

# CGS 2545: Database Concepts Spring 2014

## Introduction To MySQL Installation Of MySQL 5.6.16

Instructor : Dr. Mark Llewellyn  
markl@cs.ucf.edu  
HEC 236, 407-823-2790  
<http://www.cs.ucf.edu/courses/cgs2545/spr2014>

Department of Electrical Engineering and Computer Science  
Computer Science Division  
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 5.6.16 and can be downloaded from [www.mysql.com](http://www.mysql.com).
- 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 :: The world's most popular open source database

MySQL.com Downloads (GA)

Products Services Partners Customers Why MySQL? News & Events How to Buy

**GET STARTED**

- Try Now
- MySQL Enterprise Edition
- Free Webinars
- White Papers
- ISVs and OEMs
- Buy Now
- Contact Us

**MySQL Enterprise Edition**  
Download Now »

**Improved!**  
**MySQL Query Analyzer**  
Performance Tuning & Analysis  
Learn More »

**Free Webinars**

- MySQL 5.7 最新情報と Fabric  
Thursday, February 13, 2014
- Monitoring MySQL at Scale  
Wednesday, February 19, 2014
- 50 Tips to Boost MySQL Performance  
Wednesday, February 26, 2014
- Developing Windows Applications with MySQL: Part III, Migrating from Microsoft SQL Server to MySQL  
Thursday, February 27, 2014

More Webinars

Products Resources News & Events

Click here to go to download page



This should be the next page you see.  
Scroll down this page until you find the MySQL  
Community Edition and click that link.

MySQL.com Downloads (GA)

Current Downloads (Generally Available)

## MySQL Downloads

### Contact Sales

USA: +1-866-221-0634  
Canada: +1-866-221-0634

Germany: +49 89 143 01280  
France: +33 1 57 60 83 57  
Italy: +39 02 249 59 120  
UK: +44 207 553 8447

Japan: 0120-065556  
China: 10800-811-0823  
India: 0008001005870

[More Countries »](#)

[Contact Us Online »](#)

### MySQL Enterprise Edition (commercial)

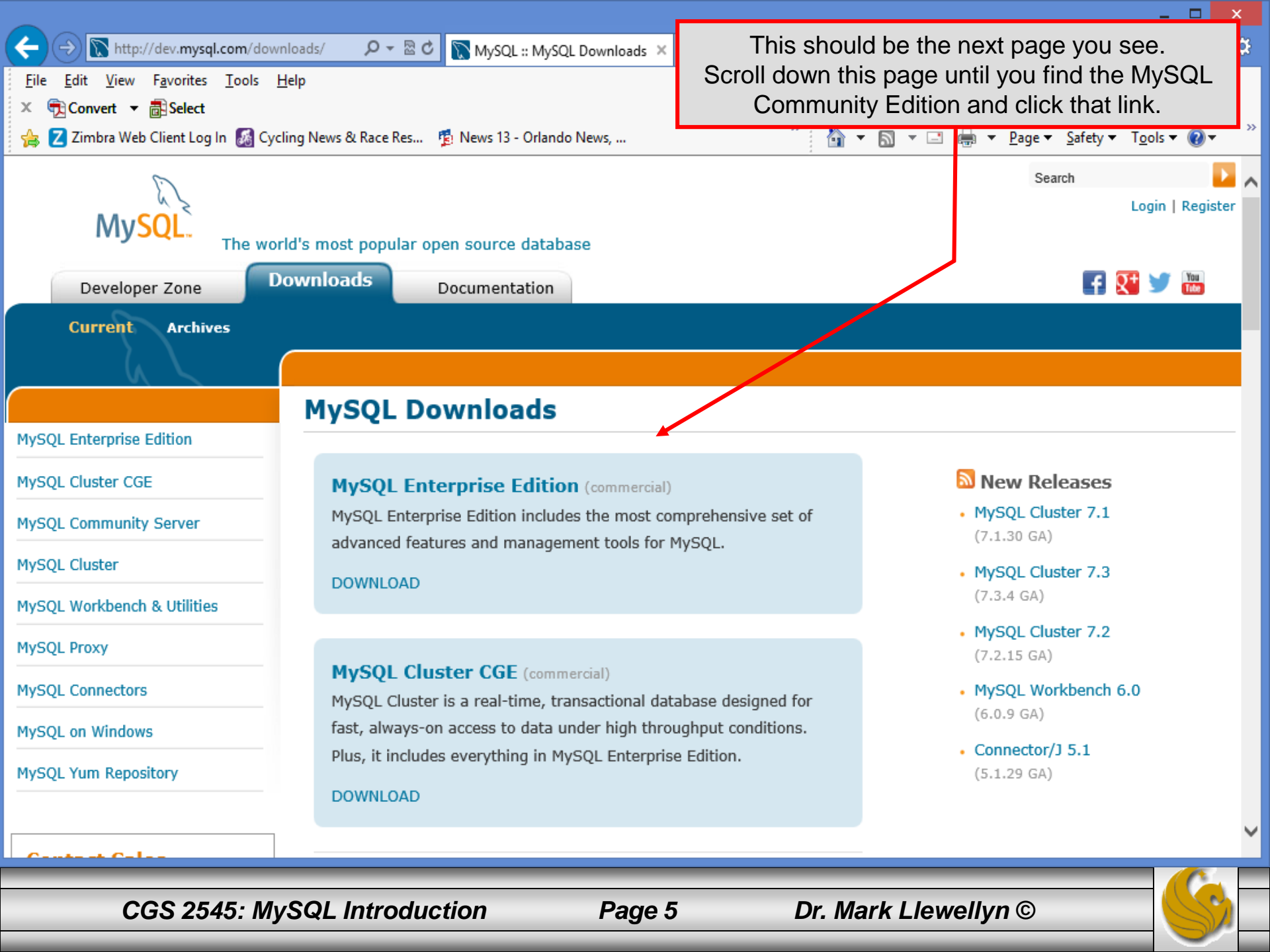
MySQL Enterprise Edition includes the most comprehensive set of advanced features and management tools for MySQL.

- MySQL Database
- MySQL Storage Engines (InnoDB, MyISAM, etc.)
- MySQL Connectors (JDBC, ODBC, .Net, etc.)
- MySQL Workbench
- MySQL Replication
- MySQL Partitioning
- MySQL Enterprise Backup
- MySQL Enterprise Monitor
- MySQL Enterprise HA
- MySQL Enterprise Scalability
- MySQL Enterprise Security
- MySQL Enterprise Audit

[Learn More »](#)

[Download from Oracle eDelivery »](#)





This should be the next page you see. Scroll down this page until you find the MySQL Community Edition and click that link.

## MySQL Downloads

### MySQL Enterprise Edition (commercial)

MySQL Enterprise Edition includes the most comprehensive set of advanced features and management tools for MySQL.

[DOWNLOAD](#)

### MySQL Cluster CGE (commercial)

MySQL Cluster is a real-time, transactional database designed for fast, always-on access to data under high throughput conditions. Plus, it includes everything in MySQL Enterprise Edition.

[DOWNLOAD](#)

### New Releases

- [MySQL Cluster 7.1](#) (7.1.30 GA)
- [MySQL Cluster 7.3](#) (7.3.4 GA)
- [MySQL Cluster 7.2](#) (7.2.15 GA)
- [MySQL Workbench 6.0](#) (6.0.9 GA)
- [Connector/J 5.1](#) (5.1.29 GA)



← → http://dev.mysql.com/downloads/m MySQL :: Download MySQL... x

File Edit View Favorites Tools Help

Convert Select

Zimbra Web Client Log In Cycling News & Race Res... News 13 - Orlando News, ...

Search Login | Register

MySQL The world's most popular open source database

Developer Zone Downloads Documentation

Current Archives

## Download MySQL Community Server

MySQL Enterprise Edition

MySQL Cluster CGE

**MySQL Community Server**

MySQL Cluster

MySQL Workbench & Utilities

MySQL Proxy

MySQL Connectors

MySQL on Windows

MySQL Yum Repository

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 Cluster Community Edition is available as a separate download. The reason for this change is so that MySQL Cluster can provide more frequent updates and support using the latest sources of MySQL Cluster Carrier Grade Edition.

MySQL open source software is provided under the [GPL License](#).

OEMs, ISVs and VARs can purchase commercial licenses.

Important Platform Support Updates

Online Documentation: Looking for previous GA versions?

The MySQL Community Server page.



http://dev.mysql.com/downloads/m MySQL :: Download MySQL... x

File Edit View Favorites Tools Help

Convert Select

Zimbra Web Client Log In Cycling News & Race Res... News 13 - Orlando News, ...

**Related Pages:**

- Technical Articles
- Documentation

## MySQL Community Server 5.6.1

Select Platform:

Microsoft Windows

**Recommended Download:**

### MySQL Installer 5.6 for Windows

All MySQL Products. For All Windows Platforms.  
In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the server-only MSI packages.

**Windows (x86, 64-bit), MySQL Installer MSI** **Download**

**Other Downloads:**

<b>Windows (x86, 32-bit), ZIP</b>	5.6.16	212.1M	<b>Download</b>
-----------------------------------	--------	--------	-----------------

Archive

http://dev.mysql.com/downloads/installer/5.6.html

Use this selection window to select the proper platform/version for your system 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.

Looking for previous GA versions?





MySQL :: Download MySQL... x

File Edit View Favorites Tools Help

Convert Select

Zimbra Web Client Log In Cycling News & Race Res...

MySQL The world's most popular open source database

Developer Zone Downloads Documentation

Current Archives

## Download MySQL Community Server

MySQL Enterprise Edition

MySQL Cluster CGE

**MySQL Community Server**

MySQL Cluster

MySQL Workbench & Utilities

MySQL Proxy

MySQL Connectors

MySQL on Windows

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 Cluster Community Edition is available as a separate download. The reason for this change is so that MySQL Cluster can provide more frequent updates and support using the latest sources of MySQL Cluster Carrier Grade Edition.

MySQL open source software is provided under the [GPL License](#).

OEMs, ISVs and VARs can purchase commercial licenses.

Important Platform Support Updates

If you are not using the all-inclusive loader, go back to the main download page and also download MySQL Workbench which contains the Administrator and MySQL Query Browser GUI tools.



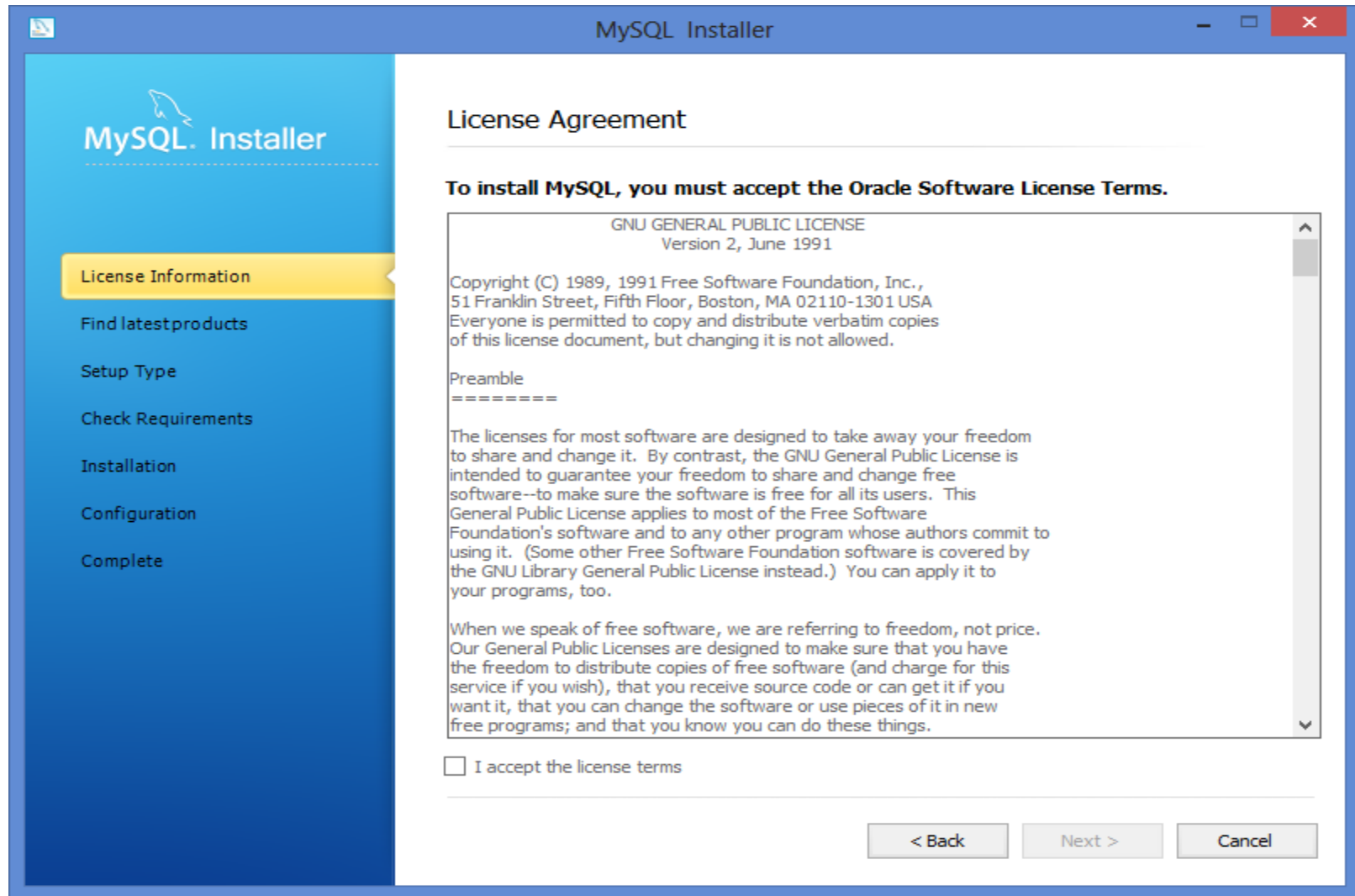


# Installing MySQL 5.6.16

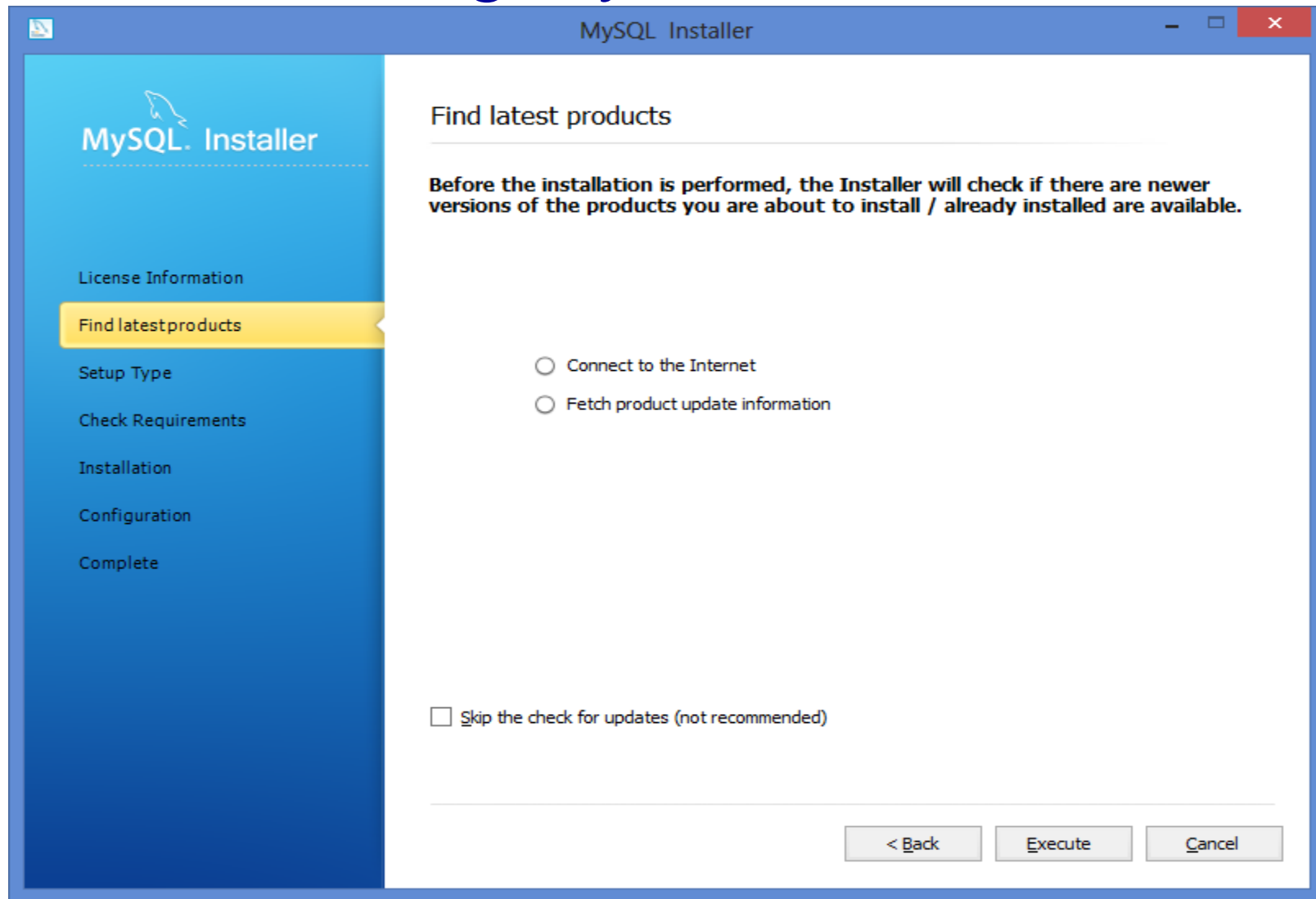
- Once you've got MySQL downloaded, go through the installation process. It may vary somewhat depending on platform.
- I've illustrated the basic install on Windows 8 over the next few pages, just to give you an idea of what you should be seeing.



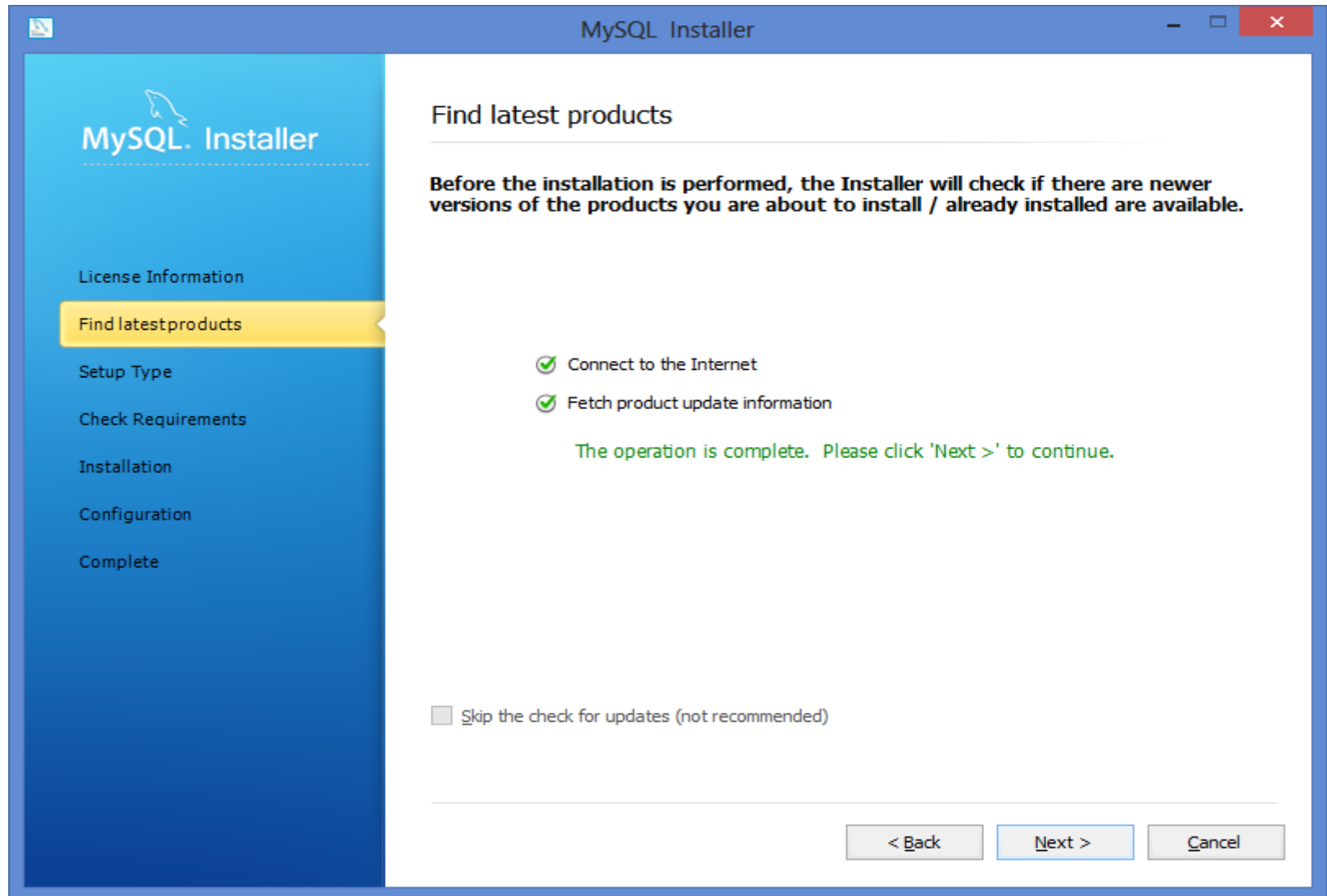
# Installing MySQL 5.6.16 (cont.)



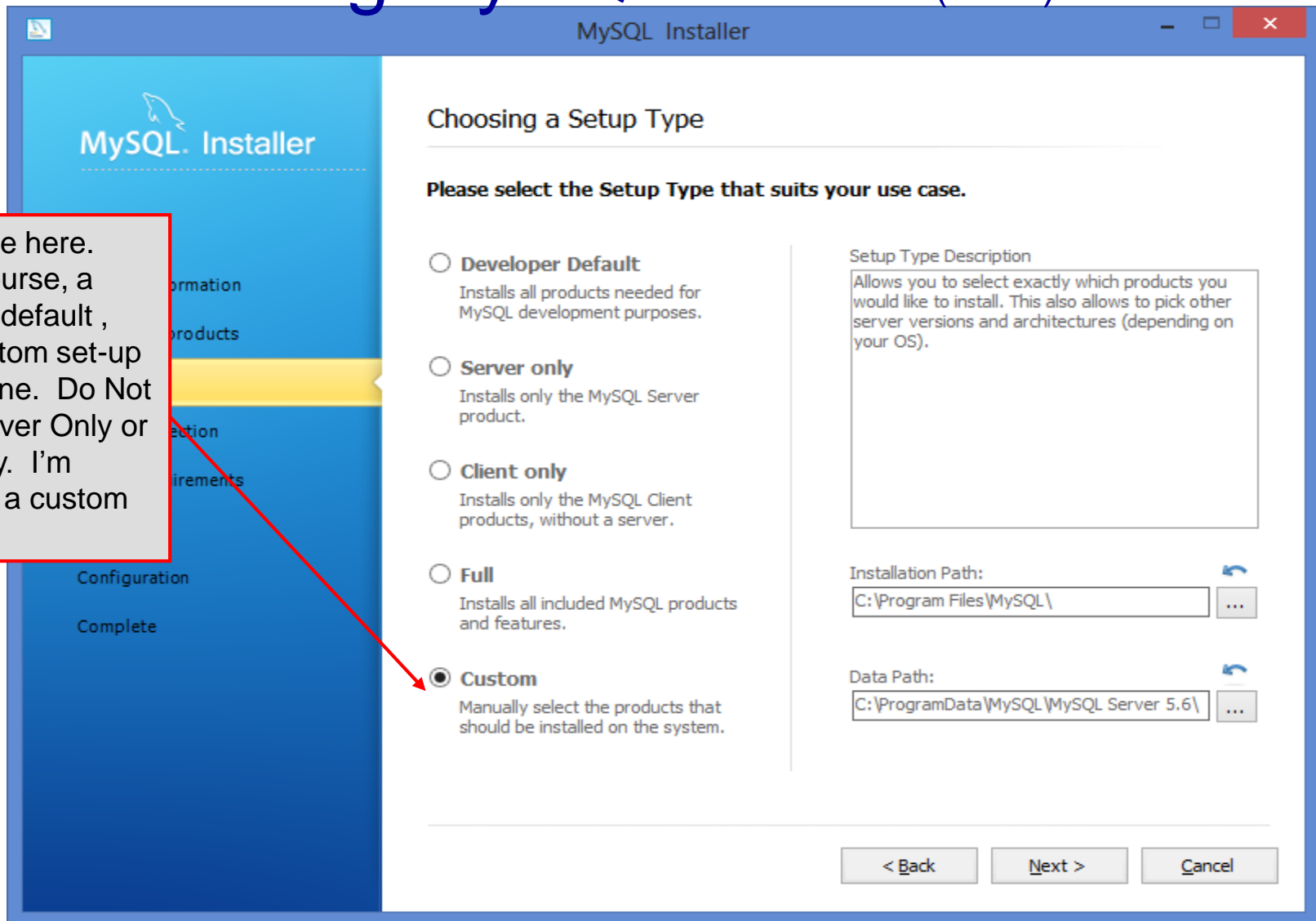
# Installing MySQL 5.6.16 (cont.)



# Installing MySQL 5.6.16 (cont.)



# Installing MySQL 5.6.16 (cont.)



MySQL Installer

## Choosing a Setup Type

Please select the Setup Type that suits your use case.

- ☐ **Developer Default**  
Installs all products needed for MySQL development purposes.
- ☐ **Server only**  
Installs only the MySQL Server product.
- ☐ **Client only**  
Installs only the MySQL Client products, without a server.
- ☐ **Full**  
Installs all included MySQL products and features.
- ☒ **Custom**  
Manually select the products that should be installed on the system.

Setup Type Description  
Allows you to select exactly which products you would like to install. This also allows to pick other server versions and architectures (depending on your OS).

Installation Path:  
C:\Program Files\MySQL\

Data Path:  
C:\ProgramData\MySQL\MySQL Server 5.6\

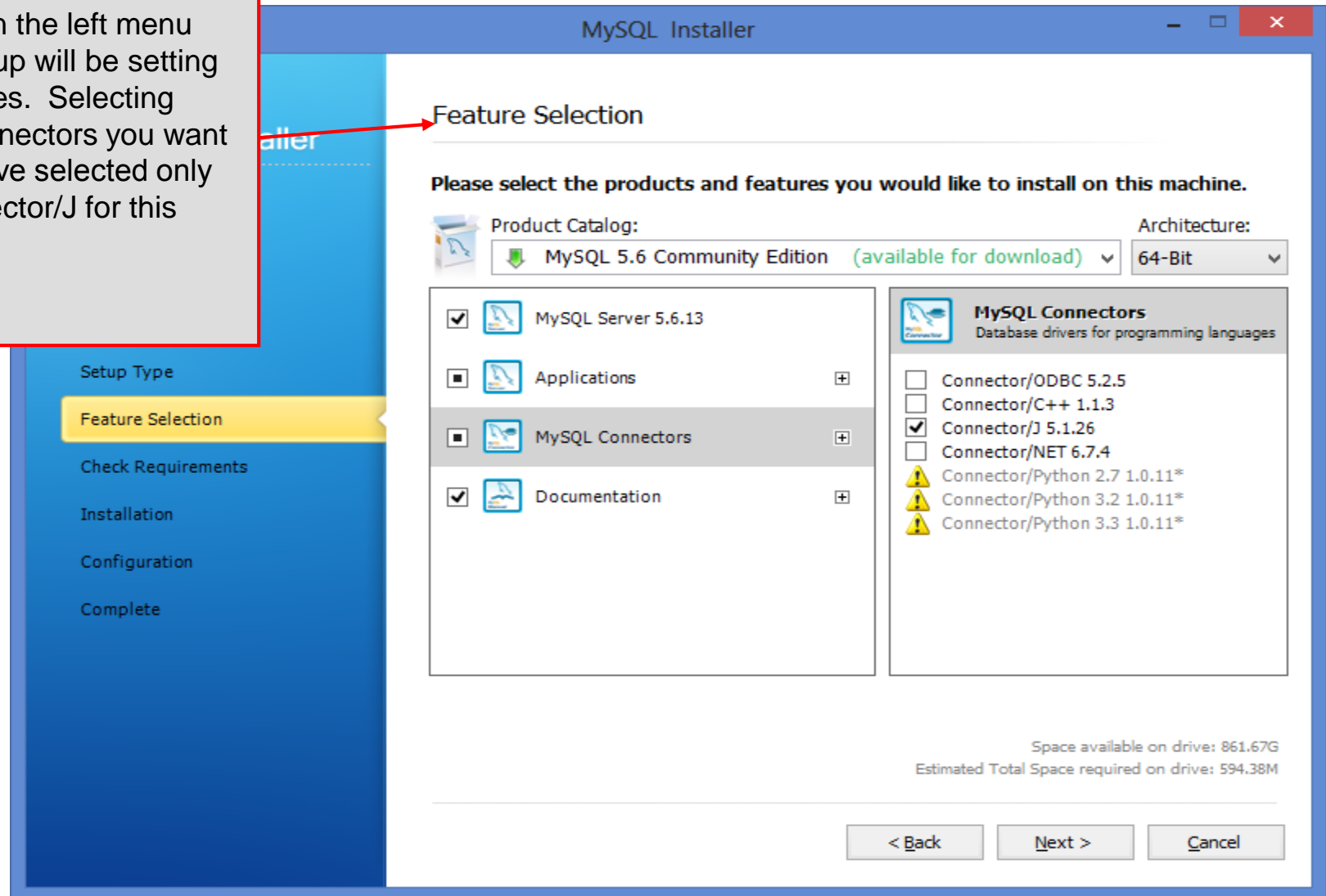
< Back   Next >   Cancel

Your choice here.  
For this course, a developer default, full, or custom set-up will work fine. Do Not Select Server Only or Client Only. I'm illustrating a custom set-up.



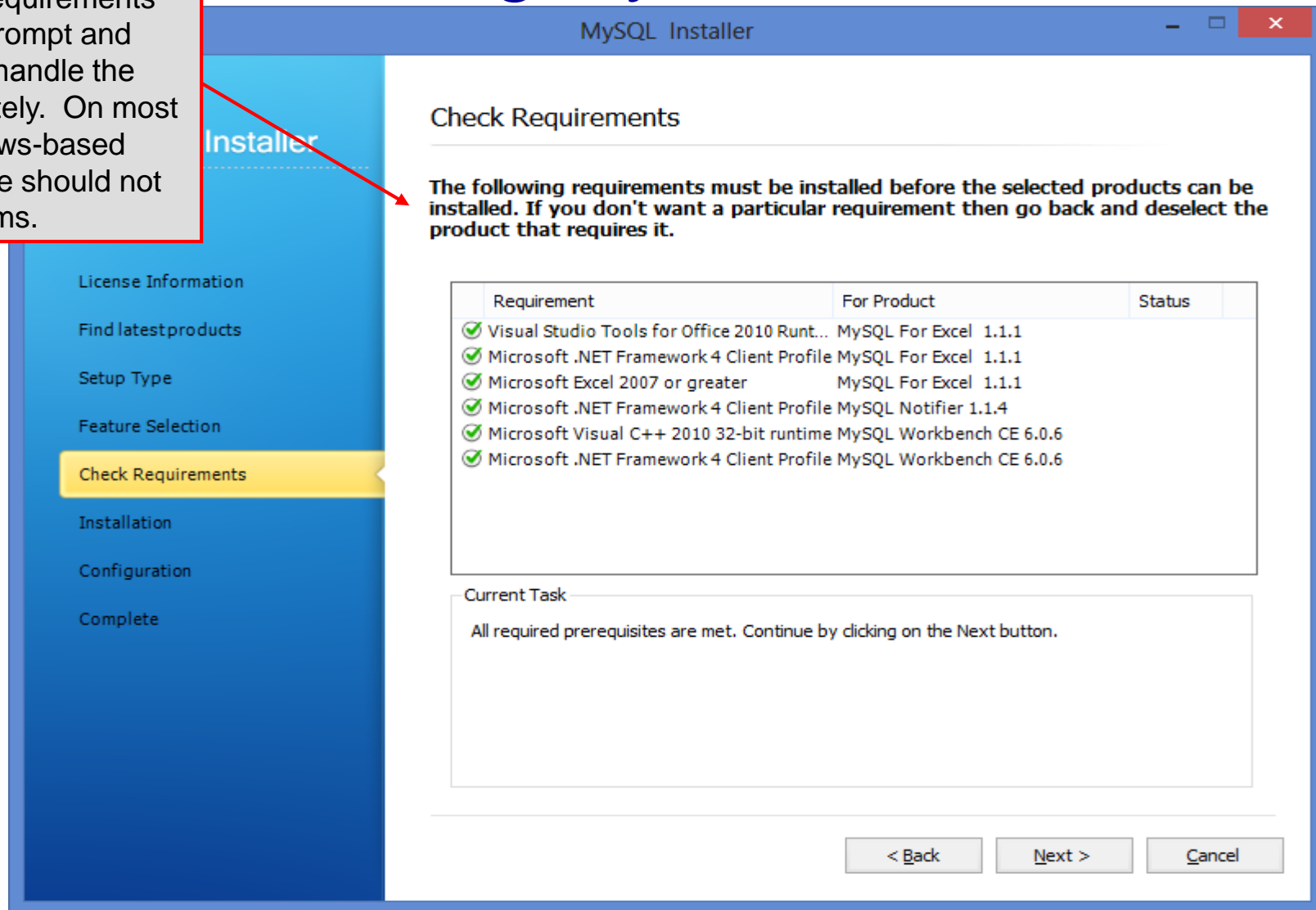
On a custom install, you'll go through each of the choices on the left menu list. First up will be setting the features. Selecting which connectors you want loaded. I've selected only the Connector/J for this server.

# Installing MySQL 5.6.16 (cont.)



A requirements check looks for all of the supporting tools that MySQL needs. Any missing requirements will initiate a prompt and you'll need to handle the issues separately. On most current Windows-based machines there should not be any problems.

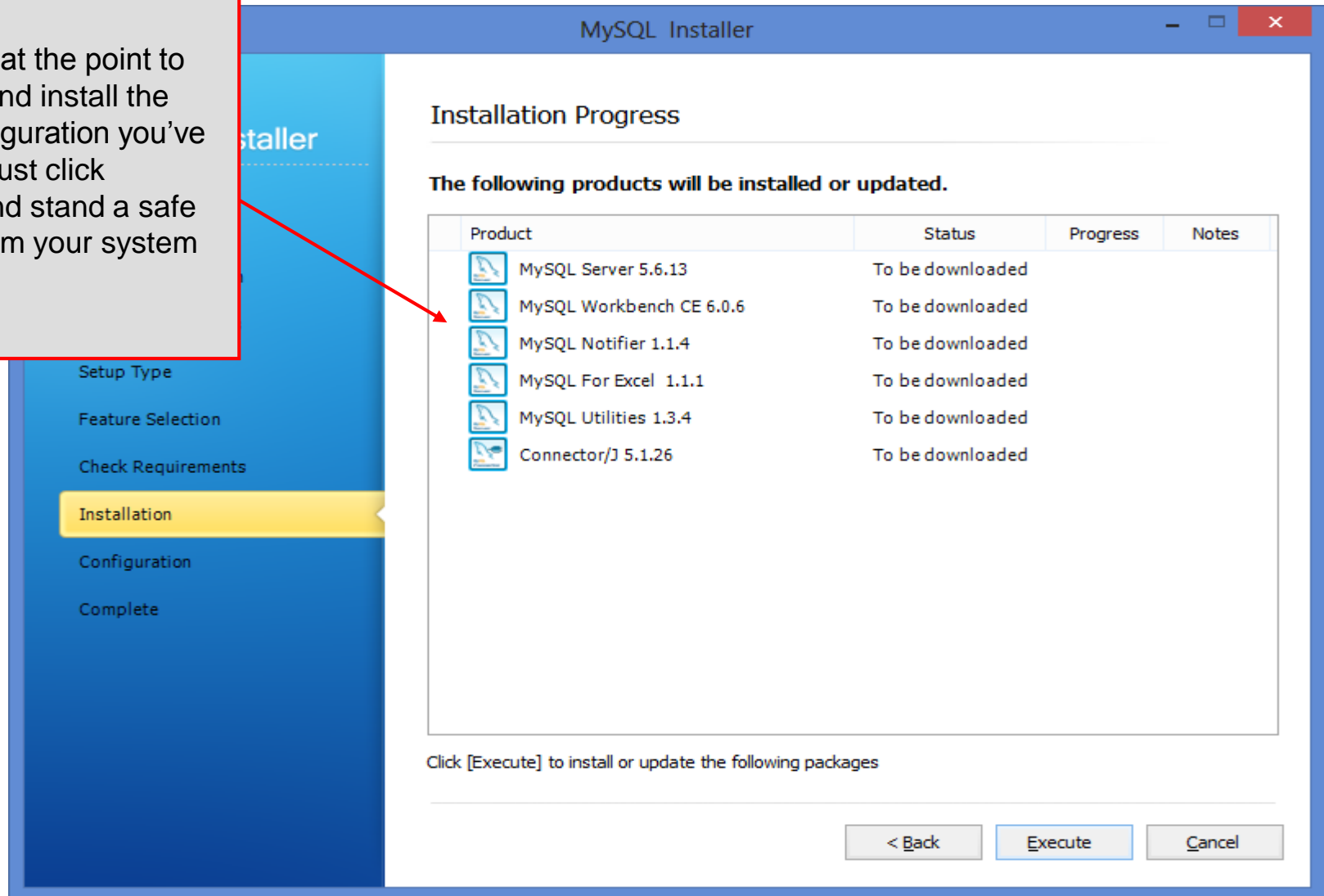
# Installing MySQL 5.6.16 (cont.)





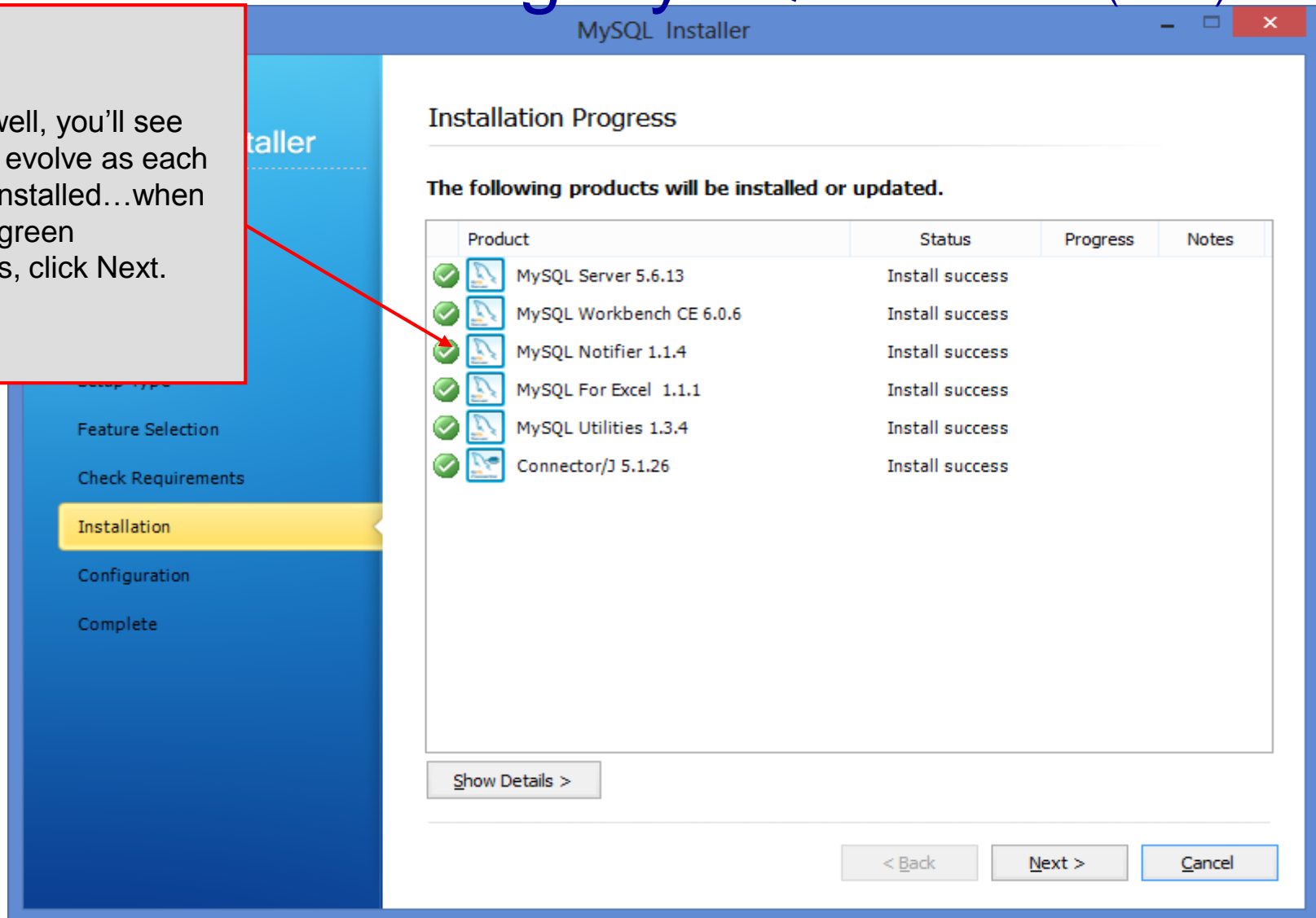
# Installing MySQL 5.6.16 (cont.)

You're now at the point to download and install the server configuration you've selected. Just click Execute (and stand a safe distance from your system 😊).



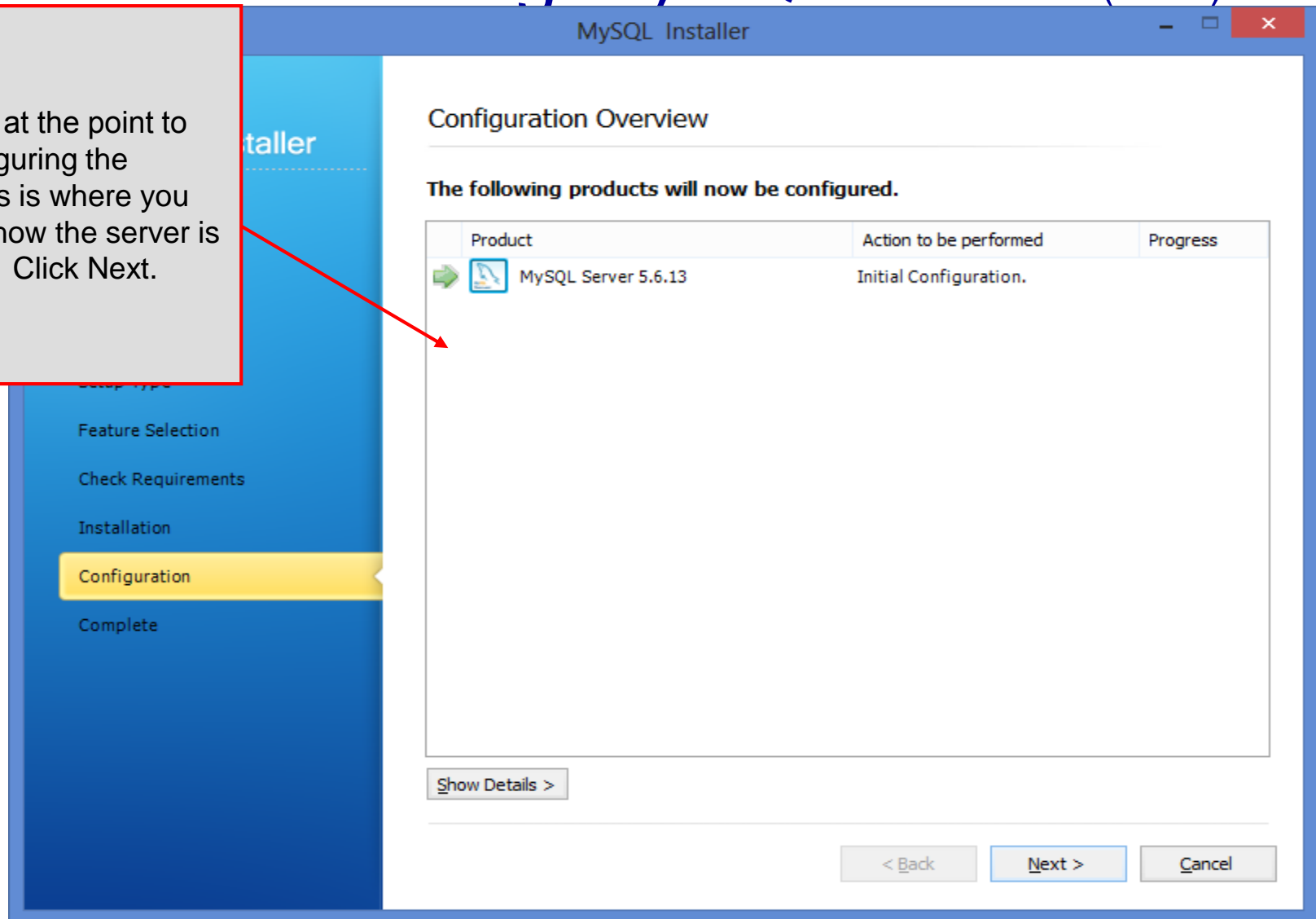
# Installing MySQL 5.6.16 (cont.)

If all went well, you'll see this screen evolve as each product is installed...when you get all green checkmarks, click Next.



# Installing MySQL 5.6.16 (cont.)

You're now at the point to begin configuring the server. This is where you customize how the server is to behave. Click Next.



# Installing MySQL 5.6.16 (cont.)

Select Config Type:  
Development Machine.  
Check TCP/IP enable. Port  
will default to 3306, which is  
fine. I have several MySQL  
servers running on different  
ports. We don't need any  
advanced options at this  
point. Click Next.

MySQL Installer

MySQL Server Configuration 1 / 3

**Server Configuration Type**

Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.

**Config Type:** Development Machine

☒ **Enable TCP/IP Networking**

Enable this to allow TCP/IP networking. Only localhost connections through named pipes are allowed when this option is skipped.

**Port Number:** 3310

☒ **Open Firewall port for network access**

**Advanced Configuration**

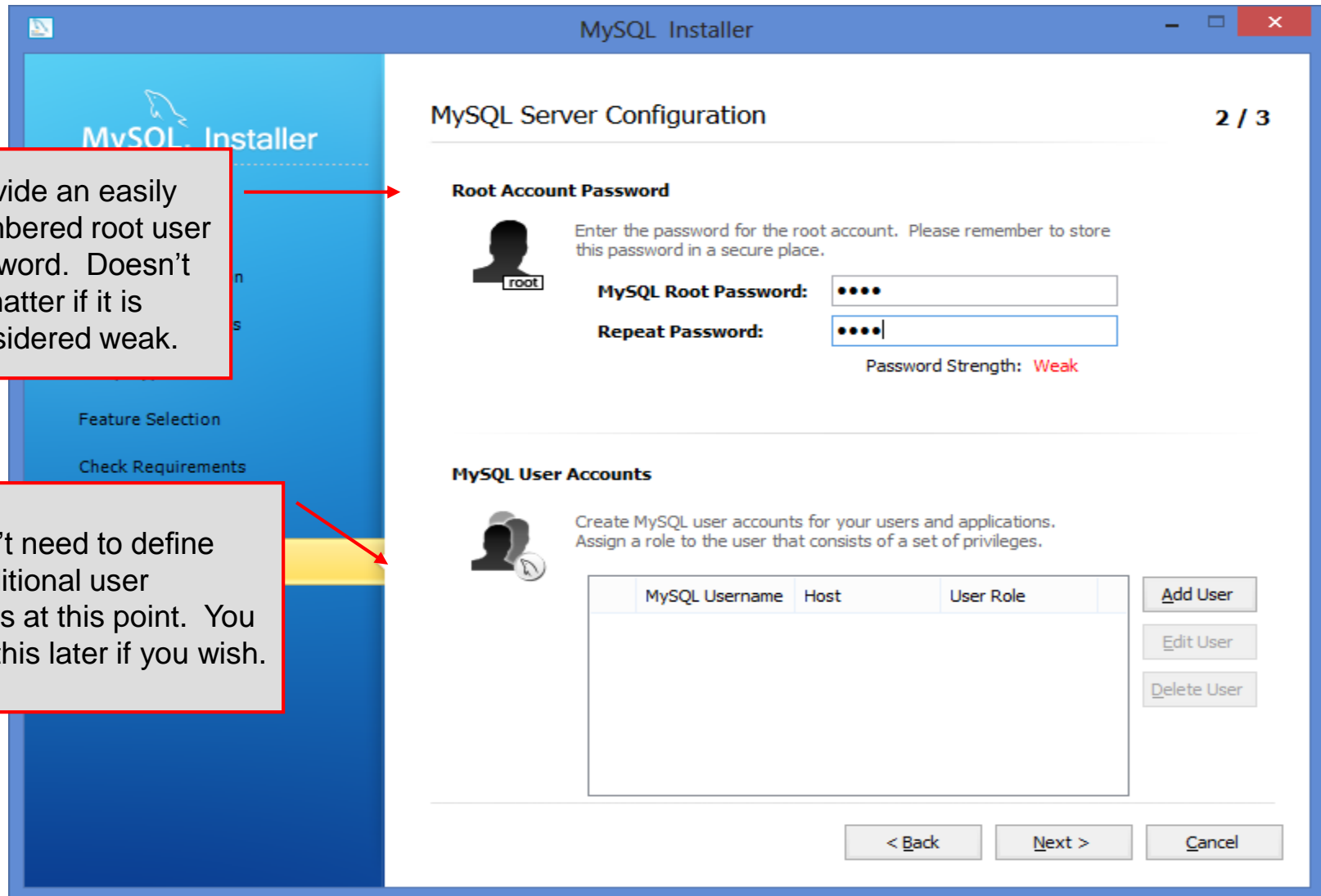
Select the checkbox below to get additional configuration page where you can set advanced options for this server instance.

☐ **Show Advanced Options**

< Back Next > Cancel



# Installing MySQL 5.6.16 (cont.)



The screenshot shows the MySQL Installer window at the 'MySQL Server Configuration' step (2 / 3). The left sidebar has 'Feature Selection' and 'Check Requirements' options. The main area is divided into two sections: 'Root Account Password' and 'MySQL User Accounts'. In the 'Root Account Password' section, there are two password input fields labeled 'MySQL Root Password' and 'Repeat Password', both containing four dots. Below them, the 'Password Strength' is indicated as 'Weak'. In the 'MySQL User Accounts' section, there is a table with columns 'MySQL Username', 'Host', and 'User Role'. To the right of the table are buttons for 'Add User', 'Edit User', and 'Delete User'. At the bottom of the window are '< Back', 'Next >', and 'Cancel' buttons. Two red arrows point from text boxes on the left to the password fields and the 'MySQL User Accounts' section.

MySQL Installer

MySQL Server Configuration 2 / 3

**Root Account Password**

Enter the password for the root account. Please remember to store this password in a secure place.

**MySQL Root Password:** ••••

**Repeat Password:** ••••

Password Strength: **Weak**

**MySQL User Accounts**

Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL Username	Host	User Role
----------------	------	-----------

[Add User](#)  
[Edit User](#)  
[Delete User](#)

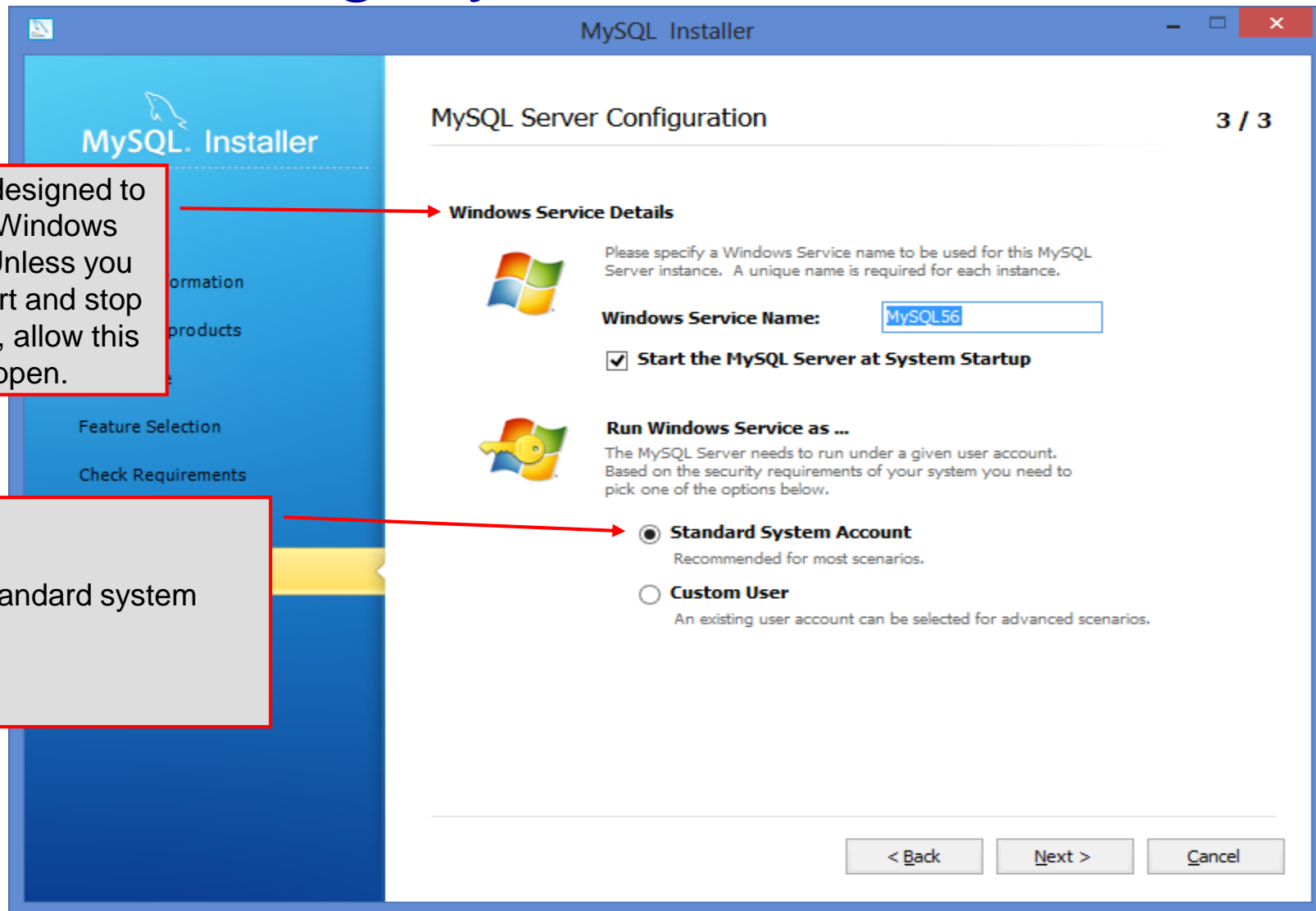
< Back   Next >   Cancel

Provide an easily remembered root user password. Doesn't matter if it is considered weak.

We don't need to define any additional user accounts at this point. You can do this later if you wish.



# Installing MySQL 5.6.16 (cont.)

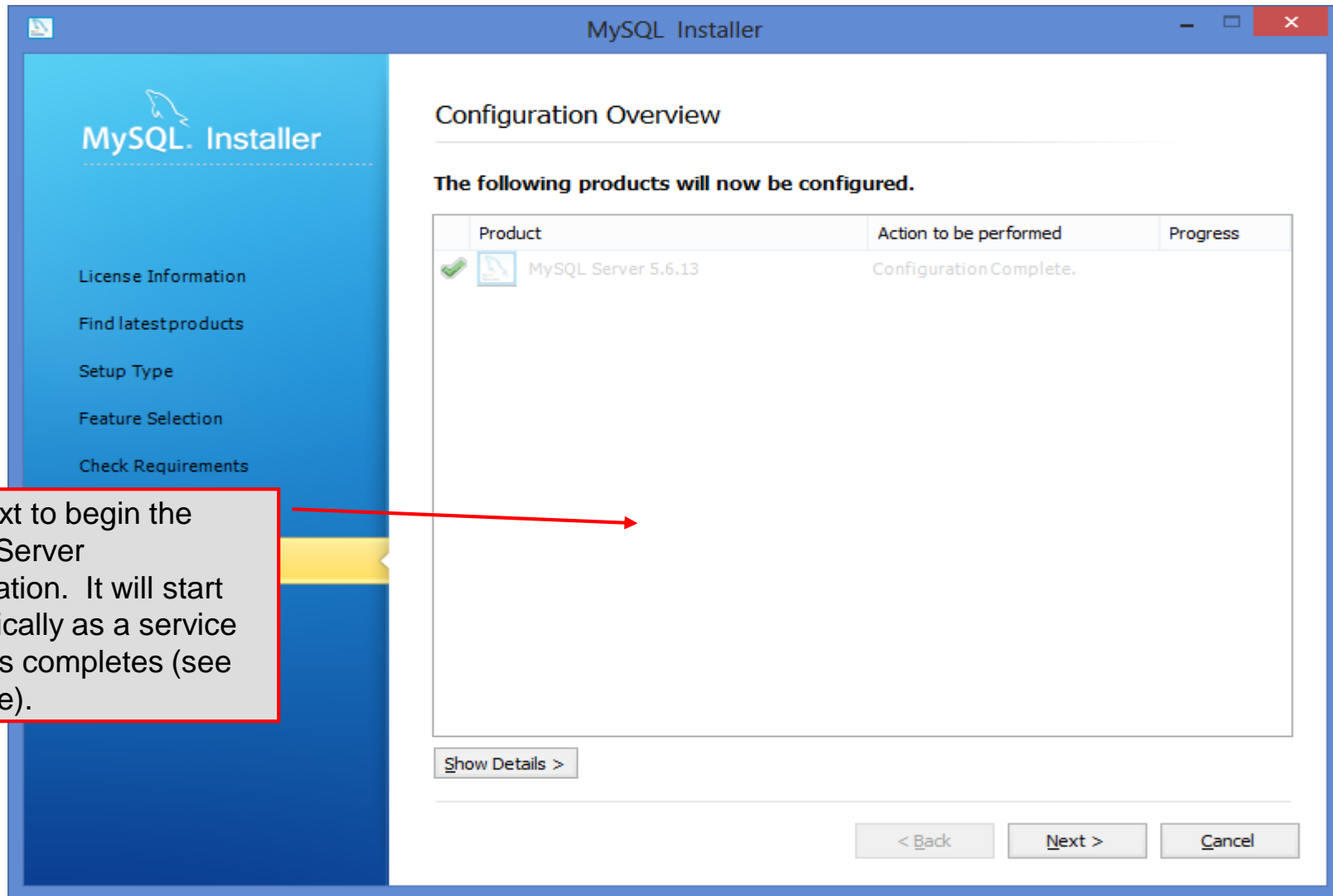


MySQL is designed to run as a Windows service. Unless you want to start and stop it manually, allow this to happen.

Select standard system account.



# Installing MySQL 5.6.16 (cont.)

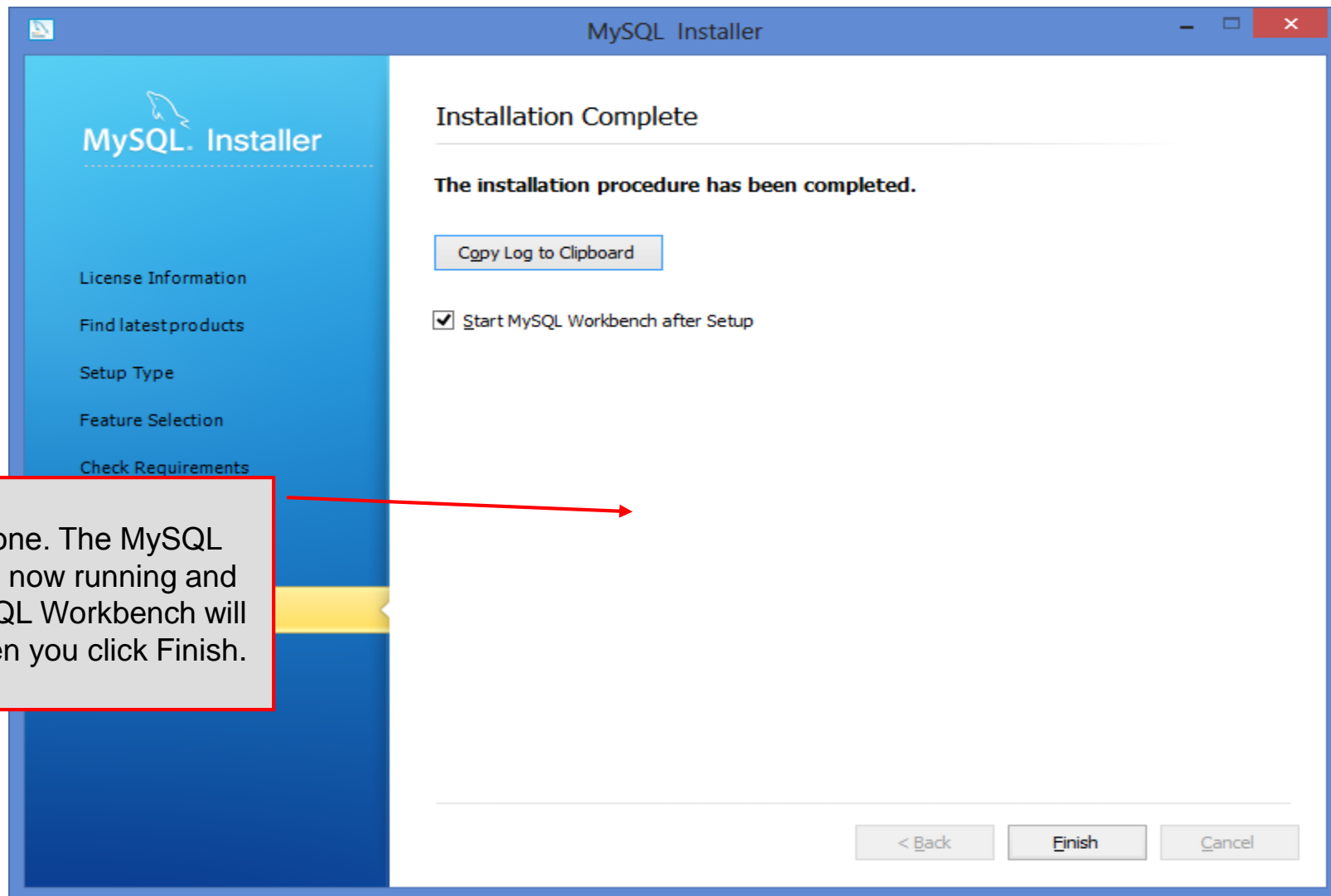


Click Next to begin the MySQL Server configuration. It will start automatically as a service when this completes (see next slide).





# Installing MySQL 5.6.16 (cont.)



You're done. The MySQL Server is now running and the MySQL Workbench will start when you click Finish.



File Edit View Database Tools Scripting Help

## MySQL Connections + 🔍

Filter connections

**Local instance MySQL56**  
root n/a  
localhost:3310

**client1**  
client1 n/a  
127.0.0.1:3310

**client2**  
client2 n/a  
127.0.0.1:3310

This is the MySQL WorkBench. It is basically waiting for you to select a server to connect to at this point. You will have only the one you just created, so your screen will look like this one. Click on the instance and login as the root user.

## Models + 🔍 >

## Shortcuts



MySQL Doc Library



MySQL Utilities



Database Migration



MySQL Bug Reporter



Workbench Blogs



Planet MySQL

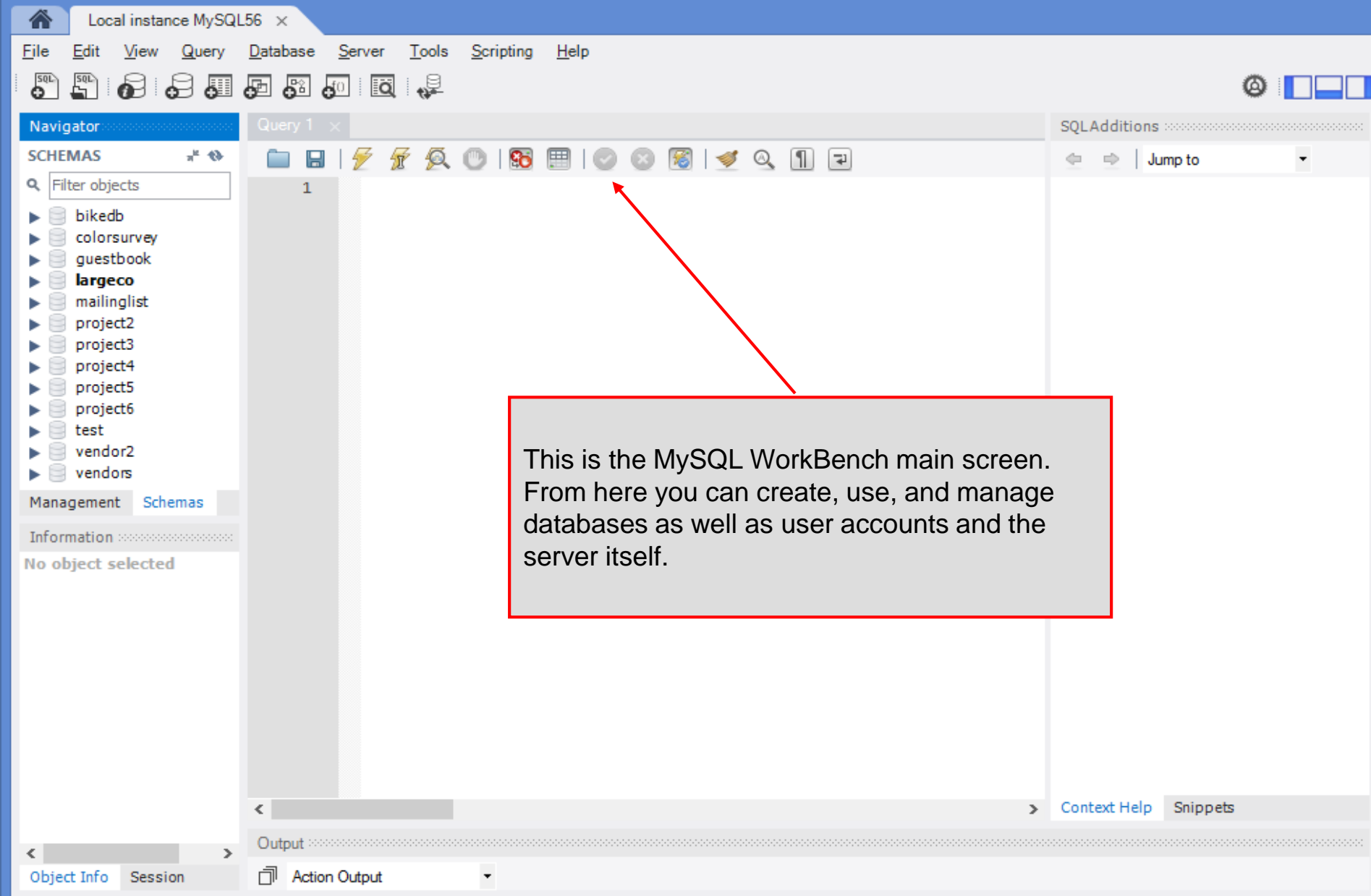


Workbench Forum



Scripting Shell





This is the MySQL WorkBench main screen. From here you can create, use, and manage databases as well as user accounts and the server itself.



The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar shows the Navigator with a tree view of Schemas, including a database named 'bikedb' with tables, views, stored procedures, and functions. The main editor window displays a script file named 'bikedbscript' with the following SQL code:

```
1 # Script file for creating the bikedb that is used in many of the S
2 # examples for the CNT 4714 notes
3
4 drop database if exists bikedb;
5
6 create database bikedb;
7
8 use bikedb;
9
10 create table bikes (
11     bikename varchar(30) not null,
12     size int(2),
13     color varchar(15),
14     cost int(6),
15     purchased date,
16     mileage int(6),
17     primary key (bikename)
18 );
19
20 insert into bikes values ('Colnago', 58, 'celeste', 4800, '1998-01-01', 0, 0);
21 insert into bikes values ('Bianchi Evolution 3', 58, 'celeste', 4800, '1998-01-01', 0, 0);
22 insert into bikes values ('Eddy Merckx Molteni', 58, 'orange', 5100, '1998-01-01', 0, 0);
23 insert into bikes values ('Eddy Merckx Domo', 58, 'blue/black', 5300, '1998-01-01', 0, 0);
24 insert into bikes values ('Battaglin Carrera', 60, 'red/white', 4000, '1998-01-01', 0, 0);
25 insert into bikes values ('Gianni Motta Personal', 59, 'red/green', 4400, '1998-01-01', 0, 0);
26 insert into bikes values ('Gios Torino Super', 60, 'blue', 2000, '1998-01-01', 0, 0);
27 insert into bikes values ('Schwinn Paramount P14', 60, 'blue', 1800, '1998-01-01', 0, 0);
28 insert into bikes values ('Bianchi Corse Evo 4', 58, 'celeste', 5700, '1998-01-01', 0, 0);
29 insert into bikes values ('Colnago Superissimo', 59, 'red', 3800, '1996-01-01', 0, 0);
30 insert into bikes values ('Ridley Damocles', 58, 'blue/black', 7500, '1998-01-01', 0, 0);
31 insert into bikes values ('Bianchi Infinito', 58, 'celeste', 8900, '1998-01-01', 0, 0);
```

A red arrow points from the 'File' menu to the 'Open SQL Script' icon in the toolbar. A text box with a red border contains the following text:

This is the MySQL WorkBench main screen shown with the script file available on the course website for you to play around with loaded. This script will create and populate a small database. Under File, Select Open SQL Script. Navigate to where you placed the script file and open it into this editing window.



Local instance MySQL56 x

File Edit View Query Database Server Tools Scripting Help



Navigator

Query 1 bikedbscript x

SQLAdditions

Jump to

## MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

## INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

## SCHEMAS

Filter objects

- bikedb
- test

Information

No object selected

Object Info Session

```

1 # Script file for creating the bikedb that is used in man
2 # examples for the COP 4710 MySQL notes
3
4 drop database if exists bikedb;
5
6 create database bikedb;
7
8 use bikedb;
9

```

The script executed, which created and populated the database, then executed the query at the bottom of the script. Shown in the results window is the execution results of that query. Shown in the output window is the MySQL Server output

Result Set Filter:



Edit



Export/Import



Wrap Cell Content



IA

bikename	color	price	total_miles
Gios Torino Super	blue	2000	9000
Schwinn Paramount P14	blue	1800	200
NULL	NULL	NULL	NULL

bikes 1 bluebikes 2 x

Apply

Cancel

Context Help

Snippets

Output



Action Output

	Time	Action	Message	Duration / Fetch
21	15:29:14	insert into bluebikes select bikename, color, cost, mileage from bikes where c...	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.031 sec
22	15:29:14	select * from bluebikes LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec



Local instance MySQL56 x

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

SCHEMAS

Filter objects

- bikedb
- test

Query 1 x bikedbscript

```
1 select * from bikes
2 where color = "red"
```

Result Set Filter:

	bikename	size	color	cost	purchased	mileage
▶	Colnago Superissimo	59	red	3800	1996-03-01	13000
*	NULL	NULL	NULL	NULL	NULL	NULL

bikes 1 x

Output

Action Output

	Time	Action	Message	Duration / Fetch
✓	22 15:29:14	select * from bluebikes LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
✓	23 15:32:30	select * from bikes where color = "red" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

SQLAdditions

Jump to

A different query executed against the same database instance.



# Running MySQL 5.6.16

- 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 5.6 Command Line 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 5.6.16 (cont.)

```
MySQL 5.6 Command Line Client - Unicode
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.6.16 MySQL Community Server (GPL)
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> status;
-----
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe  Ver 14.14 Distrib 5.6.16,
  for Win64 (x86_64)

Connection id:          4
Current database:
Current user:           root@localhost
SSL:                    Not in use
Using delimiter:        ;
Server version:         5.6.16 MySQL Community Server (GPL)
Protocol version:       10
Connection:             localhost via TCP/IP
Server characterset:    utf8
Db characterset:        utf8
Client characterset:    utf8
Conn. characterset:     utf8
TCP port:               3310
Uptime:                 3 min 26 sec

Threads: 3  Questions: 84  Slow queries: 0  Opens:
les: 63  Queries per second avg: 0.407
-----

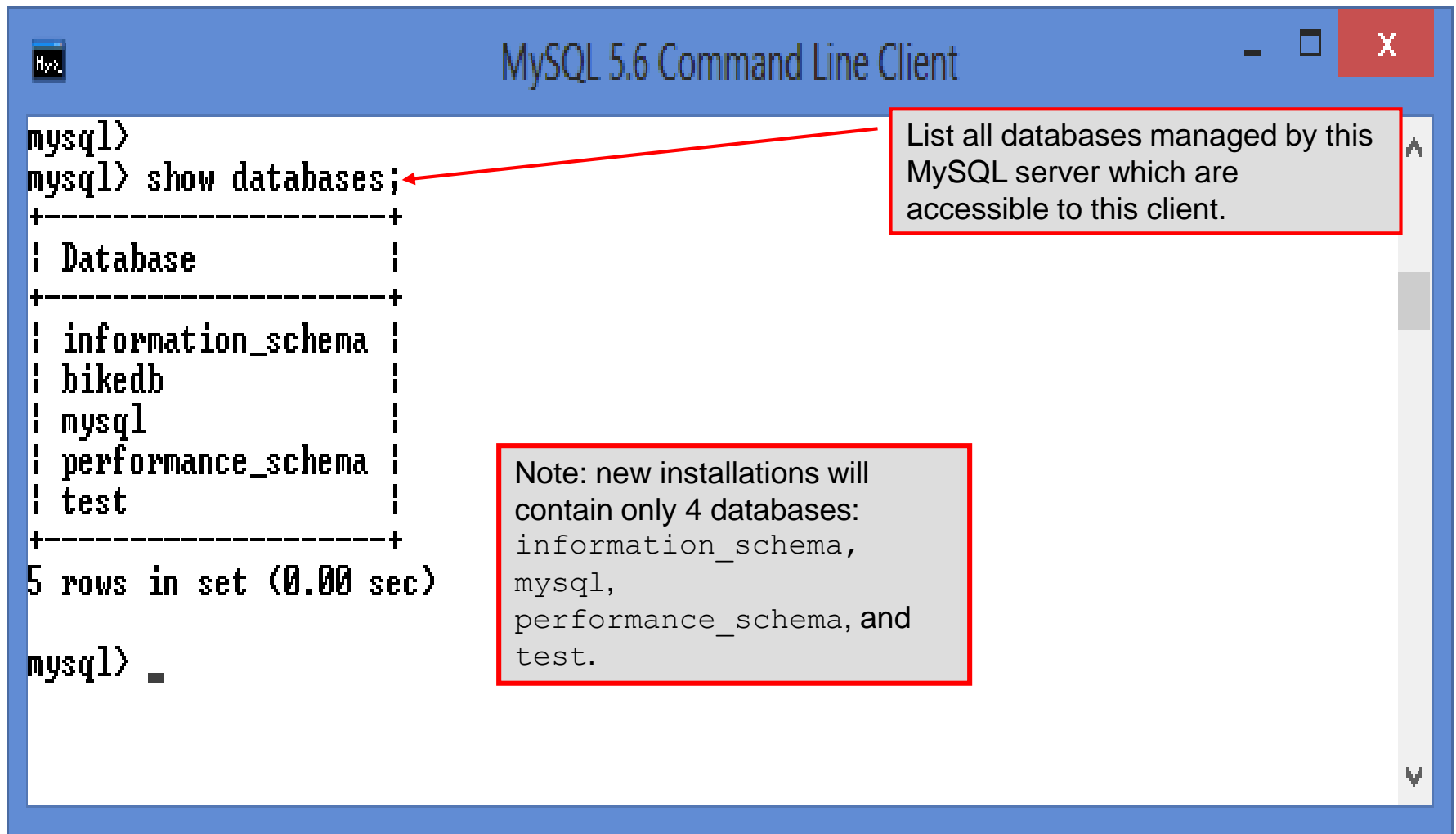
mysql> _
```

Server version

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



# Running MySQL 5.6.16 (cont.)



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

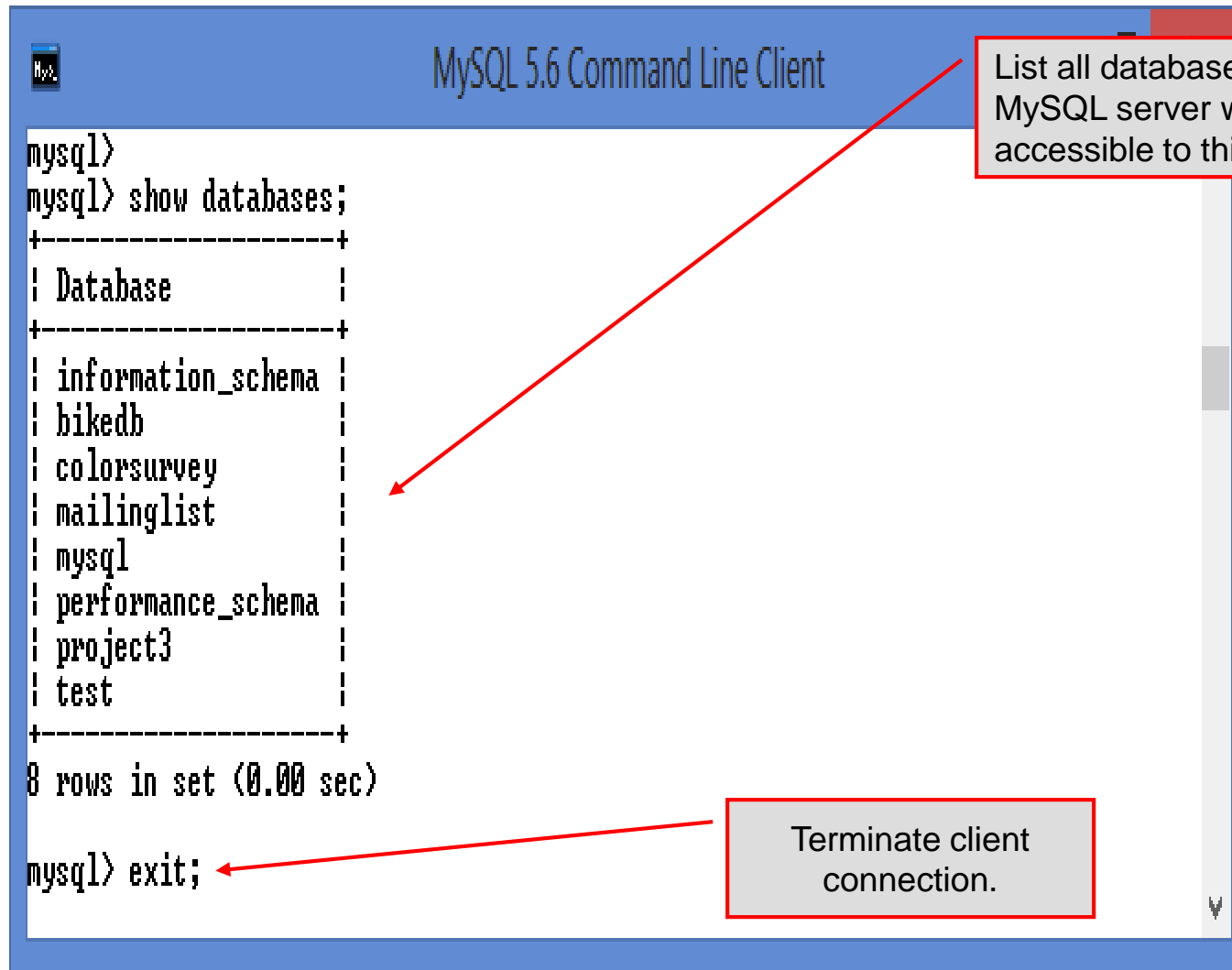
mysql> _
```

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

Note: new installations will contain only 4 databases: information\_schema, mysql, performance\_schema, and test.



# Running MySQL 5.6.16 (cont.)



The screenshot shows a terminal window titled "MySQL 5.6 Command Line Client". The prompt is "mysql>". The command "show databases;" has been entered. The output is a table with one column, "Database", listing the following databases: information\_schema, bikedb, colorsurvey, mailinglist, mysql, performance\_schema, project3, and test. Below the table, it says "8 rows in set (0.00 sec)". The prompt "mysql> exit;" is shown at the bottom. A red arrow points from a text box on the right to the list of databases. Another red arrow points from a text box at the bottom right to the "exit;" command.

```
mysql>
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb          |
| colorsurvey     |
| mailinglist     |
| mysql           |
| performance_schema |
| project3        |
| test            |
+-----+
8 rows in set (0.00 sec)

mysql> exit;
```

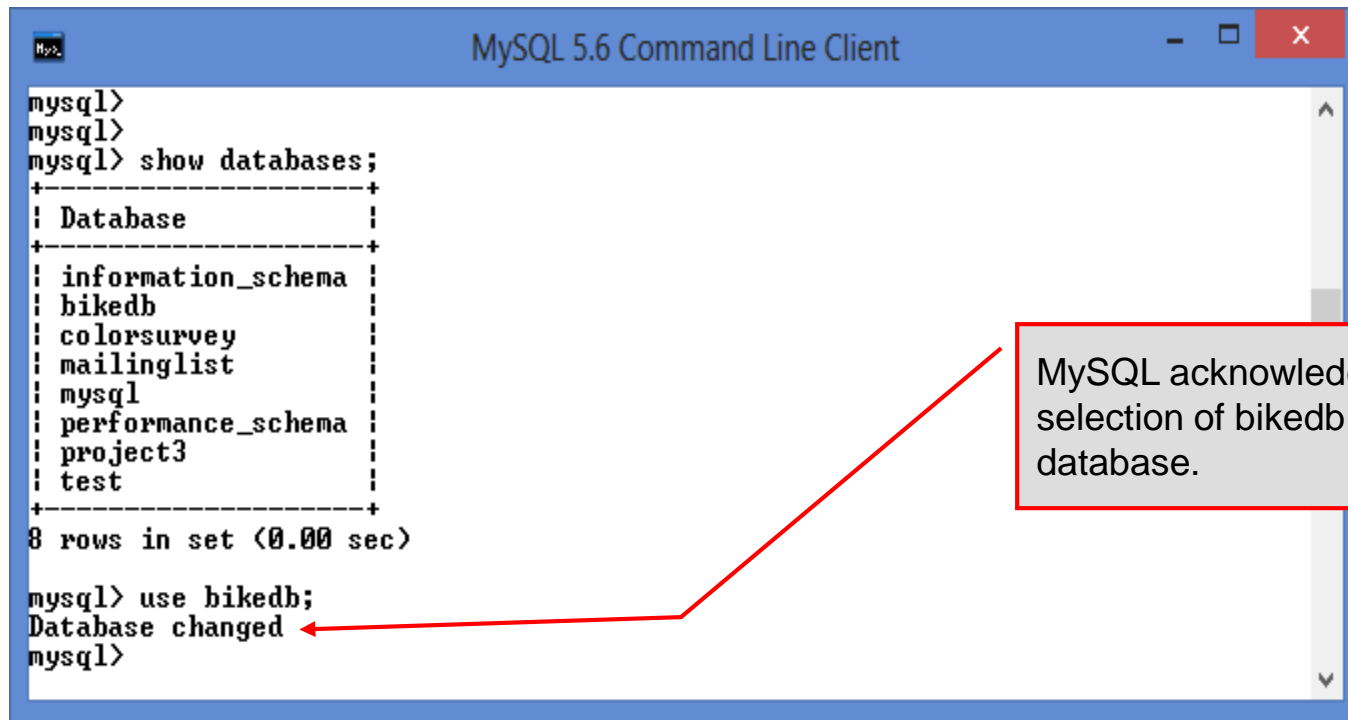
List all databases managed by this MySQL server which are accessible to this client.

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.



```
mysql>
mysql>
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb      |
| colorsurvey |
| mailinglist |
| mysql       |
| performance_schema |
| project3    |
| test        |
+-----+
8 rows in set (0.00 sec)

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

MySQL acknowledges selection of bikedb database.



Local instance MySQL56 x

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

SCHEMAS

Filter objects

- bikedb
  - Tables
  - Views
  - Stored Procedures
  - Functions
- coloursurvey

Information

Schema: bikedb

Query 1 bikedbscript project3dbscript coloursurvey script mailing list script x

```

1 # SQL commands to create and populate the MySQL database for
2 # CNT 4714 - Spring 2012
3 #
4 # delete the database if it already exists
5 drop database if exists mailinglist;
6
7 #create a new database named mailinglist
8 create database mailinglist;
9
10 #switch to the new database
11 use mailinglist;
12
13 #create the schemas for the four rel
14 create table contacts (
15     ID integer unsigned zerofill auto_increment not null,
16     LastName varchar(30),
17     FirstName varchar(30),
18     Email varchar(30),
19     Phone varchar(14),
20     Magazine varchar(60),
21     OS varchar(30),
22     primary key (ID)
23 );
24
25

```

SQLAdditions

Jump to

Output

Action Output

	Time	Action	Message	Duration / Fetch
✓ 123	15:39:11	use mailinglist	0 row(s) affected	0.000 sec
✓ 124	15:39:11	create table contacts ( ID integer unsigned zerofill auto_increment not null, ...	0 row(s) affected	0.453 sec

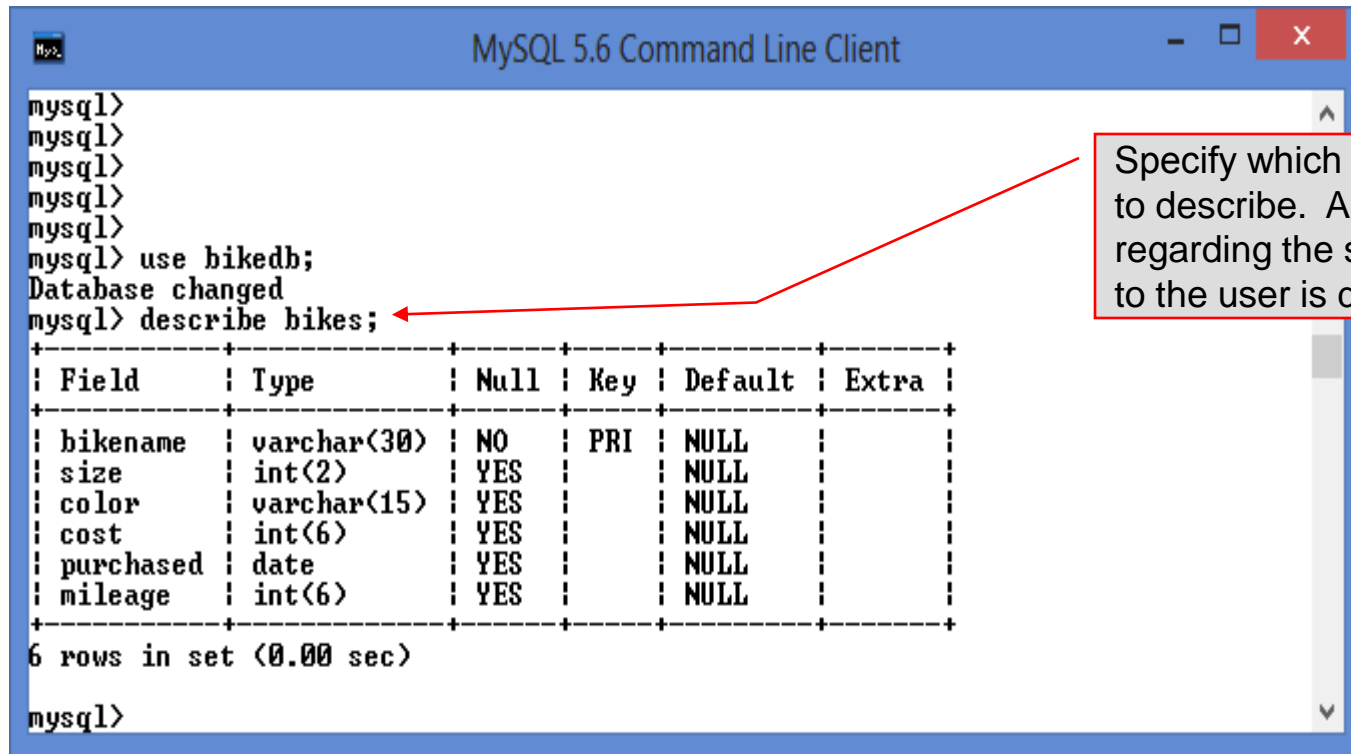
Object Info Session

In the Workbench, you select a database by clicking on it.



# 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>
mysql>
mysql>
mysql>
mysql>
mysql> use bikedb;
Database changed
mysql> describe bikes;
```

Field	Type	Null	Key	Default	Extra
bikename	varchar(30)	NO	PRI	NULL	
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 (0.00 sec)

mysql>
```

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



Local instance MySQL56 x

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

SCHEMAS

Filter objects

- Columns
  - bikename
  - size
  - color
  - cost
  - purchased

Information

Column: **bikename**

Collation: utf8\_general\_ci

Definition:

**bikename** varchar(30) PK

Object Info Session

Query 1 bikedbscript project3dbscript colorsurvey script mailing list script bikedbscript x

```

1 # Script file for creating the bikedb that is used in many of the SQL and MySQL
2 # examples for the COP 4710 MySQL notes
3
4 drop database if exists bikedb;
5
6 create database bikedb;
7
8 use bikedb;
9
10 create table bikes (
11     bikename varchar(30) not null,
12     size int(2),
13     color varchar(15),
14     cost int(6),
15     purchased date,
16     mileage int(6),
17     primary key (bikename)
18 );
19
20 insert into bikes values ('Colnago Dream Rabobank',60,'blue/orange',5500,'2002-07-07',4300);
21 insert into bikes values ('Bianchi Evolution 3',58,'celeste',4800,'2003-11-12',2000);
22 insert into bikes values ('Eddy Merckx Molteni',58,'orange',5100,'2004-08-12',0);
23 insert into bikes values ('Eddy Merckx Domo',58,'blue/black',5300,'2004-02-02',0);
24 insert into bikes values ('Battaglin Carrera',60,'red/white',4000,'2001-03-10',11200);
25 insert into bikes values ('Gianni Motta Personal',59,'red/green',4400,'2000-05-01',8700);
26 insert into bikes values ('Giant Tornado Special',60,'black',3000,'1999-11-01',0000);

```

SQLAdditions

Jump to

Output

Action Output

	Time	Action	Message	Duration / Fetch
✓ 123	15:39:11	use mailinglist	0 row(s) affected	0.000 sec
✓ 124	15:39:11	create table contacts ( ID integer unsigned zerofill auto_increment not null, ...	0 row(s) affected	0.453 sec

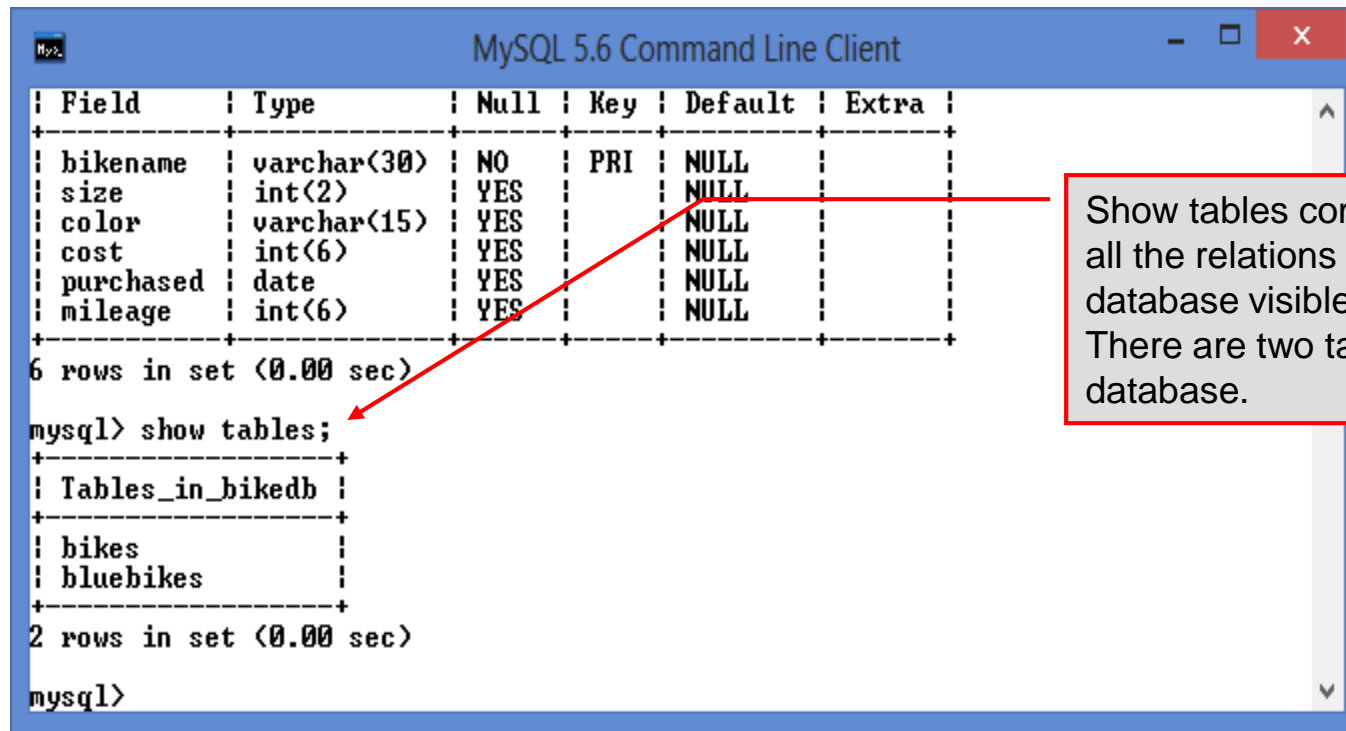
To see the details of a table's schema, use the icons to select more or less detail. Details shown in information area below.





# 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.



The screenshot shows the MySQL 5.6 Command Line Client window. The top part displays the structure of a table with the following columns: Field, Type, Null, Key, Default, and Extra. The table has 6 rows in the set, taking 0.00 seconds to display. The fields are: bikename (varchar(30), NO, PRI, NULL), size (int(2), YES, NULL), color (varchar(15), YES, NULL), cost (int(6), YES, NULL), purchased (date, YES, NULL), and mileage (int(6), YES, NULL). Below this, the command `mysql> show tables;` is entered, and the output shows two tables: `Tables_in_bikedb`, `bikes`, and `bluebikes`. The output indicates 2 rows in the set, taking 0.00 seconds to display. The prompt `mysql>` is shown at the bottom.

Field	Type	Null	Key	Default	Extra
bikename	varchar(30)	NO	PRI	NULL	
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 (0.00 sec)

```
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.

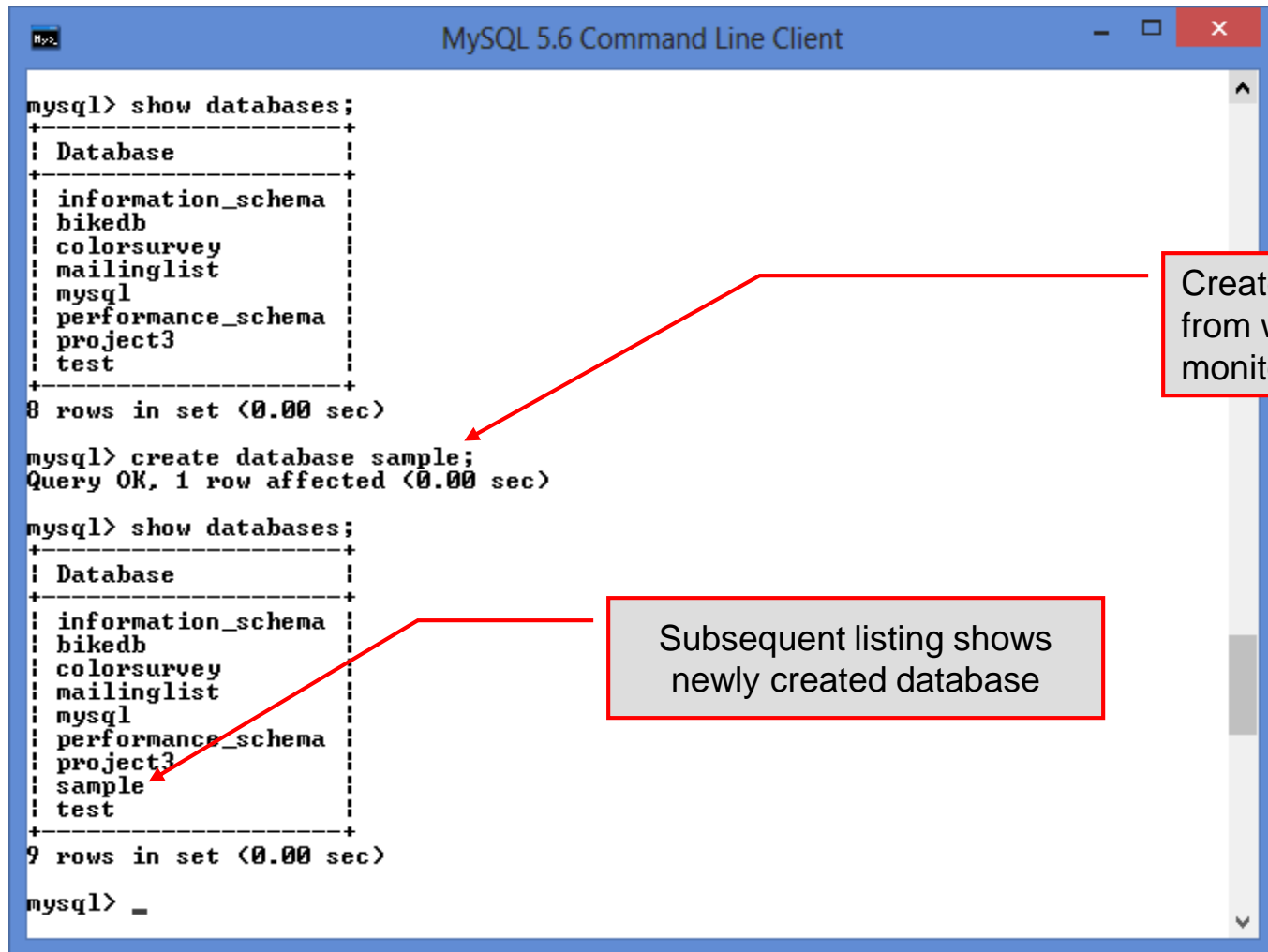
```
MySQL 5.6 Command Line Client
+-----+
2 rows in set (0.00 sec)
mysql> select * from bikes;
+-----+-----+-----+-----+-----+-----+
| bikename          | size | color          | cost | purchased | mileage |
+-----+-----+-----+-----+-----+-----+
| Battaglin Carrera | 60   | red/white      | 4000 | 2001-03-10 | 11200   |
| Bianchi Corse Evo 4 | 58   | celeste        | 5700 | 2004-12-02 | 300     |
| Bianchi Evolution 3 | 58   | celeste        | 4800 | 2003-11-12 | 2000    |
| Bianchi Infinito   | 58   | celeste        | 8900 | 2011-07-14 | 0       |
| BMC SLC01 - Swiss  | 58   | red/black/white | 8000 | 2010-06-23 | 0       |
| Colnago Dream Rabobank | 60   | blue/orange    | 5500 | 2002-07-07 | 4300    |
| Colnago Superissimo | 59   | red            | 3800 | 1996-03-01 | 13000   |
| Eddy Merckx Domo   | 58   | blue/black     | 5300 | 2004-02-02 | 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    |
```

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>`



The screenshot shows the MySQL 5.6 Command Line Client window. The first command executed is `mysql> show databases;`, which returns a list of 8 databases: `information_schema`, `bikedb`, `colorsurvey`, `mailinglist`, `mysql`, `performance_schema`, `project3`, and `test`. The output is displayed in a table format with 8 rows in set. The second command is `mysql> create database sample;`, which returns `Query OK, 1 row affected (0.00 sec)`. The third command is `mysql> show databases;`, which returns a list of 9 databases, including the newly created `sample` database. The output is displayed in a table format with 9 rows in set. The prompt `mysql> _` is shown at the bottom.

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb         |
| colorsurvey    |
| mailinglist     |
| mysql          |
| performance_schema |
| project3       |
| test          |
+-----+
8 rows in set (0.00 sec)

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

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb         |
| colorsurvey    |
| mailinglist     |
| mysql          |
| performance_schema |
| project3       |
| sample        |
| test          |
+-----+
9 rows in set (0.00 sec)

mysql> _
```

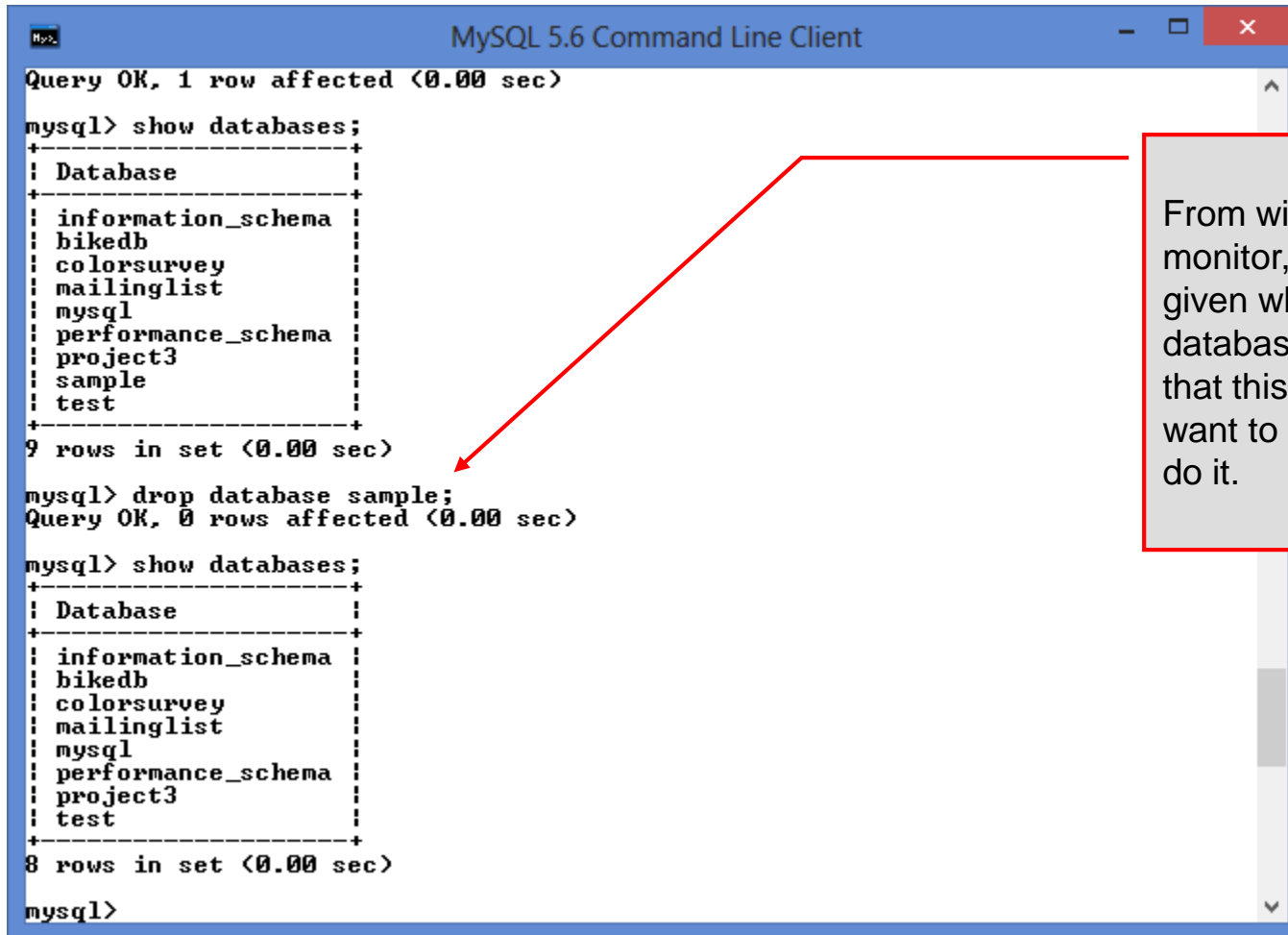
Create new database from within MySQL monitor.

Subsequent listing shows newly created database



# Dropping a Database in MySQL

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



The screenshot shows the MySQL 5.6 Command Line Client interface. It displays two queries and their results. The first query is `show databases;`, which returns a list of 9 databases: `information_schema`, `bikedb`, `colorsurvey`, `mailinglist`, `mysql`, `performance_schema`, `project3`, `sample`, and `test`. The second query is `drop database sample;`, which returns `Query OK, 0 rows affected (0.00 sec)`. A third query, `show databases;`, is shown below, returning a list of 8 databases, with `sample` removed. A red arrow points from the `sample` database in the first list to the `drop database sample;` command.

```
MySQL 5.6 Command Line Client
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb |
| colorsurvey |
| mailinglist |
| mysql |
| performance_schema |
| project3 |
| sample |
| test |
+-----+
9 rows in set (0.00 sec)

mysql> drop database sample;
Query OK, 0 rows affected (0.00 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bikedb |
| colorsurvey |
| mailinglist |
| mysql |
| performance_schema |
| project3 |
| test |
+-----+
8 rows in set (0.00 sec)

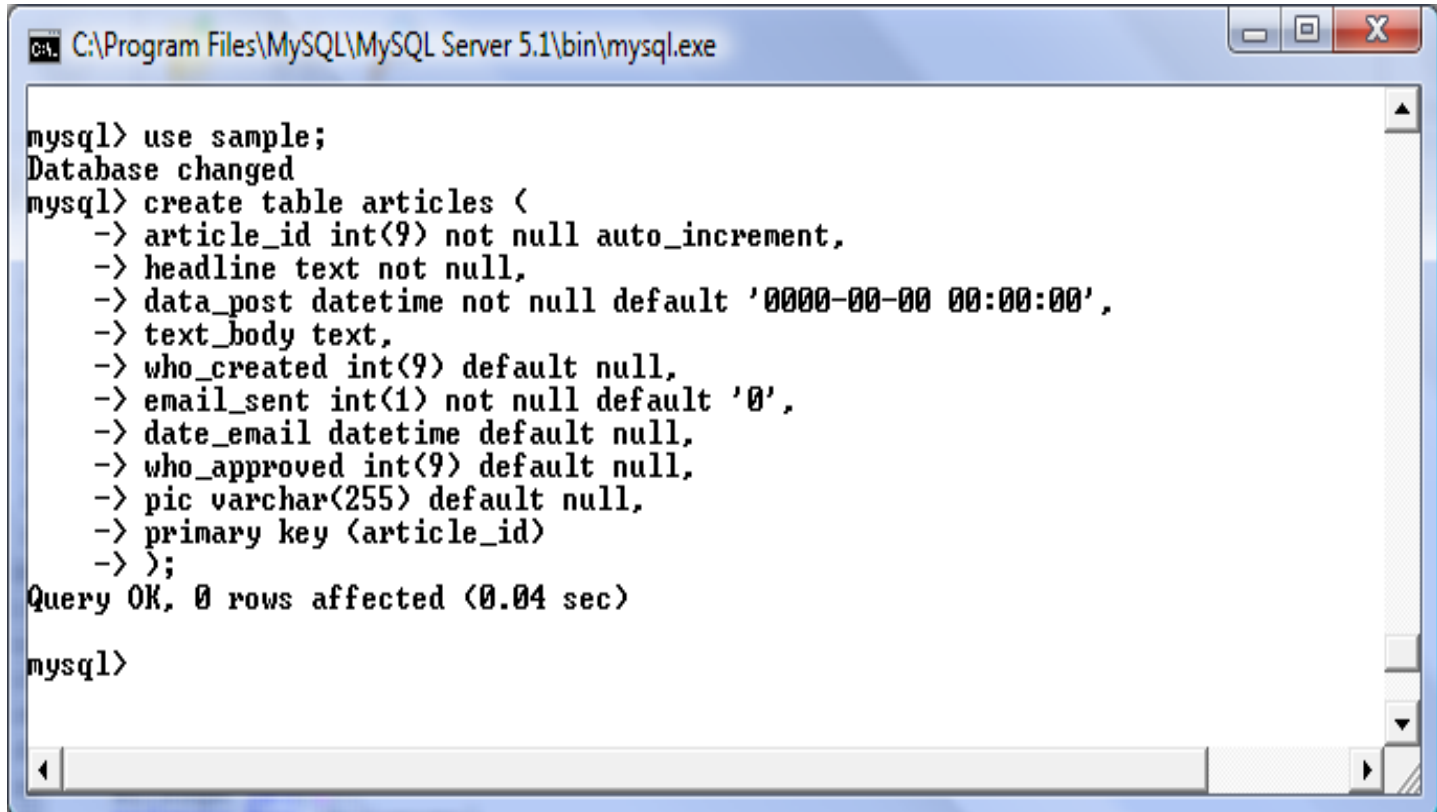
mysql>
```

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.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

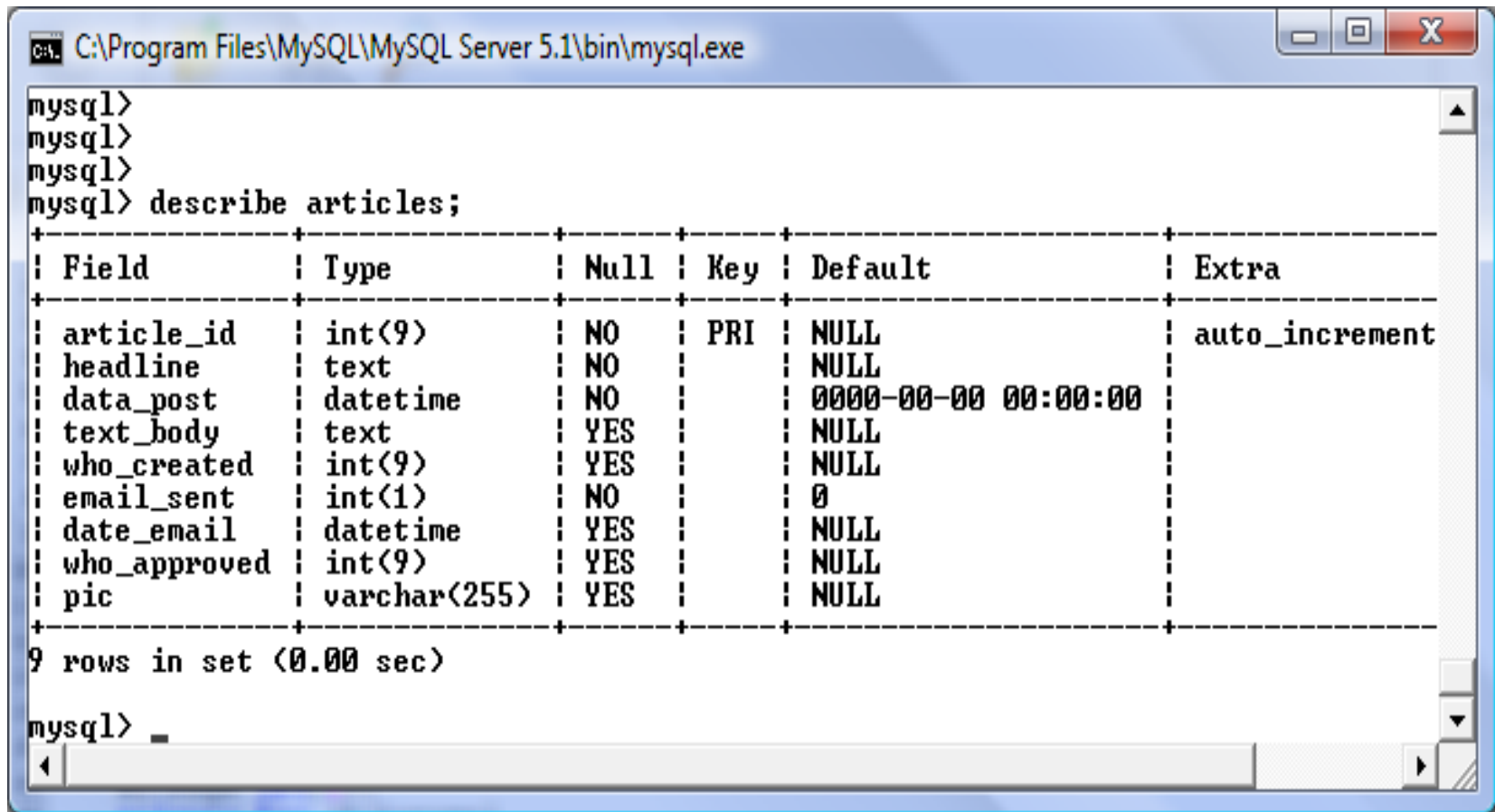
mysql> use sample;
Database changed
mysql> create table articles (
  -> article_id int(9) not null auto_increment,
  -> headline text not null,
  -> data_post datetime not null default '0000-00-00 00:00:00',
  -> text_body text,
  -> who_created int(9) default null,
  -> email_sent int(1) not null default '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 (0.04 sec)

mysql>
```



# Manipulating Tables in MySQL (cont.)

Screen shot that describes the newly created table.



The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe". The user has entered the command "describe articles;" and the output is displayed as a table with 6 columns: Field, Type, Null, Key, Default, and Extra. The table has 9 rows of data. The first row, "article\_id", is the primary key and is auto-incrementing. The other fields are: "headline" (text), "data\_post" (datetime), "text\_body" (text), "who\_created" (int), "email\_sent" (int), "date\_email" (datetime), "who\_approved" (int), and "pic" (varchar).

```
mysql>
mysql>
mysql>
mysql> describe articles;
```

Field	Type	Null	Key	Default	Extra
article_id	int(9)	NO	PRI	NULL	auto_increment
headline	text	NO		NULL	
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 (0.00 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

```
C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe

+-----+-----+-----+-----+-----+-----+
| Battaglin Carrera      | 60 | red/white | 4000 | 2001-03-10 | 11200 |
| Bianchi Corse Evo 4    | 58 | celeste   | 5700 | 2004-12-02 | 300   |
| Bianchi Evolution 3    | 58 | celeste   | 4800 | 2003-11-12 | 2000  |
| Bianchi Infinito       | 58 | celeste   | 8900 | 2011-07-14 | 0     |
| BMC SLC01 - Swiss      | 58 | red/black/white | 8000 | 2010-06-23 | 430   |
| Colnago Dream Rabobank | 60 | blue/orange | 5500 | 2002-07-07 | 13000 |
| Colnago Superissimo    | 59 | red       | 3800 | 1996-03-01 | 0     |
| Eddy Merckx Domo       | 58 | blue/black | 5300 | 2004-02-02 | 0     |
| Eddy Merckx Molteni    | 58 | orange    | 5100 | 2004-08-12 | 0     |
| Gianni Motta Personal  | 59 | red/green  | 4400 | 2000-05-01 | 870   |
| Gios Torino Super      | 60 | blue      | 2000 | 1998-11-08 | 900   |
| Ridley Damocles        | 58 | blue/black | 7500 | 2008-06-27 | 0     |
| Ridley X-Fire          | 58 | red/white  | 7500 | 2011-09-01 | 0     |
| Schwinn Paramount P14  | 60 | blue      | 1800 | 1992-03-01 | 20    |
+-----+-----+-----+-----+-----+
14 rows in set (0.00 sec)

mysql> create temporary table celestebikes
-> select *
-> from bikes
-> where color = "celeste";
Query OK, 3 rows affected (0.11 sec)
Records: 3  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 | 2004-12-02 | 300     |
| Bianchi Evolution 3 | 58   | celeste | 4800 | 2003-11-12 | 2000    |
| Bianchi Infinito   | 58   | celeste | 8900 | 2011-07-14 | 0       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

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 InnoDB for MySQL versions 5.5.20.x.



# 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
<code>add [column] <i>column_declaration</i> [first   after <i>column_name</i>]</code>	Add a column to the table
<code>alter [column] <i>column_name</i> {set default <i>literal</i>   drop default}</code>	Specify new default value for a column or remove old default
<code>change [column] <i>column_name</i> <i>column_declaration</i></code>	Modify column declaration with renaming of column
<code>modify [column] <i>column_declaration</i></code>	Modify column declaration without renaming column
<code>drop [column] <i>column_name</i></code>	Drop a column and all data contained within it.
<code>rename [as] <i>new_table_name</i></code>	Rename a table
<code><i>table_options</i></code>	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.

The screenshot shows a MySQL command window with the following content:

```
mysql> describe bikes;
```

Field	Type	Null	Key	Default	Extra
bikename	varchar(30)	NO	PRI	NULL	
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 (0.00 sec)

```
mysql> alter table bikes  
-> add column races_won int(3) default 0;
```

Query OK, 10 rows affected (0.05 sec)  
Records: 10 Duplicates: 0 Warnings: 0

```
mysql> describe bikes;
```

Field	Type	Null	Key	Default	Extra
bikename	varchar(30)	NO	PRI	NULL	
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 (0.00 sec)

```
mysql>
```

Annotations:

- A red box labeled "Schema of bikes before alteration" points to the first table.
- A blue box labeled "There are 10 rows affected because this table currently contains 10 tuples (rows) and the new attribute has been added to all rows." points to the "alter table" command.
- A green box labeled "Bikes table after the addition of a new column named races\_won" points to the second table.



# 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.

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> select * from bikes;
+-----+-----+-----+-----+-----+-----+-----+
| bikename      | size | color   | cost  | purchased | mileage | races_won |
+-----+-----+-----+-----+-----+-----+-----+
| Colnago Dream Rabobank | 60   | blue/orange | 5500  | 2002-07-07 | 4300   | 0         |
| Bianchi Evolution 3    | 58   | celeste   | 4800  | 2003-11-12 | 2000   | 0         |
| Eddy Merckx Molteni    | 58   | orange    | 5100  | 2004-08-12 | 0       | 0         |
| Eddy Merckx Domo       | 58   | blue/black | 5300  | 2004-02-02 | 0       | 0         |
| Battaglin Carrera      | 60   | red/white  | 4000  | 2001-03-10 | 11200  | 0         |
| Gianni Motta Personal  | 59   | red/green  | 4400  | 2000-05-01 | 8700   | 0         |
| Gios Torino Super      | 60   | blue      | 2000  | 1998-11-08 | 9000   | 0         |
| Schwinn Paramount P14  | 60   | blue      | 1800  | 1992-03-01 | 200    | 0         |
| Bianchi Corse Evo 4    | 58   | celeste   | 5700  | 2004-12-02 | 300    | 0         |
| Colnago Superissimo    | 59   | red       | 3800  | 1996-03-01 | 13000  | 0         |
+-----+-----+-----+-----+-----+-----+-----+

10 rows in set (0.00 sec)

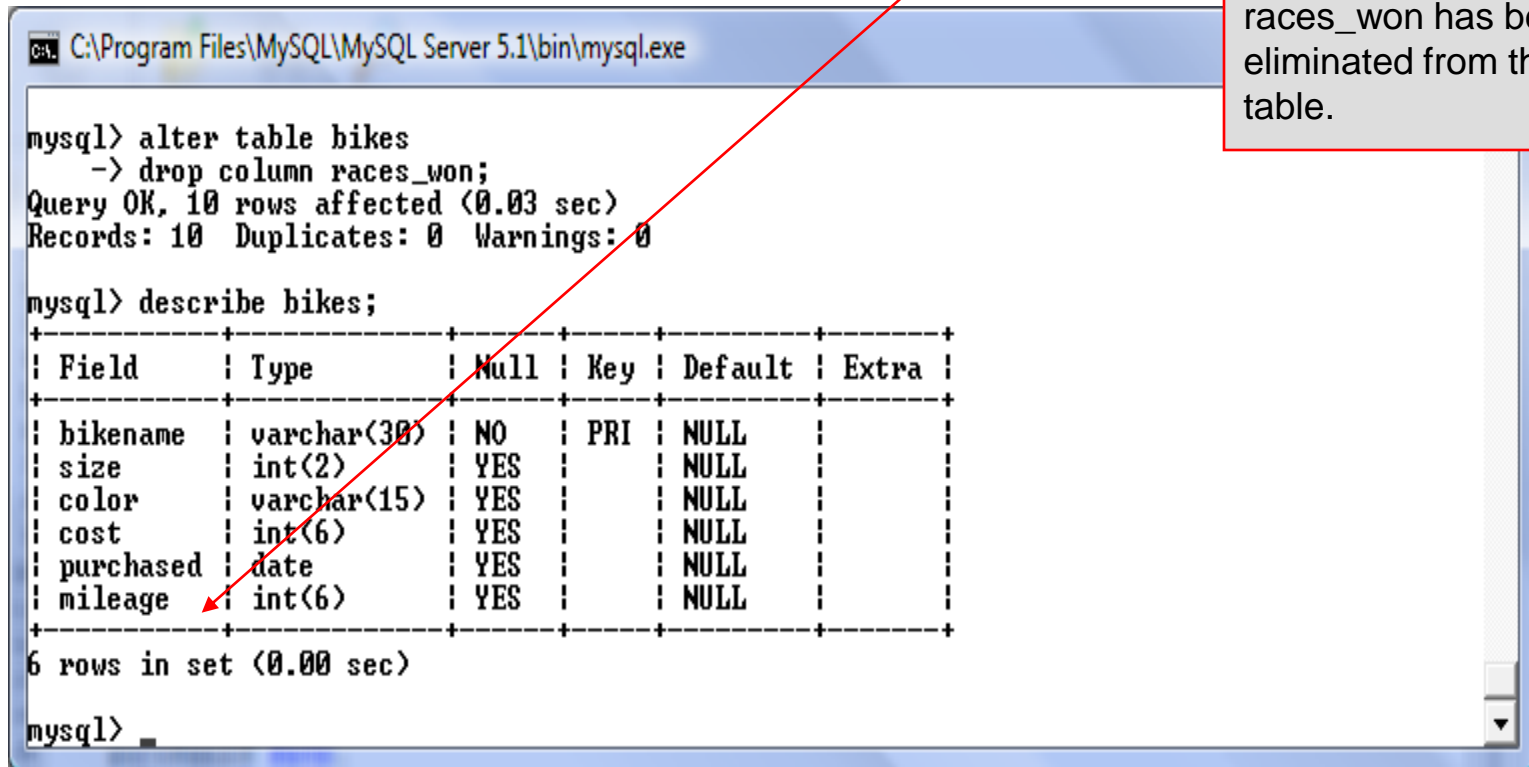
mysql>
```

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.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

mysql> alter table bikes
-> drop column races_won;
Query OK, 10 rows affected (0.03 sec)
Records: 10 Duplicates: 0 Warnings: 0

mysql> describe bikes;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| bikename   | varchar(30)   | NO   | PRI | NULL    |       |
| 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 (0.00 sec)

mysql>
```

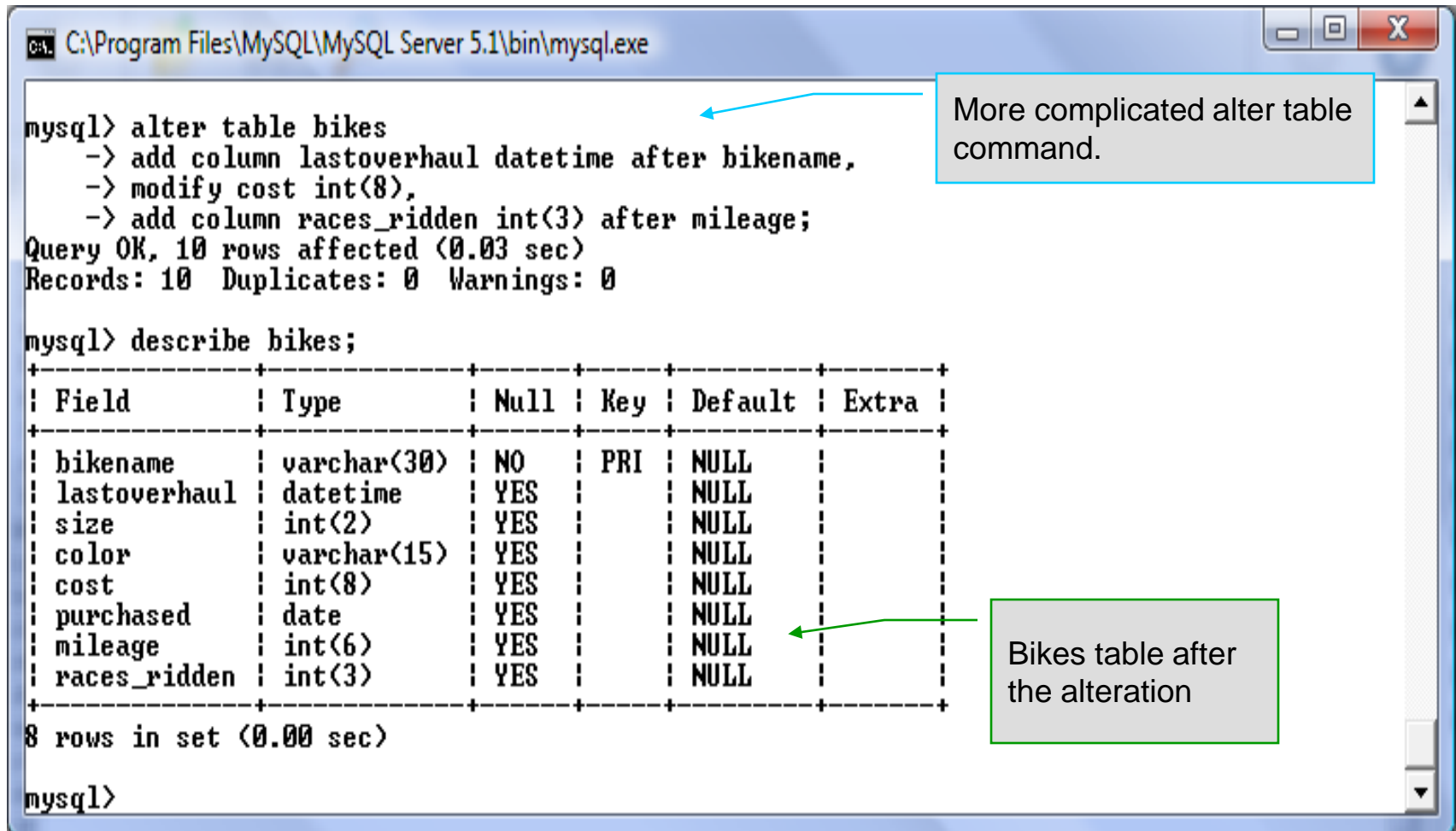
The attribute  
races\_won has been  
eliminated from the  
table.





# Altering A Table (cont.)

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



The screenshot shows a MySQL command prompt window with the following text:

```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

mysql> alter table bikes
  -> add column lastoverhaul datetime after bikename,
  -> modify cost int(8),
  -> add column races_ridden int(3) after mileage;
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> describe bikes;
```

Field	Type	Null	Key	Default	Extra
bikename	varchar(30)	NO	PRI	NULL	
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>
```

Annotations:

- A blue arrow points from the text "More complicated alter table command." to the `alter table bikes` command.
- A green arrow points from the text "Bikes table after the alteration" to the `describe bikes;` output table.



# 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.



Battaglin Carrera	60	red/white	4000	2001-03-10	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-02	300
Bianchi Evolution 3	58	celeste	4800	2003-11-12	2000
Bianchi Infinito	58	celeste	8900	2011-07-14	0
BMC SLC01 - Swiss	58	red/black/white	8000	2010-06-23	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-07	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2004-02-02	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
Ridley Damocles	58	blue/black	7500	2008-06-27	0
Ridley X-Fire	58	red/white	7500	2011-09-01	0
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

14 rows in set (0.00 sec)

```
mysql> insert into bikes
-> set bikename = "Eddy Merckx EM7",
-> cost=9500,
-> mileage=100,
-> purchased="2011-01-01",
-> color="red/white/blue",
-> 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	2001-03-10	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-02	300
Bianchi Evolution 3	58	celeste	4800	2003-11-12	2000
Bianchi Infinito	58	celeste	8900	2011-07-14	0
BMC SLC01 - Swiss	58	red/black/white	8000	2010-06-23	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-07	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2004-02-02	0
Eddy Merckx EM7	58	red/white/blue	9500	2011-01-01	100
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
Ridley Damocles	58	blue/black	7500	2008-06-27	0
Ridley X-Fire	58	red/white	7500	2011-09-01	0
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

15 rows in set (0.00 sec)

```
mysql> _
```

## Examples: Inserting Data Into A Table

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



mysql> select \* from bikes;

bikename	size	color	cost	purchased	mileage
Battaglin Carrera	60	red/white	4000	2001-03-10	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-02	300
Bianchi Evolution 3	58	celeste	4800	2003-11-12	2000
Bianchi Infinito	58	celeste	8900	2011-07-14	0
BMC SLC01 - Swiss	58	red/black/white	8000	2010-06-23	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-07	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2004-02-02	0
Eddy Merckx EM7	58	red/white/blue	9500	2011-01-01	100
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
Ridley Damocles	58	blue/black	7500	2008-06-27	0
Ridley X-Fire	58	red/white	7500	2011-09-01	0
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

15 rows in set (0.00 sec)

mysql> insert into bikes  
-> values("Ridley Crosswind",58,"black",6500,"2010-04-05",2000);

Query OK, 1 row affected (0.05 sec)

mysql> select \* from bikes;

bikename	size	color	cost	purchased	mileage
Battaglin Carrera	60	red/white	4000	2001-03-10	11200
Bianchi Corse Evo 4	58	celeste	5700	2004-12-02	300
Bianchi Evolution 3	58	celeste	4800	2003-11-12	2000
Bianchi Infinito	58	celeste	8900	2011-07-14	0
BMC SLC01 - Swiss	58	red/black/white	8000	2010-06-23	0
Colnago Dream Rabobank	60	blue/orange	5500	2002-07-07	4300
Colnago Superissimo	59	red	3800	1996-03-01	13000
Eddy Merckx Domo	58	blue/black	5300	2004-02-02	0
Eddy Merckx EM7	58	red/white/blue	9500	2011-01-01	100
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
Ridley Crosswind	58	black	6500	2010-04-05	2000
Ridley Damocles	58	blue/black	7500	2008-06-27	0
Ridley X-Fire	58	red/white	7500	2011-09-01	0
Schwinn Paramount P14	60	blue	1800	1992-03-01	200

16 rows in set (0.00 sec)

mysql> \_

Using Form 2  
for insertion –  
attribute order  
is important.



# Examples: Inserting Data Into A Table

```
C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe

mysql>
mysql>
mysql> create table celestebikes like bikes;
Query OK, 0 rows affected (0.11 sec)

mysql> select * from celestebikes;
Empty set (0.00 sec)

mysql> insert into celestebikes
-> select *
-> from bikes
-> where color = "celeste";
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from celestebikes;
+-----+-----+-----+-----+-----+-----+
| bikename          | size | color  | cost  | purchased | mileage |
+-----+-----+-----+-----+-----+-----+
| Bianchi Corse Evo 4 | 58   | celeste | 5700  | 2004-12-02 | 300    |
| Bianchi Evolution 3 | 58   | celeste | 4800  | 2003-11-12 | 2000   |
| Bianchi Infinito    | 58   | celeste | 8900  | 2011-07-14 | 0      |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

Creates an initially empty table just like the bikes table

Table creation did not place any data into the 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

```
C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe

3 rows in set (0.00 sec)

mysql> drop table celestebikes;
Query OK, 0 rows affected (0.05 sec)

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 (0.10 sec)

mysql> insert into celestebikes
  -> select bikename, color, cost, mileage
  -> from bikes
  -> where color = "celeste";
Query OK, 3 rows affected (0.05 sec)
Records: 3  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         |
| Bianchi Infinito    | celeste | 8900  | 0            |
+-----+-----+-----+-----+

3 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. The third and more preferable option is to use the MySQL Workbench tool (see page 98 and on.)
- 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.)

\*C:\state script.sql - Notepad++

File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?

template.html commentform.html fourthCSS.css state script.sql

```
1 #SQL commands in a script file
2 drop database if exists testdb;
3
4 create database testdb;
5
6 use testdb;
7
8 create table states (
9     name varchar(15) not null,
10    abbrev char(2),
11    capital varchar(25),
12    population integer,
13    square_miles integer,
14    primary key (name)
15 );
16
17 insert into states values ('Florida', 'FL', 'Tallahassee', 18328240, 54153);
18 insert into states values ('New York', 'NY', 'Albany', 194909297, 54556);
19 insert into states values ('Indiana', 'IN', 'Indianapolis', 6376792, 35789);
20 insert into states values ('Maryland', 'MD', 'Annapolis', 5633597, 9975);
21
22 select * from states;
```

Drop the database if it already exists.

Create a new database.

Switch to the new database.

Define schema for the new table.

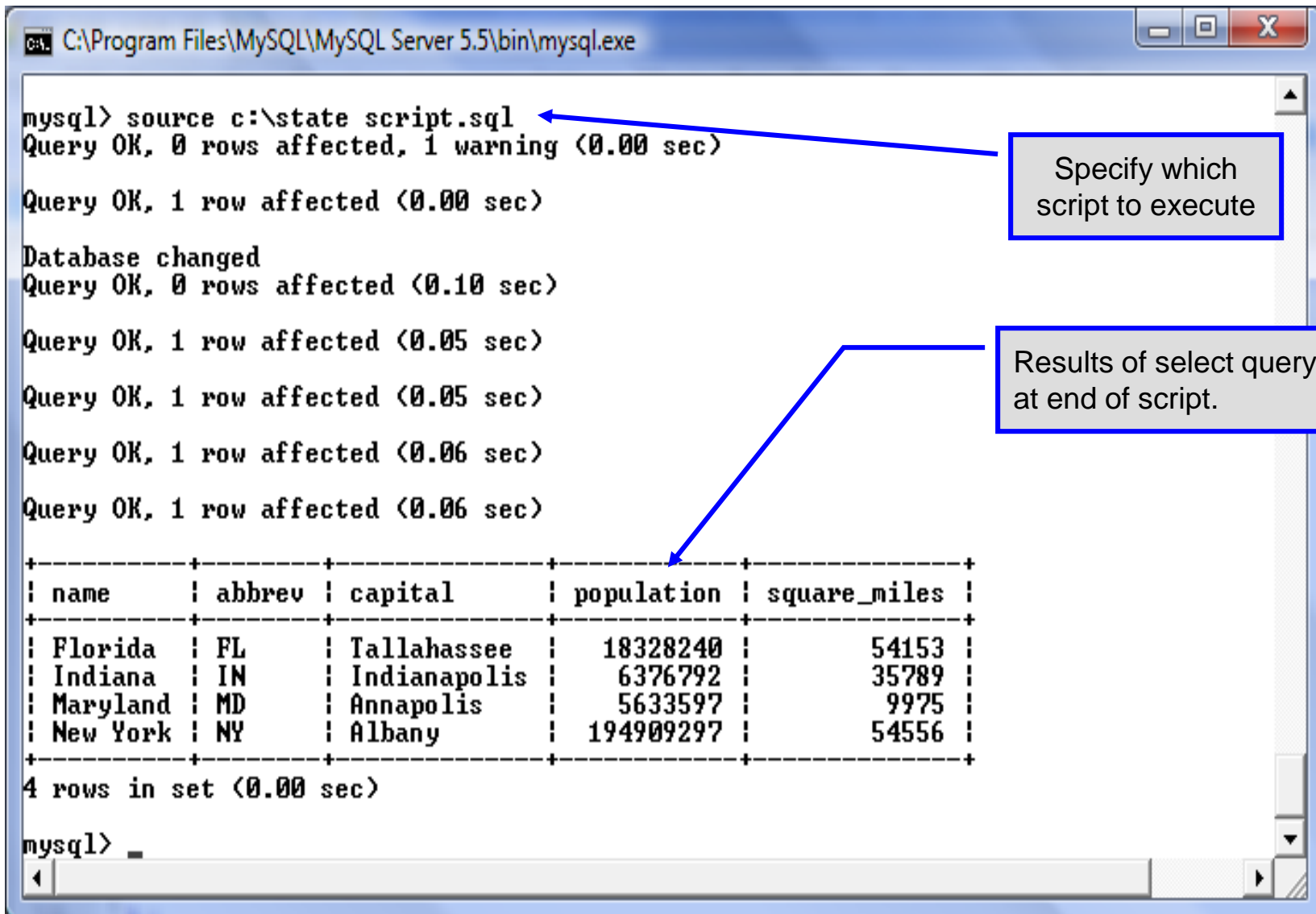
Insert some tuples

Run a simple selection query on the new table.

Structured Query Language file nb char : 616 nb line : 22



# Using Scripts with MySQL (cont.)



The screenshot shows a MySQL command prompt window titled "C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe". The prompt is "mysql>". The user has entered the command "source c:\state script.sql". The output shows several "Query OK" messages, indicating successful execution of queries. A table of state data is displayed, with columns: name, abbrev, capital, population, and square\_miles. The table contains 4 rows of data for Florida, Indiana, Maryland, and New York. The prompt is now "mysql> \_".

```
mysql> source c:\state 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 (0.10 sec)

Query OK, 1 row affected (0.05 sec)

Query OK, 1 row affected (0.05 sec)

Query OK, 1 row affected (0.06 sec)

Query OK, 1 row affected (0.06 sec)

+-----+-----+-----+-----+-----+
| name   | abbrev | capital   | population | square_miles |
+-----+-----+-----+-----+-----+
| Florida | FL     | Tallahassee | 18328240   | 54153         |
| Indiana | IN     | Indianapolis | 6376792    | 35789         |
| Maryland | MD    | Annapolis   | 5633597    | 9975          |
| New York | NY    | Albany      | 194909297  | 54556         |
+-----+-----+-----+-----+-----+

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:

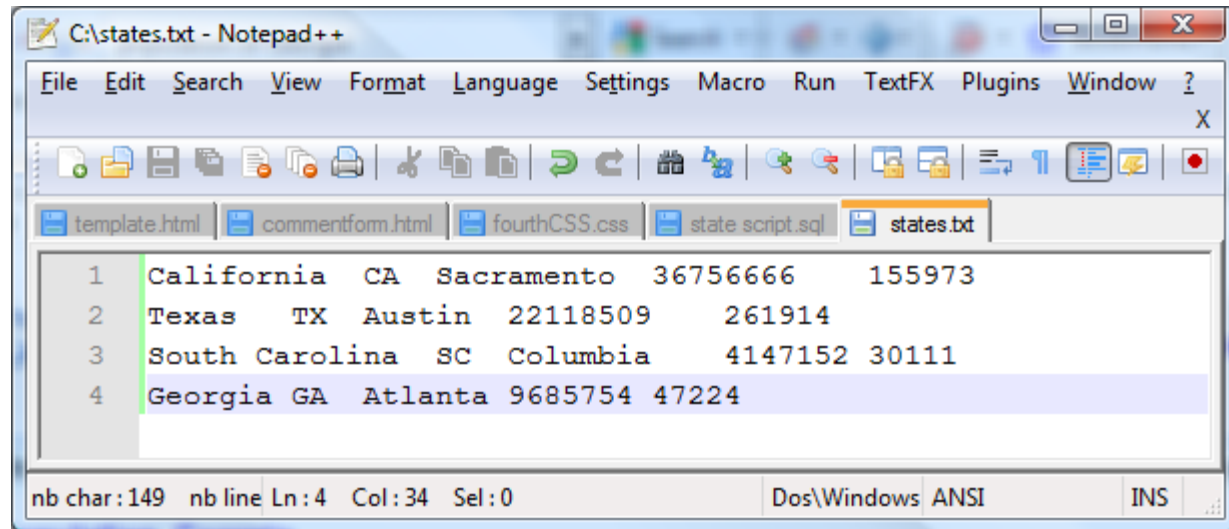
```
mysqlimport [options] database_name file1 file2 ...
```

- 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 `mysqlimport` Utility (cont.)

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



```
1 California CA Sacramento 36756666 155973
2 Texas TX Austin 22118509 261914
3 South Carolina SC Columbia 4147152 30111
4 Georgia GA Atlanta 9685754 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 65-66.



# Importing Data Using the `mysqlimport` Utility

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

```
Prompt

C:\Program Files\MySQL\MySQL Server 5.1\bin>mysqlimport -u root -vr testdb c:\st
Connecting to localhost
Selecting database testdb
Loading data from SERVER file: c:/states.txt into states
testdb.states: Records: 4 Deleted: 0 Skipped: 0 Warnings: 3
Disconnecting from localhost

C:\Program Files\MySQL\MySQL Server 5.1\bin>
```

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

Table updated



# Importing Data Using the mysqlimportUtility

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

+-----+-----+-----+-----+-----+
| name   | abbrev | capital | population | square_miles |
+-----+-----+-----+-----+-----+
| Florida | FL     | Tallahassee | 18328240 | 54153 |
| New York | NY    | Albany      | 194909297 | 54556 |
| Indiana  | IN    | Indianapolis | 6376792 | 35789 |
| Maryland | MD    | Annapolis   | 5633597 | 9975 |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> select * from states;

+-----+-----+-----+-----+-----+
| name           | abbrev | capital      | population | square_miles |
+-----+-----+-----+-----+-----+
| Florida        | FL     | Tallahassee  | 18328240 | 54153 |
| New York       | NY    | Albany       | 194909297 | 54556 |
| Indiana        | IN    | Indianapolis  | 6376792 | 35789 |
| Maryland       | MD    | Annapolis    | 5633597 | 9975 |
| California     | CA     | Sacramento   | 36756666 | 155973 |
| Texas          | TX     | Austin       | 22118509 | 261914 |
| South Carolina | SC     | Columbia     | 4147152 | 30111 |
| Georgia        | GA     | Atlanta      | 9685754 | 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 70 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 75.

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

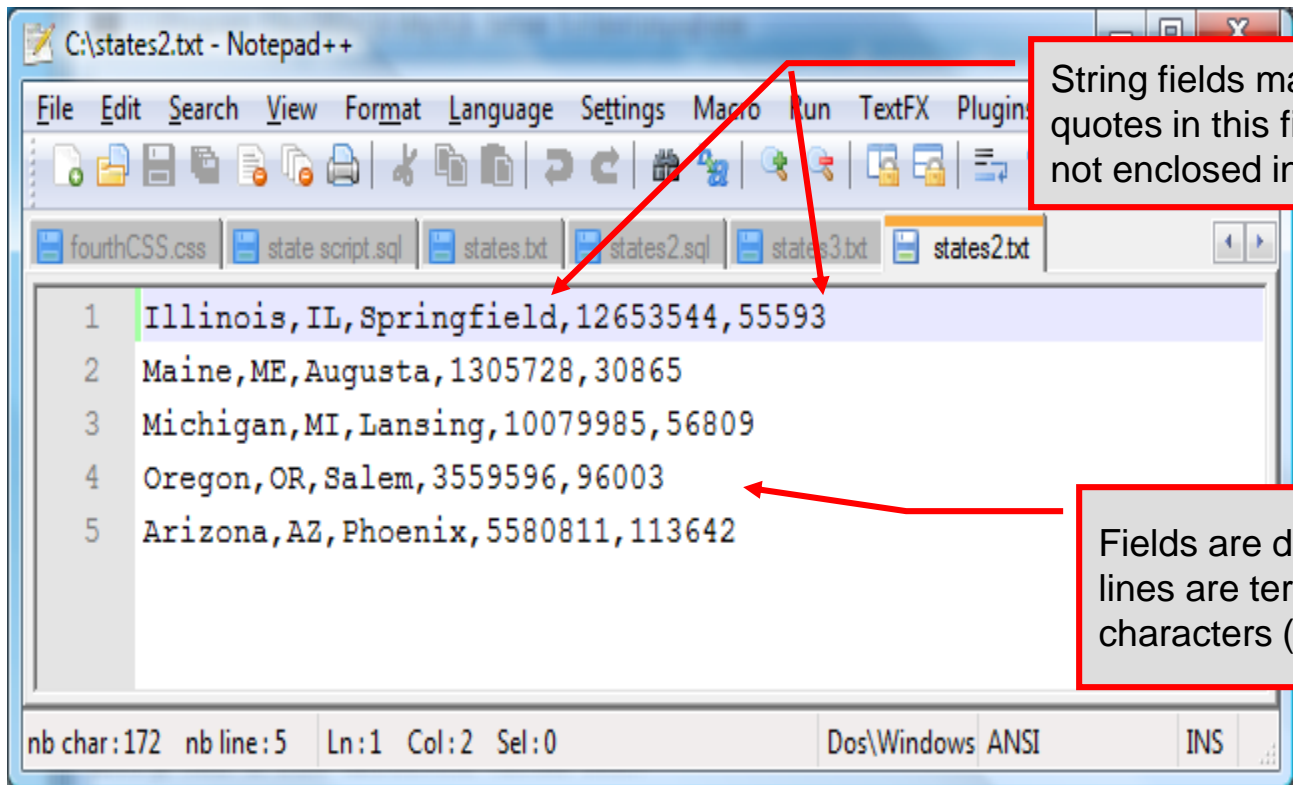
Sets the characters that delimit and enclose the fields and lines in the data file. Similar to `mysqlimport` syntax.

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

Used to load only certain columns (not entire rows)



# Load Data Infile Example



The screenshot shows a Notepad++ window titled 'C:\states2.txt - Notepad++'. The menu bar includes File, Edit, Search, View, Format, Language, Settings, Macro, Run, TextFX, and Plugin. The toolbar contains various icons for file operations and editing. The tab bar shows several files, with 'states2.txt' selected. The text area contains the following data:

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

Two callout boxes provide additional information:

- A box at the top right states: "String fields may be enclosed by double quotes in this file. Numeric values are not enclosed in quotes." Red arrows point from this box to the state names and the numeric values in the first line of the data.
- A box at the bottom right states: "Fields are delimited by commas and lines are terminated by newline characters (an invisible \n)". A red arrow points from this box to the comma between 'Salem' and '3559596' in the fourth line.

The status bar at the bottom indicates 'nb char:172 nb line:5 Ln:1 Col:2 Sel:0' and 'Dos\Windows ANSI' encoding.

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



```
mysql>
mysql> select * from states;
```

name	abbrev	capital	population	square_miles
Florida	FL	Tallahassee	18328240	54153
New York	NY	Albany	194909297	54556
Indiana	IN	Indianapolis	6376792	35789
Maryland	MD	Annapolis	5633597	9975
California	CA	Sacramento	36756666	155973
Texas	TX	Austin	22118509	261914
South Carolina	SC	Columbia	4147152	30111
Georgia	GA	Atlanta	9685754	47224

8 rows in set (0.00 sec)

```
mysql> load data infile 'c:/states2.txt'
-> into table states
-> fields
-> terminated by ','
-> optionally enclosed by '"';
Query OK, 5 rows affected (0.00 sec)
Records: 5 Deleted: 0 Skipped: 0 Warnings: 0
```

```
mysql> select * from states;
```

name	abbrev	capital	population	square_miles
Florida	FL	Tallahassee	18328240	54153
New York	NY	Albany	194909297	54556
Indiana	IN	Indianapolis	6376792	35789
Maryland	MD	Annapolis	5633597	9975
California	CA	Sacramento	36756666	155973
Texas	TX	Austin	22118509	261914
South Carolina	SC	Columbia	4147152	30111
Georgia	GA	Atlanta	9685754	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> _
```

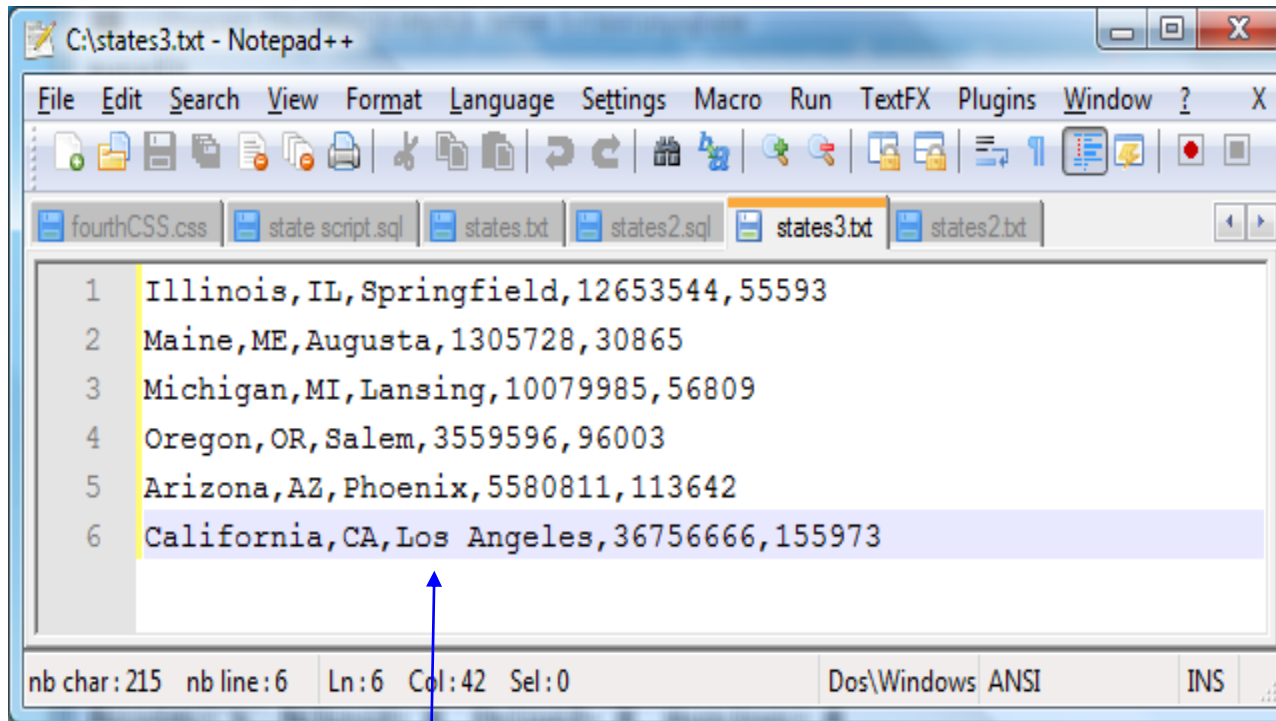
States table **before** addition of data

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



The screenshot shows a Notepad++ window titled 'C:\states3.txt - Notepad++'. The menu bar includes File, Edit, Search, View, Format, Language, Settings, Macro, Run, TextFX, Plugins, Window, and Help. The toolbar contains various icons for file operations and editing. The tab bar shows several files, with 'states3.txt' selected. The text area contains six lines of data, with the sixth line highlighted in blue. A blue arrow points from the text below to the sixth line.

```
1 Illinois,IL,Springfield,12653544,55593
2 Maine,ME,Augusta,1305728,30865
3 Michigan,MI,Lansing,10079985,56809
4 Oregon,OR,Salem,3559596,96003
5 Arizona,AZ,Phoenix,5580811,113642
6 California,CA,Los Angeles,36756666,155973
```

nb char: 215 nb line: 6 Ln: 6 Col: 42 Sel: 0 Dos\Windows ANSI INS

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

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



mysql> select \* from states;

name	abbrev	capital	population	square_miles
Florida	FL	Tallahassee	18328240	54153
New York	NY	Albany	194909297	54556
Indiana	IN	Indianapolis	6376792	35789
Maryland	MD	Annapolis	5633597	9975
California	CA	Sacramento	36756666	155973
Texas	TX	Austin	22118509	261914
South Carolina	SC	Columbia	4147152	30111
Georgia	GA	Atlanta	9685754	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 'c:/states3.txt'  
-> 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> select \* from states;

name	abbrev	capital	population	square_miles
Florida	FL	Tallahassee	18328240	54153
New York	NY	Albany	194909297	54556
Indiana	IN	Indianapolis	6376792	35789
Maryland	MD	Annapolis	5633597	9975
California	CA	Los Angeles	36756666	155973
Texas	TX	Austin	22118509	261914
South Carolina	SC	Columbia	4147152	30111
Georgia	GA	Atlanta	9685754	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!



# 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 an 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,...), (...)`

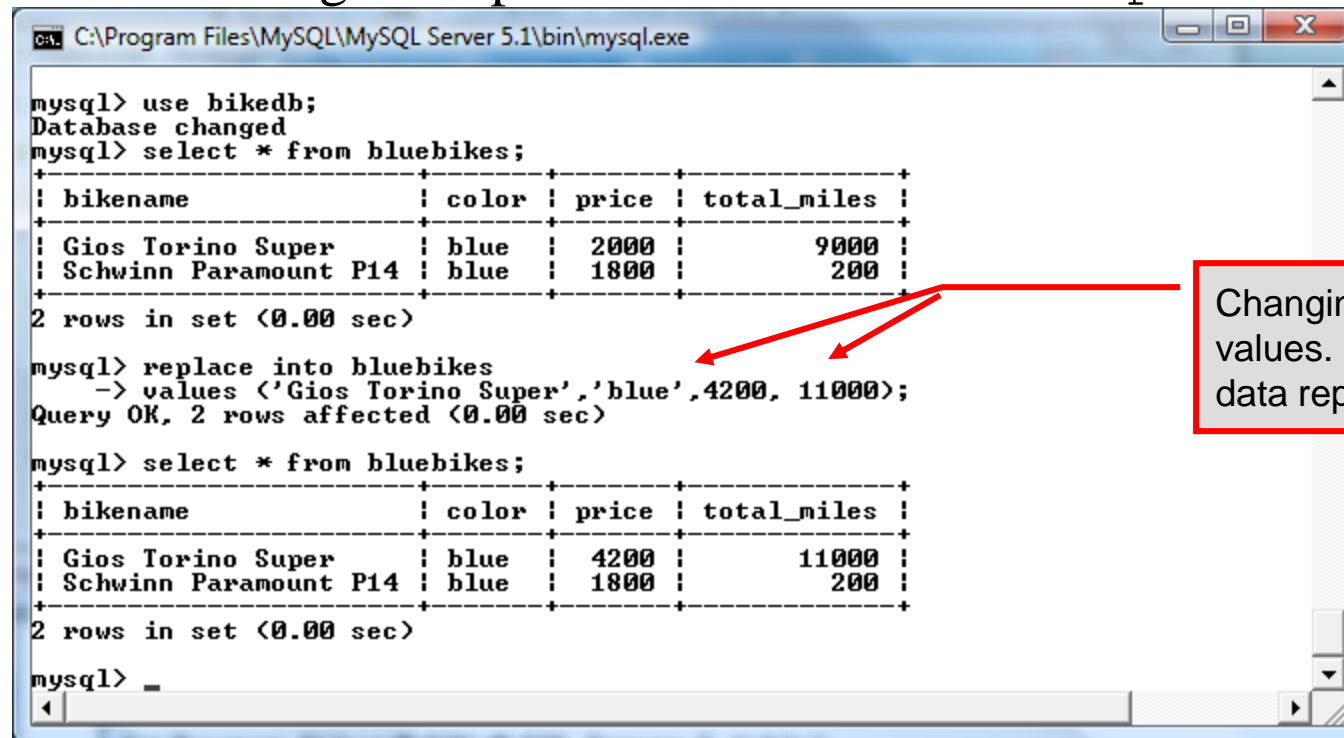
---

**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.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

mysql> use bikedb;
Database changed
mysql> select * from bluebikes;
+-----+-----+-----+-----+
| bikename          | color | price | total_miles |
+-----+-----+-----+-----+
| Gios Torino Super | blue  | 2000  | 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> _
```

The screenshot shows a MySQL command prompt window. The user first uses the 'use bikedb;' command to select the database. Then, they run 'select \* from bluebikes;' which displays a table with two rows: 'Gios Torino Super' (blue, 2000, 9000 miles) and 'Schwinn Paramount P14' (blue, 1800, 200 miles). Next, the user runs 'replace into bluebikes values ('Gios Torino Super','blue',4200,11000);'. The output shows 'Query OK, 2 rows affected (0.00 sec)'. Finally, the user runs 'select \* from bluebikes;' again, showing the updated table where the 'Gios Torino Super' row now has a price of 4200 and 11000 miles. Red arrows point from the 'price' and 'total\_miles' columns in the second query result to a text box on the right.

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



# Using Replace (cont.)

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

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 ('Ridley Damocles','blue',8500,1000);
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         |
| Ridley Damocles    | blue  | 8500  | 1000        |
+-----+-----+-----+-----+
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.)

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> select * from bluebikes;
+-----+-----+-----+-----+
| bikename          | color | price | total_miles |
+-----+-----+-----+-----+
| Gios Torino Super | blue  | 4200  | 11000       |
| Schwinn Paramount P14 | blue  | 1800  | 200         |
| Ridley Damocles    | blue  | 8500  | 1000        |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> update bluebikes
-> set price=price*1.05;
Query OK, 3 rows affected (0.00 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         |
| Ridley Damocles    | blue  | 8925  | 1000        |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

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



# Using update (cont.)

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql>
mysql>
mysql>
mysql> select * from bluebikes;
+-----+-----+-----+-----+
| bikename          | color | price | total_miles |
+-----+-----+-----+-----+
| Gios Torino Super | blue  | 4410  | 11000       |
| Schwinn Paramount P14 | blue  | 1890  | 200         |
| Ridley Damocles    | blue  | 8925  | 1000        |
+-----+-----+-----+-----+
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  | 4410  | 11000       |
| Schwinn Paramount P14 | blue  | 1890  | 200         |
| Ridley Damocles    | blue  | 9371  | 1000        |
+-----+-----+-----+-----+
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] [SQL_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

The mysql database contains user information

Details on user privileges at the database level. See page 94.

Specific details on privileges at the table level. See page 93

Details on user privileges. See page 91.

Details about the various users. See page 92.

C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

```
mysql> show databases;
```

Database
information_schema
bikedb
mysql
sample
test
testdb

6 rows in set (0.00 sec)

```
mysql> show tables;
```

Tables_in_mysql
columns_priv
db
event
func
general_log
help_category
help_keyword
help_relation
help_topic
host
ndb_binlog_index
plugin
proc
procs_priv
servers
slow_log
tables_priv
time_zone
time_zone_leap_second
time_zone_name
time_zone_transition
time_zone_transition_type
user
user_info

24 rows in set (0.00 sec)

```
mysql> _
```



# Contents of the user Table

outt; - Notepad

File Edit Format View Help

```
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

```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

mysql> describe user_info;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| User           | varchar(16)   | NO   | PRI | NULL     |       |
| 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 (0.02 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','Alter') |
| Column_priv | set('Select','Insert','Update','References') |
+-----+-----+
8 rows in set (0.00 sec)
```

```
mysql>
mysql> show index from tables_priv;
+-----+-----+-----+-----+-----+
| Table | Non_unique | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| tables_priv | YES | YES | PRI | CURRENT_TIMESTAMP | INDEX,Alter') |
+-----+-----+-----+-----+-----+
```



# Contents of the db Table

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> describe db;
+-----+-----+-----+-----+-----+-----+
| Field                | Type                | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Host                 | char(60)            | NO   | PRI |          |       |
| Db                   | char(64)            | NO   | PRI |          |       |
| User                 | char(16)            | NO   | PRI |          |       |
| Select_priv          | enum('N','Y')       | NO   |     | N        |       |
| Insert_priv          | enum('N','Y')       | NO   |     | N        |       |
| Update_priv          | enum('N','Y')       | NO   |     | N        |       |
| Delete_priv          | enum('N','Y')       | NO   |     | N        |       |
| Create_priv          | enum('N','Y')       | NO   |     | N        |       |
| Drop_priv            | enum('N','Y')       | NO   |     | N        |       |
| Grant_priv           | enum('N','Y')       | NO   |     | N        |       |
| References_priv      | enum('N','Y')       | NO   |     | N        |       |
| Index_priv           | enum('N','Y')       | NO   |     | N        |       |
| Alter_priv           | enum('N','Y')       | NO   |     | N        |       |
| Create_tmp_table_priv | enum('N','Y')       | NO   |     | N        |       |
| Lock_tables_priv     | enum('N','Y')       | NO   |     | N        |       |
| Create_view_priv     | enum('N','Y')       | NO   |     | N        |       |
| Show_view_priv       | enum('N','Y')       | NO   |     | N        |       |
| Create_routine_priv  | enum('N','Y')       | NO   |     | N        |       |
| Alter_routine_priv   | enum('N','Y')       | NO   |     | N        |       |
| Execute_priv         | enum('N','Y')       | NO   |     | N        |       |
| Event_priv           | enum('N','Y')       | NO   |     | N        |       |
| Trigger_priv         | enum('N','Y')       | NO   |     | N        |       |
+-----+-----+-----+-----+-----+-----+
22 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:

```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

2 rows in set (0.00 sec)

mysql> show grants for mark;

+-----+
| Grants for mark@% |
+-----+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON *.* TO 'mark'@'%' IDENTIFIED BY PASSWORD 'E6ACCEDB2495496B191ED488F598F04239C85E73' |
| GRANT ALL PRIVILEGES ON `bikedb`.* TO 'mark'@'%' WITH GRANT OPTION |
+-----+

2 rows in set (0.00 sec)

mysql>
```

This user has only SELECT, INSERT, UPDATE, DELETE, and CREATE global privileges.

The user has all privileges on the bikedb database.



# 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 REVOKE

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

+-----+
| Grants for mark@% |
+-----+
+-----+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON *.* TO 'mark'@'%' IDENTIFIED BY PASSWORD '*E6ACCEDB2495496B191ED488F598F04239C85E73' |
| GRANT SELECT ON `testdb`.* TO 'mark'@'%' |
+-----+
| GRANT ALL PRIVILEGES ON `hikedb`.* TO 'mark'@'%' WITH GRANT OPTION |
| GRANT SELECT ON `testdb`.`states` TO 'mark'@'%' |
+-----+

4 rows in set (0.00 sec)

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

mysql> show grants for mark;
+-----+
| Grants for mark@% |
+-----+
+-----+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON *.* TO 'mark'@'%' IDENTIFIED BY PASSWORD '*E6ACCEDB2495496B191ED488F598F04239C85E73' |
| GRANT SELECT ON `testdb`.* TO 'mark'@'%' |
+-----+
| GRANT ALL PRIVILEGES ON `hikedb`.* TO 'mark'@'%' WITH GRANT OPTION |
+-----+

3 rows in set (0.00 sec)

mysql>
```

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.



# More Details On The MySQL Workbench

- The Workbench contains a fairly extensive set of administrator tools for maintaining your MySQL Server instances.
- The following slides illustrate some of these features. I'd encourage you to play around with the Workbench and get familiar with using it.



# The MySQL Workbench

Local instance MySQL56 (bikedb) x Local instance MySQL56 (bik... x

File Edit View Query Database Server Tools Window Help

Navigator: Query 1 Administration - Server Status x

**SCHEMAS**

Filter objects

- bikedb
- coloursurvey
- guestbook
- largeco
- mailinglist
- project2
- project3
- project4
- project5
- project6
- test
- vendor2
- vendors

Management Schemas

Information

No object selected

Object Info Session

**MySQL Server 5.6**

Connection Name: Local instance MySQL56

Host: HEC-236-G4TB8Y1

Socket: MySQL

Port: 3310

Version: 5.6.16  
MySQL Community Server (GPL)

Compiled For: Win64 (x86\_64)

**Available Server Features**

Performance Schema:	<input checked="" type="radio"/> On	SSL Availability:	<input type="radio"/> Off
Thread Pool:	<input type="radio"/> n/a	Windows Authentication:	<input type="radio"/> Off
Memcached Plugin:	<input type="radio"/> n/a	Password Validation:	<input type="radio"/> n/a
Semisync Replication Plugin:	<input type="radio"/> n/a	Audit Log:	<input type="radio"/> n/a

**Server Directories**

Base Directory: C:\Program Files\MySQL\MySQL Server 5.6\  
 Data Directory: C:\ProgramData\MySQL\MySQL Server 5.6\data\  
 Disk Space in Data Dir: 715.00 GB of 922.00 GB available  
 Plugins Directory: C:\Program Files\MySQL\MySQL Server 5.6\lib\plugin\  
 Tmp Directory: C:\Windows\SERVIC~2\NETWOR~1\AppData\Local\Temp

Error Log: ☒ On .\HEC-236-G4TB8Y1.err  
 General Log: ☐ Off  
 Slow Query Log: ☐ Off

**Server Status**

Running

CPU: 2%

Connections: 7

Traffic: 0.0 B/s

Key Efficiency: 0.0%

Queries per Second: 0

InnoDB Buffer Usage: 0.9%

InnoDB Reads per Second: 0

InnoDB Writes per Second: 0





MySQL Workbench

Local instance MySQL56 (bikedb) x Local instance MySQL56 (bik... x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

bikedb

coloursurvey

guestbook

largeco

mailinglist

project2

project3

project4

project5

project6

test

Management Schemas

Information

No object selected

Object Info Session

Query 1 Administration - Startup / Shutdo... x

Local instance MySQL56

### Startup / Shutdown MySQL Server

The database server is started and ready for client connections. To shut the Server down, use the "Stop Server" button

The database server instance is **running**

If you stop the server, you and your applications will not be able to use the Database and all current connections will be closed

#### Startup Message Log

2014-02-11 08:39:48 - Workbench will use cmd shell commands to start/stop this instance  
2014-02-11 08:39:48 - Status check of service 'MySQL56' returned running

Refresh Status

Starting and Stopping the Server





Navigator

Query 1

Administration - Server Logs x

## SCHEMAS



Filter objects

- bikedb
- coloursurvey
- guestbook
- largeco
- mailinglist
- project2
- project3
- project4
- project5
- project6
- test

Management Schemas

Information

No object selected



Local instance MySQL56

## Server Logs

Server log files for details on server status.

Error Log File

Timestamp	Thread	Type	Details
2014-02-11 08:33:05	9236	Note	InnoDB: 5.6.16 started; log sequence number 21621696
2014-02-11 08:33:05	9236	Note	Server hostname (bind-address): '*'; port: 3310
2014-02-11 08:33:05	9236	Note	IPv6 is available.
2014-02-11 08:33:05	9236	Note	-'::' resolves to '::';
2014-02-11 08:33:05	9236	Note	Server socket created on IP: '::'.
2014-02-11 08:33:05	9236	Note	C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqld.exe: ready for connection
			Version: '5.6.16' socket: " port: 3310 MySQL Community Server (GPL)
2014-02-11 08:33:12	4904	Note	Plugin 'FEDERATED' is disabled.
2014-02-11 08:33:12	4904	Warning	option 'innodb-autoextend-increment': unsigned value 67108864 adjusted to 100
			2014-02-11 08:33:12 2828 InnoDB: Warning: Using innodb_additional_mem_poc
2014-02-11 08:33:12	4904	Note	InnoDB: Using atomics to ref count buffer pool pages
2014-02-11 08:33:12	4904	Note	InnoDB: The InnoDB memory heap is disabled
2014-02-11 08:33:12	4904	Note	InnoDB: Mutexes and rw_locks use Windows interlocked functions
2014-02-11 08:33:12	4904	Note	InnoDB: Compressed tables use zlib 1.2.3
2014-02-11 08:33:12	4904	Note	InnoDB: Not using CPU crc32 instructions
2014-02-11 08:33:12	4904	Note	InnoDB: Initializing buffer pool, size = 740.0M
2014-02-11 08:33:12	4904	Note	InnoDB: Completed initialization of buffer pool
2014-02-11 08:33:12	4904	Note	InnoDB: Highest supported file format is Barracuda

Log File Location: C:\ProgramData\MySQL\MySQL Server 5.6\data\HEC-236-G4TB8Y1.err

Log File Size: 54.4 kB

Showing: 703 records starting at byte offset 0

Oldest

&lt; Previous Page

Next Page &gt;

Most Recent

Refresh

Object Info Session



Local instance MySQL56 (bikedb) x Local instance MySQL56 (bik... x

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS Filter objects

- bikedb
- coloursurvey
- guestbook
- largeco
- mailinglist
- project2
- project3
- project4
- project5
- project6
- test

Management Schemas

Information No object selected

Query 1 Administration - Client Connections x

Local instance MySQL56

### Client Connections

Id	User	Host	DB	Command	Time	State	Info
2	root	localhost:64695	bikedb	Sleep	294		NULL
3	root	localhost:64696	bikedb	Sleep	294		NULL
4	root	localhost:64704	None	Sleep	246		NULL
6	root	localhost:64716	bikedb	Sleep	115		NULL
7	root	localhost:64717	bikedb	Sleep	116		NULL
9	root	localhost:64719	None	Query	0	init	SHOW FULL PROCESSLIST
10	root	localhost:64720	None	Sleep	3		NULL

Client connections to the server.

Refresh Rate: Don't Refresh Hide sleeping connections Kill Query Kill Connection Refresh



Local instance MySQL56 (bikedb) x Local instance MySQL56 (bik... x

File Edit View Query Database Server Tools Scripting Help

Navigator: Query 1 Administration - Users and Privil... x

**SCHEMAS**

Filter objects

- bikedb
- coloursurvey
- guestbook
- largeco
- mailinglist
- project2
- project3
- project4
- project5
- project6
- test

Management Schemas

Information: No object selected

Local instance MySQL56

## Users and Privileges

User Accounts

User	From Host
(!) <anonymous>	%
client1	%
client2	%
root	localhost
root	127.0.0.1
root	:::1

Details for account root@localhost

Login Account Limits Administrative Roles Schema Privileges

Role	Description
<input checked="" type="checkbox"/> DBA	grants the rights to
<input checked="" type="checkbox"/> MaintenanceAdmin	grants rights need
<input checked="" type="checkbox"/> ProcessAdmin	rights needed to a
<input checked="" type="checkbox"/> UserAdmin	grants rights to cr
<input checked="" type="checkbox"/> SecurityAdmin	rights to manage
<input checked="" type="checkbox"/> MonitorAdmin	minimum set of ri
<input checked="" type="checkbox"/> DBManager	grants full rights c
<input checked="" type="checkbox"/> DBDesigner	rights to create an
<input checked="" type="checkbox"/> ReplicationAdmin	rights needed to s
<input checked="" type="checkbox"/> BackupAdmin	minimal rights nee

Global Privileges

- ☒ ALTER
- ☒ ALTER ROUTINE
- ☒ CREATE
- ☒ CREATE ROUTINE
- ☒ CREATE TABLESPACE
- ☒ CREATE TEMPORARY TABLES
- ☒ CREATE USER
- ☒ CREATE VIEW
- ☒ DELETE
- ☒ DROP
- ☒ EVENT
- ☒ EXECUTE
- ☒ FILE
- ☒ GRANT OPTION
- ☒ INDEX
- ☒ INSERT
- ☒ LOCK TABLES

Add Account Drop

Revoke All Privileges Expire Password Revert Apply Refresh

**User privileges**





Options file for fine tuning  
server behavior.



Navigator

Query 1

Administration - Options File x

## SCHEMAS

Filter objects

bikedb  
coloursurvey  
guestbook  
largeco  
mailinglist  
project2  
project3  
project4  
project5  
project6  
test

Management Schemas

Information

No object selected



Local instance MySQL56

## Options File

Locate option:

Find

General

Logging

InnoDB

Networking

Advanced

Other

Security

Replication

MyISAM

Performance

## Features

☐ event-scheduler

OFF

Enable/disable and start/stop the event scheduler. Note that this variable underwent significant changes in behavior and permitted values in MySQL 5.1.11 and 5.1.12

☐ federated

Enables the FEDERATED storage engine

☒ partition

Enable (or disable) partitioning support

☐ plugin

Prefix for specifying plugin-specific options.

☐ plugin-load

Set the list of plugins to load at startup

☐ plugin-load-add

Add to list of plugins to load at startup

☐ profiling\_history\_size

15

How many statements to maintain profiling information for

☐ skip-event-scheduler

Sets the Event Scheduler to OFF.

☐ skip-partition

Do not enable user-defined partitioning

## Transactions

☐ autocommit

Sets the autocommit mode

Configuration File: C:\ProgramData\MySQL\MySQL Server 5.6\my.ini

mysqld

Apply...

Discard

Object Info Session



An EER diagram reverse engineered from a MySQL database. To start this process, select Database, then Reverse Engineer.

