                                 |      | PRI |                   |       |
                                 |      | PRI |                   |       |
                                 |      | PRI |                   |       |
                                 |      | PRI |                   |       |
                                 |      | MUL |                   |       |
                                 | YES  |     | CURRENT_TIMESTAMP |       |
:','References','Index','Alter') |      |     |                   |       |
                                 |      |     |                   |       |
---------------------------------+------+-----+-------------------+-------+
```

Contents of the db Table



Command Prompt (2) - mysql -u root -p

```
6 rows in set (0.00 sec)

mysql> describe db;
+-----------------------+-----------------+------+-----+---------+-------+
| Field                 | Type            | Null | Key | Default | Extra |
+-----------------------+-----------------+------+-----+---------+-------+
| Host                  | char(60)        |      | PRI |         |       |
| Db                    | char(64)        |      | PRI |         |       |
| User                  | char(16)        |      | PRI |         |       |
| Select_priv           | enum('N','Y')   |      |     | N       |       |
| Insert_priv           | enum('N','Y')   |      |     | N       |       |
| Update_priv           | enum('N','Y')   |      |     | N       |       |
| Delete_priv           | enum('N','Y')   |      |     | N       |       |
| Create_priv           | enum('N','Y')   |      |     | N       |       |
| Drop_priv             | enum('N','Y')   |      |     | N       |       |
| Grant_priv            | enum('N','Y')   |      |     | N       |       |
| References_priv       | enum('N','Y')   |      |     | N       |       |
| Index_priv            | enum('N','Y')   |      |     | N       |       |
| Alter_priv            | enum('N','Y')   |      |     | N       |       |
| Create_tmp_table_priv | enum('N','Y')   |      |     | N       |       |
| Lock_tables_priv      | enum('N','Y')   |      |     | N       |       |
+-----------------------+-----------------+------+-----+---------+-------+
15 rows in set (0.00 sec)

mysql> _
```

How The Grant Tables Work

- The various grant tables work together to define access capabilities for the various users of the databases in MySQL. The tables represent a hierarchy which begins at the database level and moves downward to finer and finer granularity in access capabilities.
- To understand how the grant tables work, it is necessary to understand the process that MySQL goes through when considering a request from a client.

Step 1: A user attempts to connect to the MySQL server. The `user` table is consulted, and on the basis of the username, password, and host from which the connection is occurring, the connection is either refused or accepted. (MySQL actually sorts the user table and looks for the first match.)

How The Grant Tables Work (cont.)

Step 2: If the connection is accepted, any privilege fields in the `user` table that are set to 'Y' will allow the user to perform that action on any database under the server's control. For administrative actions such as shutdown and reload, the entry in the `user` table is deemed absolute, and no further grant tables are consulted.

Step 3: Where the user makes a database-related request and the `user` table does not allow the user to perform that operations (the privilege is set to 'N'), MySQL consults the `db` table (see page 84).

Step 4: The `db` table is consulted to see if there is an entry for the user, database, and host. If there is a match, the `db` privilege fields determine whether the user can perform the request.

How The Grant Tables Work (cont.)

Step 5: If there is a match on the `db` table's `Db` and `User` files but `Host` is blank, the `host` table is consulted to see whether there is a match on all three fields. If there is, the privilege fields in the `host` table will determine whether the use can perform the requested operation. Corresponding entries in the `db` and `host` tables must both be 'Y' for the request to be granted. Thus, an 'N' in either table will block the request.

Step 6: If the user's request is not granted, MySQL checks the `tables_priv` (see page 83) and `columns_priv` tables. It looks for a match on the user, host, database, and table to which the request is made (and the column, if there is an entry in the `columns_priv` table). It adds any privileges it finds in these tables to the privileges already granted. The sum of these privileges determines if the request can be granted.

Managing User Privileges with GRANT and REVOKE

- The basic granting and revocation of privileges in MySQL are accomplished through the `grant` and `revoke` commands.
- The format of the `grant` command is:

```
GRANT privileges [(column_list)]
 ON  database_name.table_name
 TO username@hostname [IDENTIFIED BY ‘password’]
 [REQUIRE [SSL | X509]
    [CIPHER  cipher [AND] ]
    [ISSUER  issuer [AND] ]
    [SUBJECT subject ] ]
 [WITH GRANT OPTION |
     MAX_QUERIES_PER_HOUR  num  |
     MAX_UPDATES_PER_HOUR  num  |
     MAX_CONNECTIONS_PER_HOUR  num ]
```

Some of the Privileges Assigned with GRANT

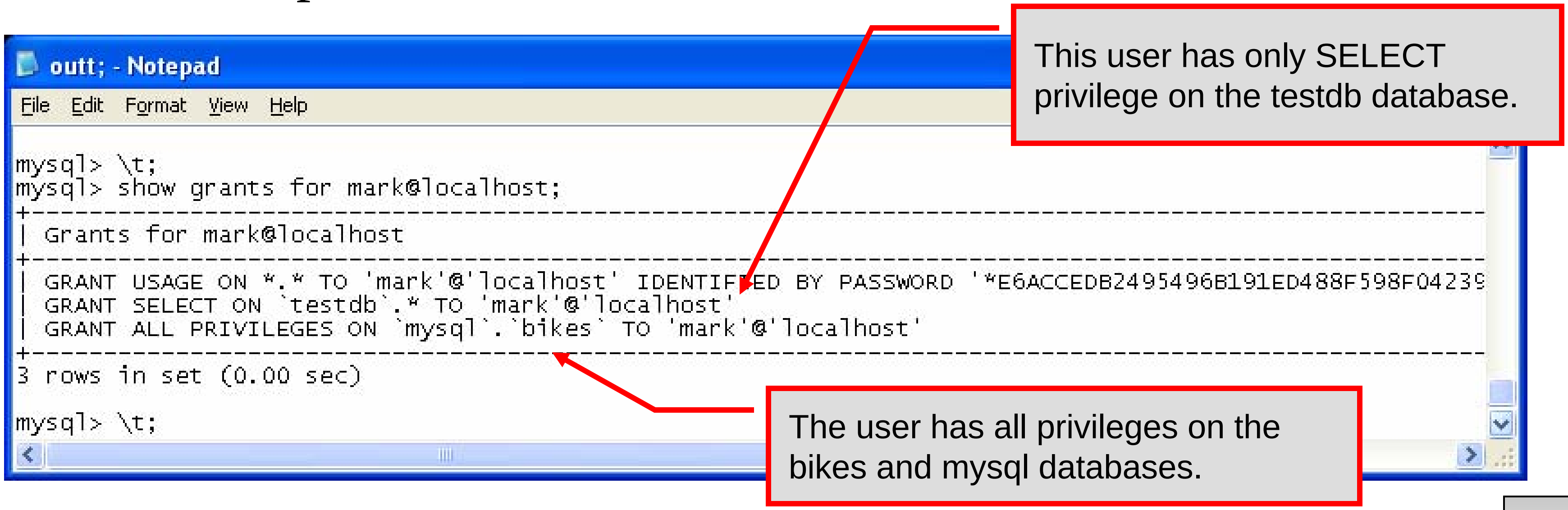
Privilege	Operations Permitted
ALL or ALL PRIVILEGES	All privileges except for GRANT
ALTER	Change a table definition using ALTER TABLE excluding the creation and dropping of indices.
CREATE	Create database or tables within a database.
CREATE TEMPORARY TABLES	Create temporary tables.
DELETE	Ability to perform deletions from tables. (Delete DML statements).
DROP	Ability to drop databases or tables.
INSERT	Ability to insert data into tables.
SHUTDOWN	Ability to shutdown the MySQL server.

Displaying Privileges with SHOW

- The SQL command `SHOW` is used to display the grant privileges for a given user.
- The syntax for the `SHOW` command is:

```
SHOW GRANTS FOR username@hostname
```

- An example is shown below:



```
mysql> \t;
mysql> show grants for mark@localhost;
+------------------------------------------------------------------------------------------------------
| Grants for mark@localhost
+------------------------------------------------------------------------------------------------------
| GRANT USAGE ON *.* TO 'mark'@'localhost' IDENTIFIED BY PASSWORD '*E6ACCEDB2495496B191ED488F598F0423
| GRANT SELECT ON `testdb`.* TO 'mark'@'localhost'
| GRANT ALL PRIVILEGES ON `mysql`.`bikes` TO 'mark'@'localhost'
+------------------------------------------------------------------------------------------------------
3 rows in set (0.00 sec)

mysql> \t;
```

This user has only SELECT privilege on the testdb database.

The user has all privileges on the bikes and mysql databases.

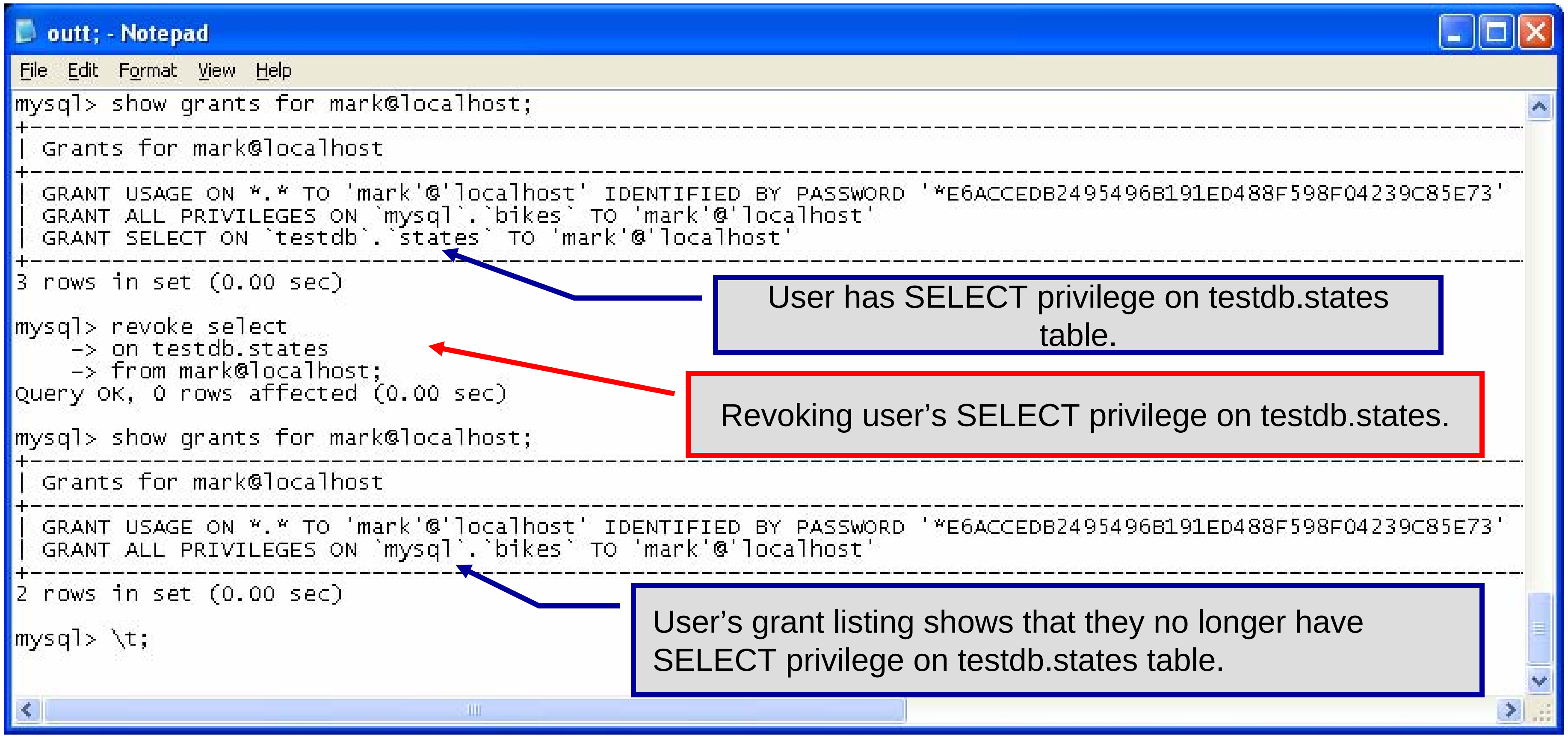
Revoking User Privileges with REVOKE

- Revocation of privileges in MySQL is accomplished with the `revoke` command.
- The format of the `revoke` command is:

```
REVOKE privileges [(column_list)]
 ON  database_name.table_name
 FROM username@hostname
```

- An example is shown on the next page.

Example - Revoking User Privileges with REV0KE



```
mysql> show grants for mark@localhost;
+----------------------------------------------------------------------------------------------------------
| Grants for mark@localhost
+----------------------------------------------------------------------------------------------------------
| GRANT USAGE ON *.* TO 'mark'@'localhost' IDENTIFIED BY PASSWORD '*E6ACCEDB2495496B191ED488F598F04239C85E73'
| GRANT ALL PRIVILEGES ON `mysql`.`bikes` TO 'mark'@'localhost'
| GRANT SELECT ON `testdb`.`states` TO 'mark'@'localhost'
+----------------------------------------------------------------------------------------------------------
3 rows in set (0.00 sec)

mysql> revoke select
    -> on testdb.states
    -> from mark@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> show grants for mark@localhost;
+----------------------------------------------------------------------------------------------------------
| Grants for mark@localhost
+----------------------------------------------------------------------------------------------------------
| GRANT USAGE ON *.* TO 'mark'@'localhost' IDENTIFIED BY PASSWORD '*E6ACCEDB2495496B191ED488F598F04239C85E73'
| GRANT ALL PRIVILEGES ON `mysql`.`bikes` TO 'mark'@'localhost'
+----------------------------------------------------------------------------------------------------------
2 rows in set (0.00 sec)

mysql> \t;
```

User has SELECT privilege on testdb.states table.

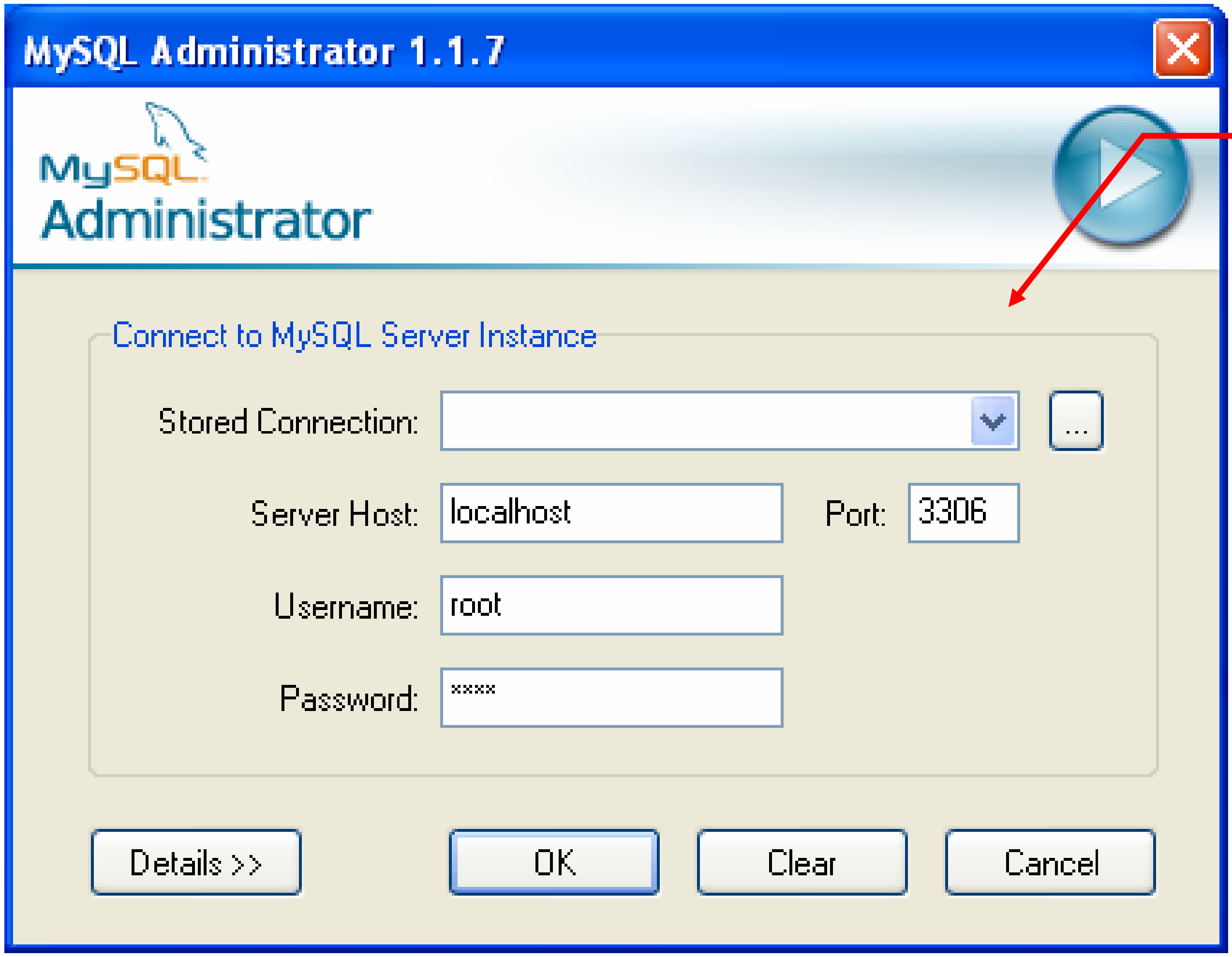
Revoking user's SELECT privilege on testdb.states.

User's grant listing shows that they no longer have SELECT privilege on testdb.states table.

The MySQL Administrator Tool

- From MySQL you can download a GUI-based administrator tool to help you administer your MySQL databases.
- This tool implements all of the GRANT, REVOKE, and SHOW functionality available in SQL.
- This tool also contains some system administrator functionality for monitoring system resources and utilization.
- You can download this tool at: http://www.mysql.com/products/.
- A few screen shots of this tool and its capabilities are shown in the next few slides.

The MySQL Administrator Tool – Screen Shots



Stored Connection:

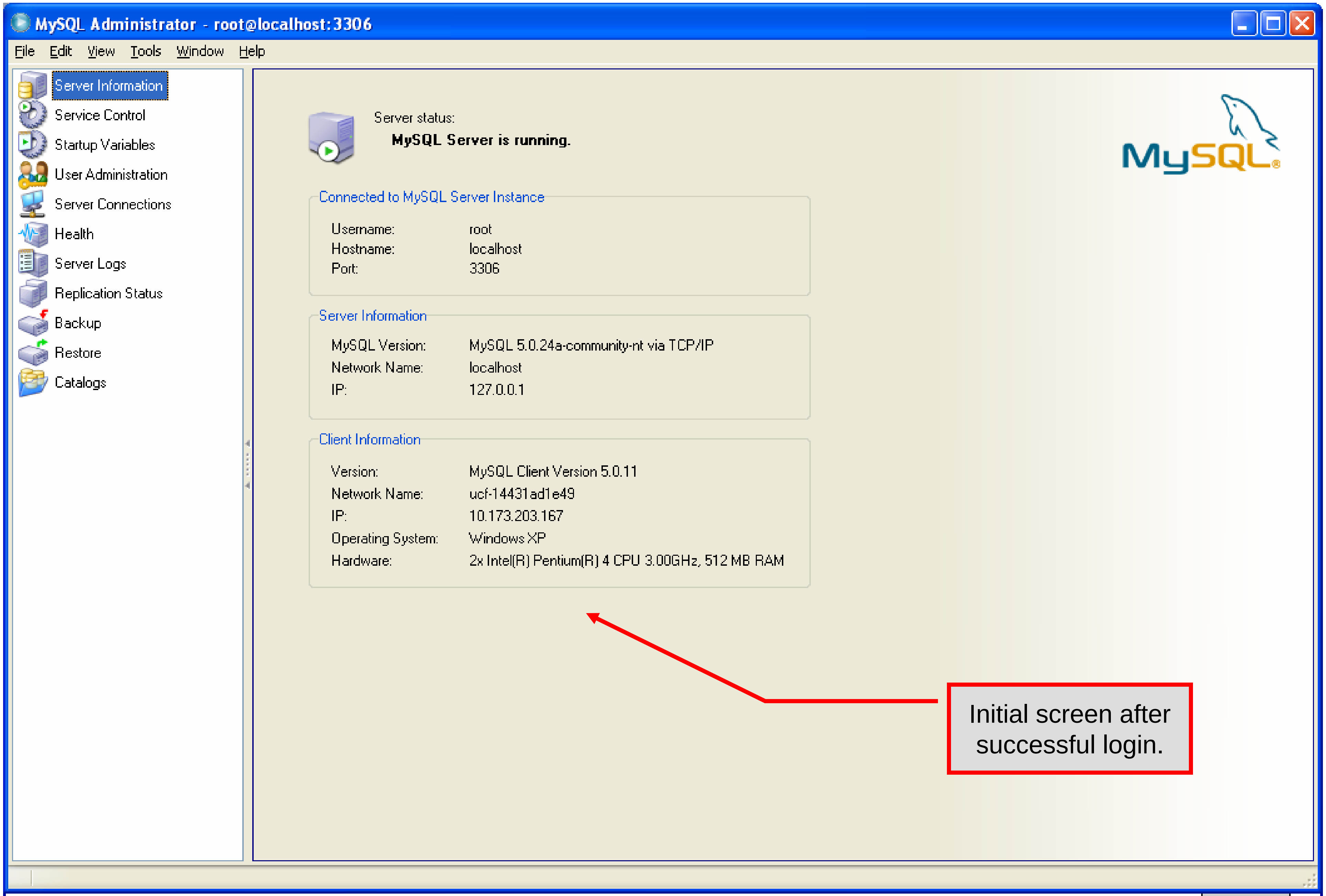
Server Host: localhost

Port: 3306

Username: root

Password: ****

Initial login screen



Server status:
MySQL Server is running.

Connected to MySQL Server Instance

Username:	root
Hostname:	localhost
Port:	3306

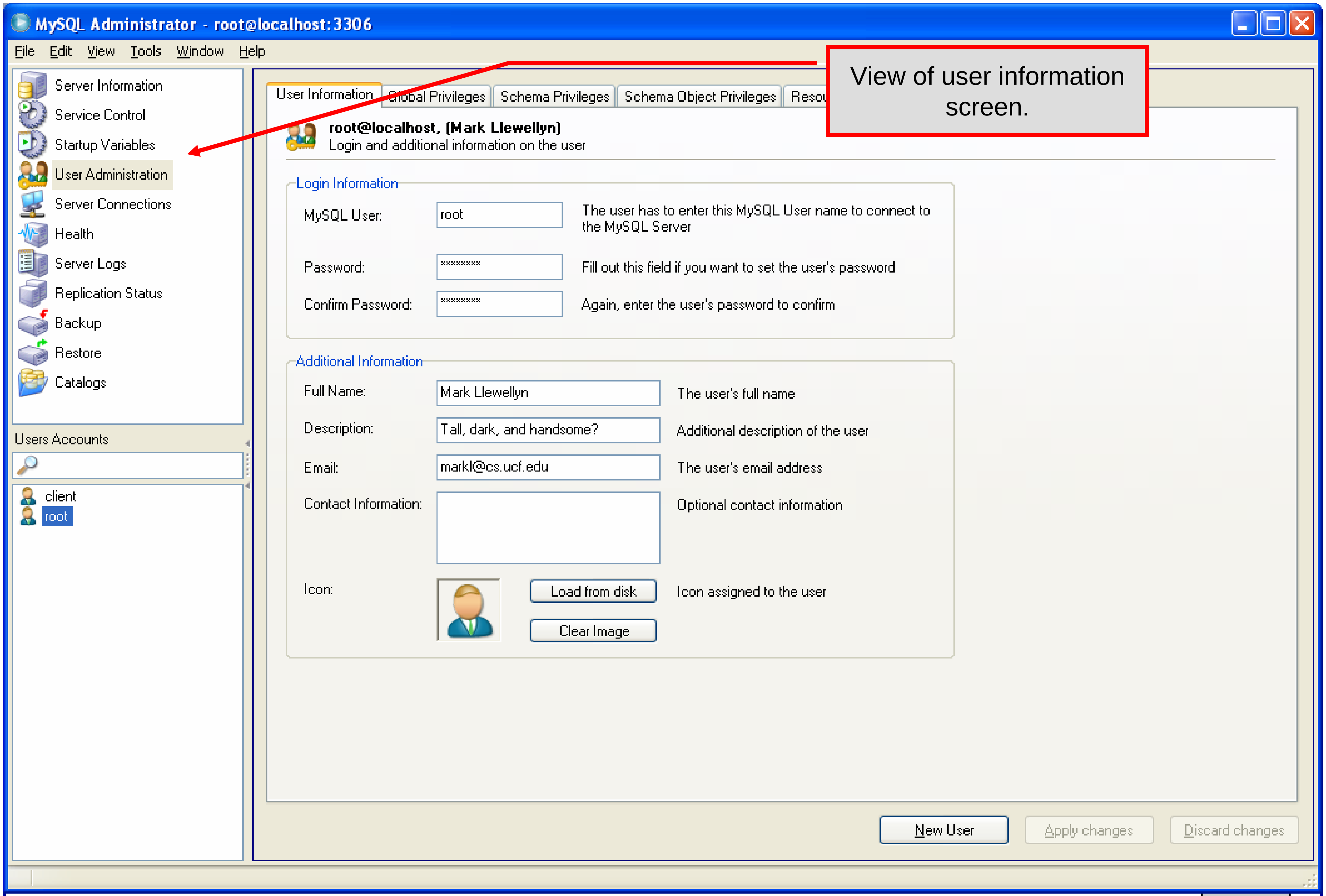
Server Information

MySQL Version:	MySQL 5.0.24a-community-nt via TCP/IP
Network Name:	localhost
IP:	127.0.0.1

Client Information

Version:	MySQL Client Version 5.0.11
Network Name:	ucf-14431ad1e49
IP:	10.173.203.167
Operating System:	Windows XP
Hardware:	2x Intel(R) Pentium(R) 4 CPU 3.00GHz, 512 MB RAM

Initial screen after successful login.



root@localhost, (Mark Llewellyn)
Login and additional information on the user

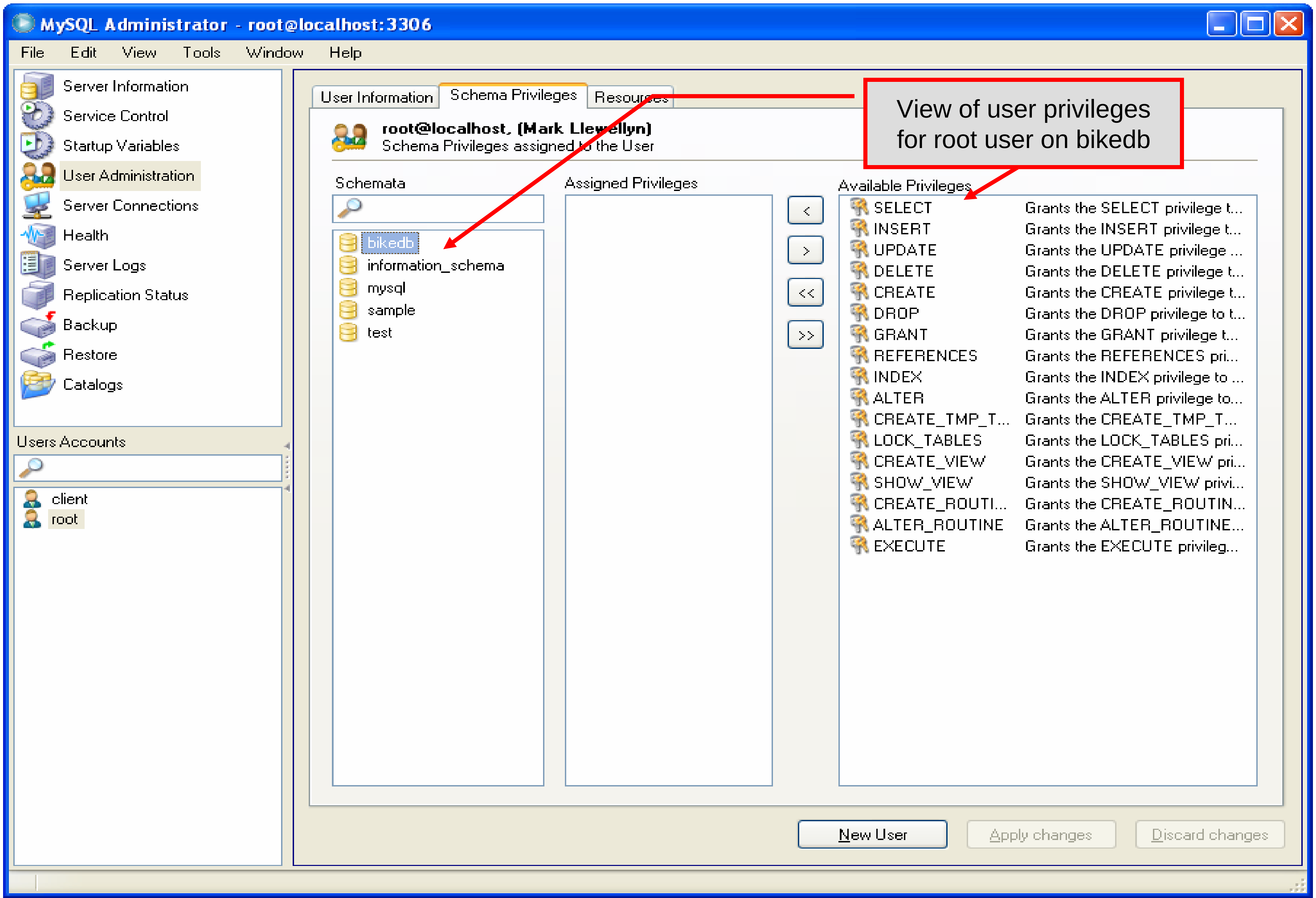
Login Information

MySQL User:	root	The user has to enter this MySQL User name to connect to the MySQL Server
Password:	********	Fill out this field if you want to set the user's password
Confirm Password:	********	Again, enter the user's password to confirm

Additional Information

Full Name:	Mark Llewellyn	The user's full name
Description:	Tall, dark, and handsome?	Additional description of the user
Email:	markl@cs.ucf.edu	The user's email address
Contact Information:		Optional contact information
Icon:		Icon assigned to the user

View of user information screen.



MySQL Administrator - root@localhost:3306

File Edit View Tools Window Help

Server Information
Service Control
Startup Variables
User Administration
Server Connections
Health
Server Logs
Replication Status
Backup
Restore
Catalogs

Users Accounts

client
root

User Information | Schema Privileges | Resources

root@localhost, (Mark Llewellyn)
Schema Privileges assigned to the User

Schemata

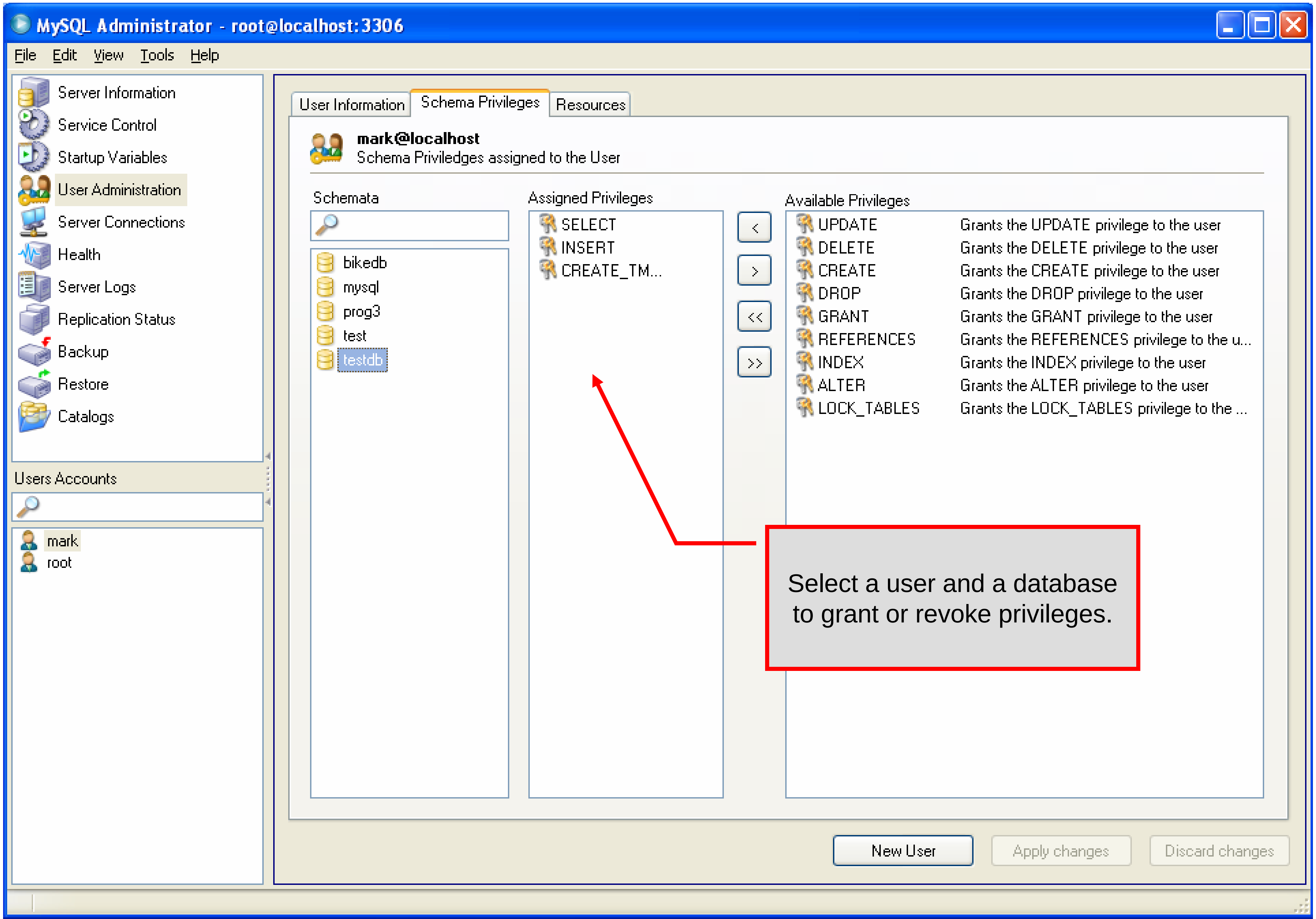
bikedb
information_schema
mysql
sample
test

Assigned Privileges

Available Privileges

SELECT	Grants the SELECT privilege t...
INSERT	Grants the INSERT privilege t...
UPDATE	Grants the UPDATE privilege ...
DELETE	Grants the DELETE privilege t...
CREATE	Grants the CREATE privilege t...
DROP	Grants the DROP privilege to t...
GRANT	Grants the GRANT privilege t...
REFERENCES	Grants the REFERENCES pri...
INDEX	Grants the INDEX privilege to ...
ALTER	Grants the ALTER privilege to...
CREATE_TMP_T...	Grants the CREATE_TMP_T...
LOCK_TABLES	Grants the LOCK_TABLES pri...
CREATE_VIEW	Grants the CREATE_VIEW pri...
SHOW_VIEW	Grants the SHOW_VIEW privi...
CREATE_ROUTI...	Grants the CREATE_ROUTIN...
ALTER_ROUTINE	Grants the ALTER_ROUTINE...
EXECUTE	Grants the EXECUTE privileg...

View of user privileges for root user on bikedb



mark@localhost
Schema Priviledges assigned to the User

Schemata	Assigned Privileges
bikedb	SELECT
mysql	INSERT
prog3	CREATE_TM...
test	
testdb	

Available Privileges

UPDATE	Grants the UPDATE privilege to the user
DELETE	Grants the DELETE privilege to the user
CREATE	Grants the CREATE privilege to the user
DROP	Grants the DROP privilege to the user
GRANT	Grants the GRANT privilege to the user
REFERENCES	Grants the REFERENCES privilege to the u...
INDEX	Grants the INDEX privilege to the user
ALTER	Grants the ALTER privilege to the user
LOCK_TABLES	Grants the LOCK_TABLES privilege to the ...

Select a user and a database to grant or revoke privileges.

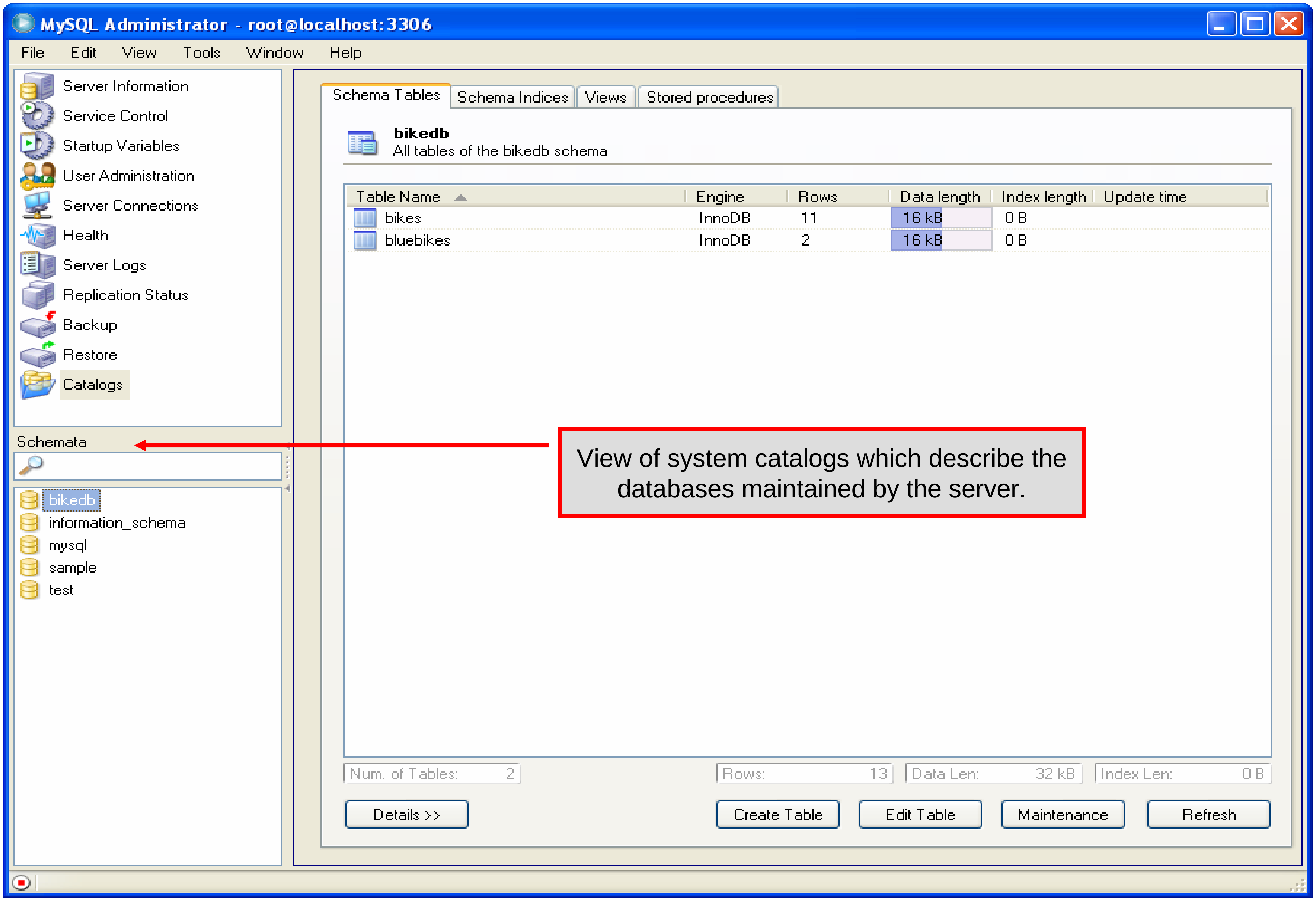


Table Name	Engine	Rows	Data length	Index length	Update time
bikes	InnoDB	11	16 kB	0 B	
bluebikes	InnoDB	2	16 kB	0 B	

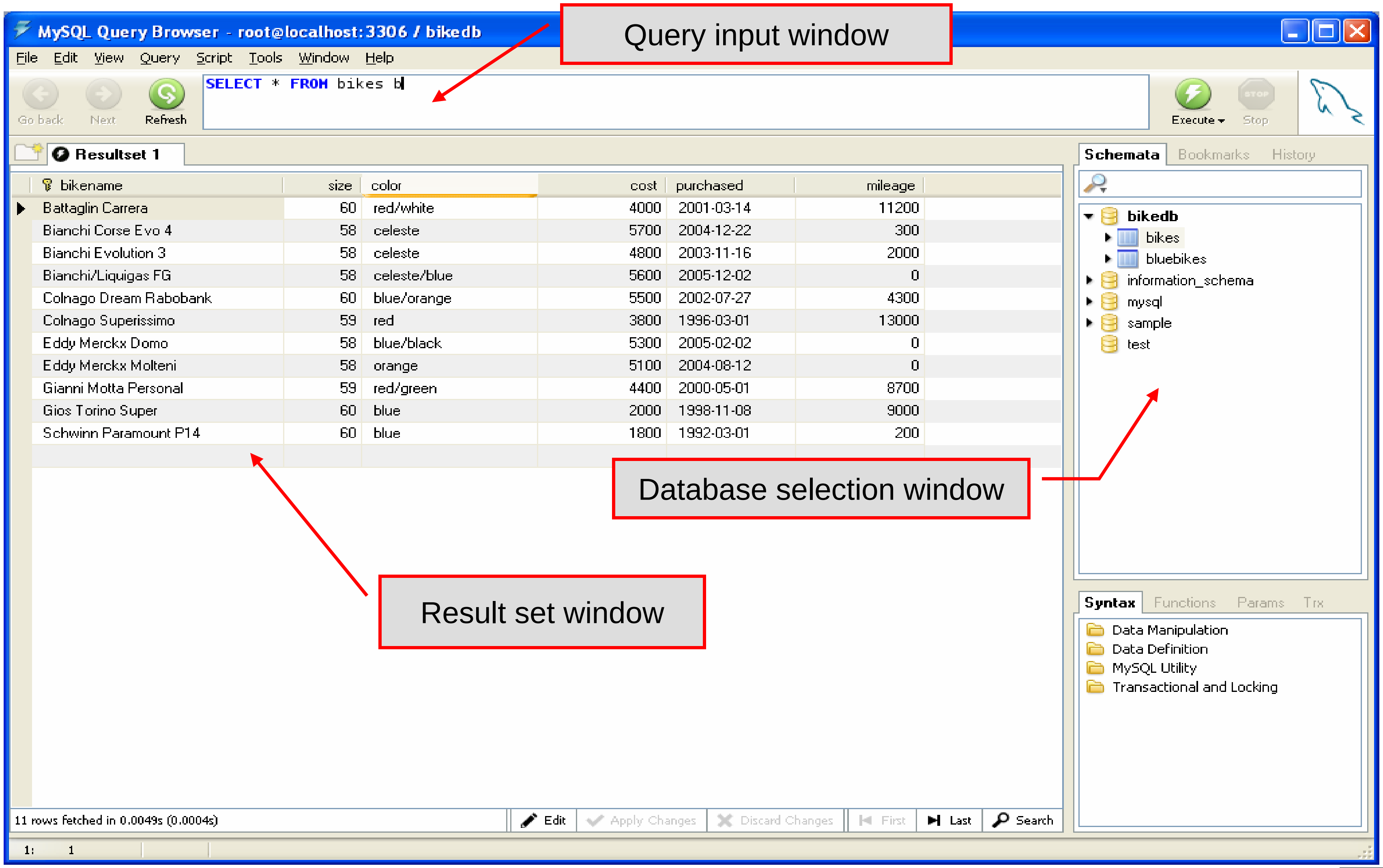
Num. of Tables: 2 Rows: 13 Data Len: 32 kB Index Len: 0 B

View of system catalogs which describe the databases maintained by the server.

The MySQL Query Browser Tool

- From MySQL you can also download a GUI-based query browser tool.
- This tool implements all of the basic DML side of SQL with some limitation. For example, editing result sets is possible only if the result set was generated from a single table. Join-based result sets are not editable. This tool also implements many DDL commands.
- This tool is helpful for developing and testing queries.
- You can download this tool at: http://dev.mysql.com/downloads/
- A few screen shots of this tool and its capabilities are shown in the next few slides.

A First Look At The MySQL Query Browser

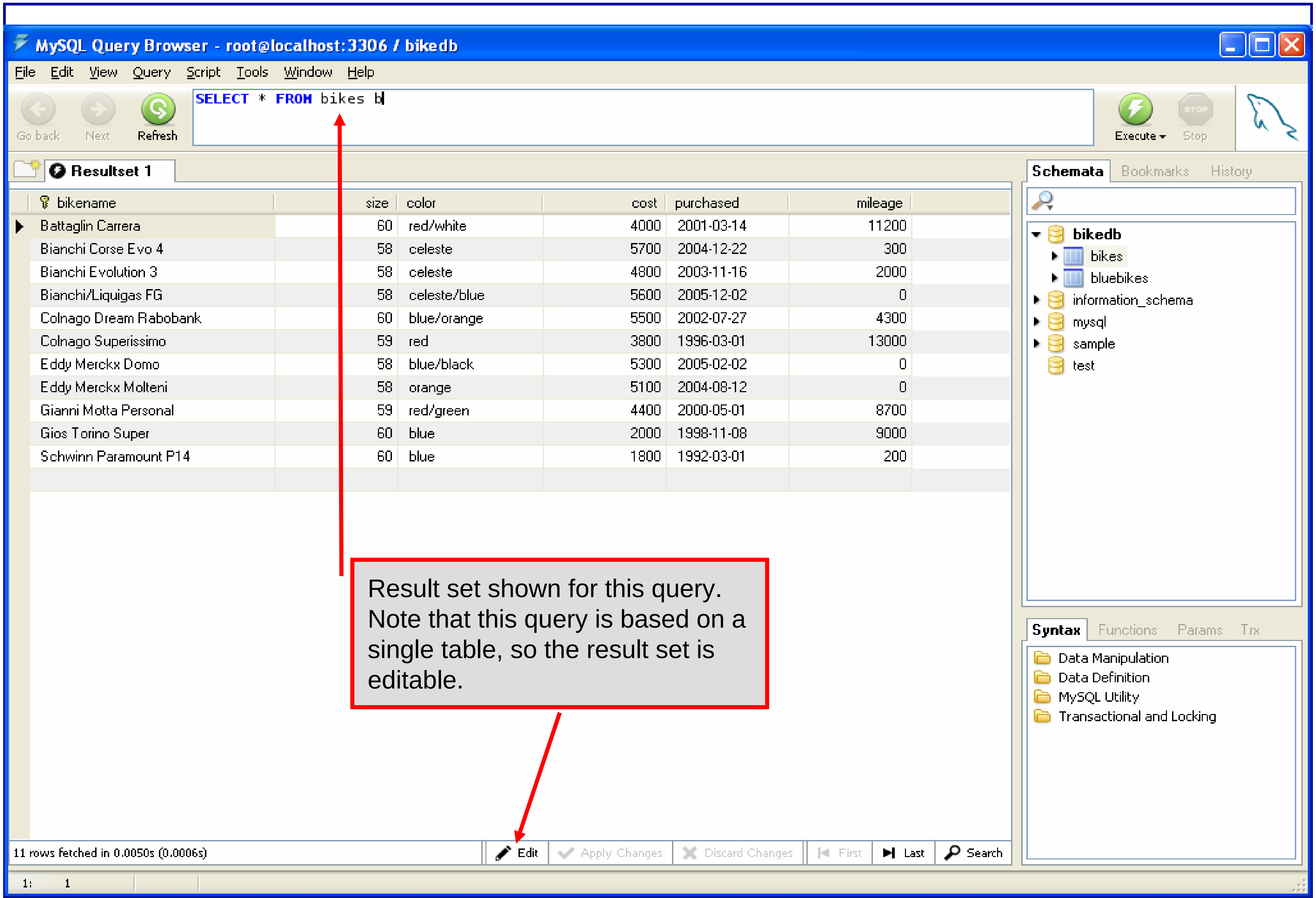


```
SELECT * FROM bikes b
```

bikename	size	color	cost	purchased	mileage
Battaglin Carrera	60	red/white	4000	2001-03-14	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-22	300
Bianchi Evolution 3	58	celeste	4800	2003-11-16	2000
Bianchi/Liquigas FG	58	celeste/blue	5600	2005-12-02	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-27	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2005-02-02	0
Eddy Merckx Molteni	58	orange	5100	2004-08-12	0
Gianni Motta Personal	59	red/green	4400	2000-05-01	8700
Gios Torino Super	60	blue	2000	1998-11-08	9000
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

11 rows fetched in 0.0049s (0.0004s)

1: 1

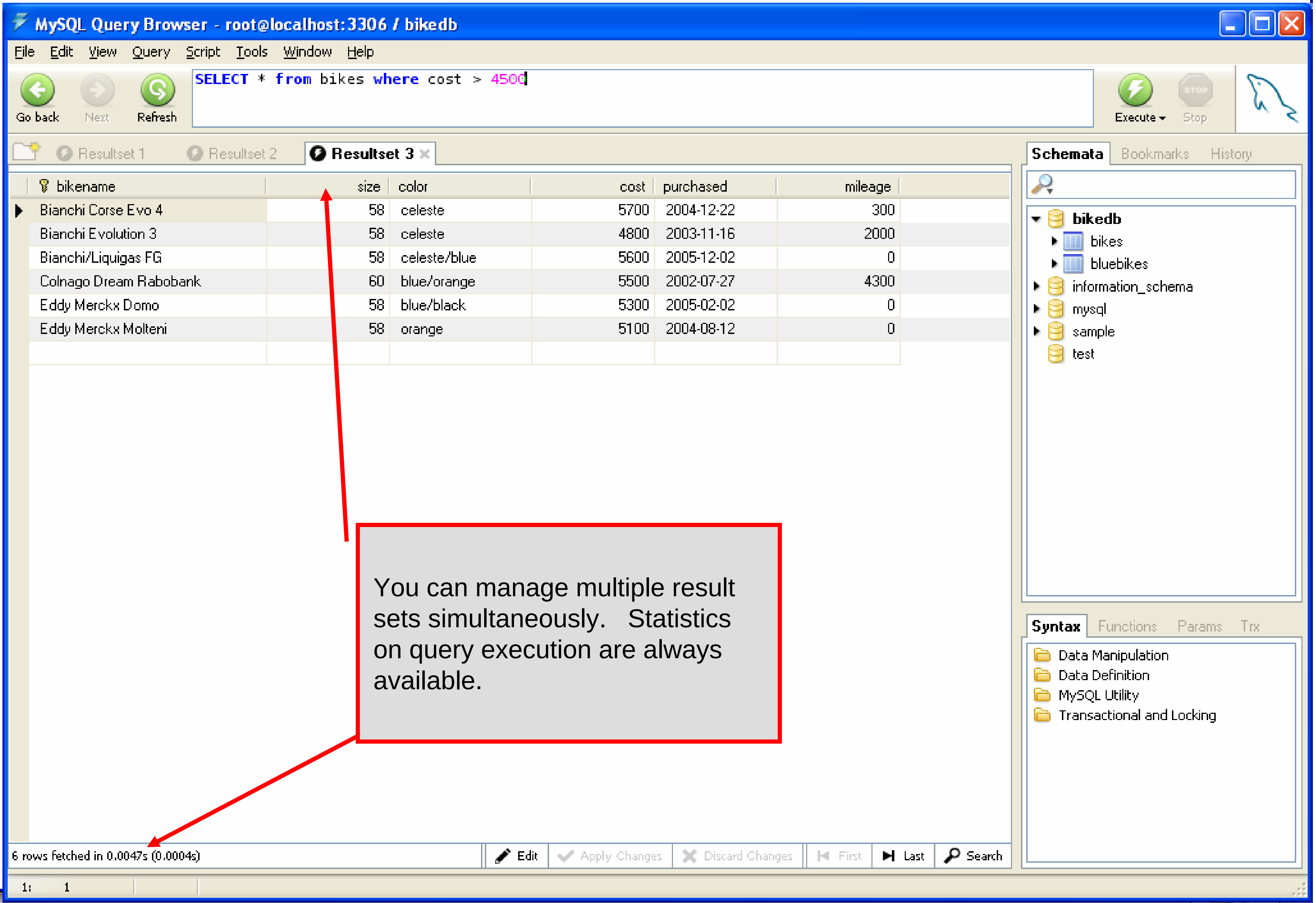


```
SELECT * FROM bikes b
```

bikename	size	color	cost	purchased	mileage
Battaglin Carrera	60	red/white	4000	2001-03-14	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-22	300
Bianchi Evolution 3	58	celeste	4800	2003-11-16	2000
Bianchi/Liquigas FG	58	celeste/blue	5600	2005-12-02	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-27	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2005-02-02	0
Eddy Merckx Molteni	58	orange	5100	2004-08-12	0
Gianni Motta Personal	59	red/green	4400	2000-05-01	8700
Gios Torino Super	60	blue	2000	1998-11-08	9000
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

```
11 rows fetched in 0.0050s (0.0006s)
```

Result set shown for this query. Note that this query is based on a single table, so the result set is editable.



```
SELECT * from bikes where cost > 4500
```

bikename	size	color	cost	purchased	mileage
Bianchi Corse Evo 4	58	celeste	5700	2004-12-22	300
Bianchi Evolution 3	58	celeste	4800	2003-11-16	2000
Bianchi/Liquigas FG	58	celeste/blue	5600	2005-12-02	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-27	4300
Eddy Merckx Domo	58	blue/black	5300	2005-02-02	0
Eddy Merckx Molteni	58	orange	5100	2004-08-12	0

```
6 rows fetched in 0.0047s (0.0004s)
```

You can manage multiple result sets simultaneously. Statistics on query execution are always available.