

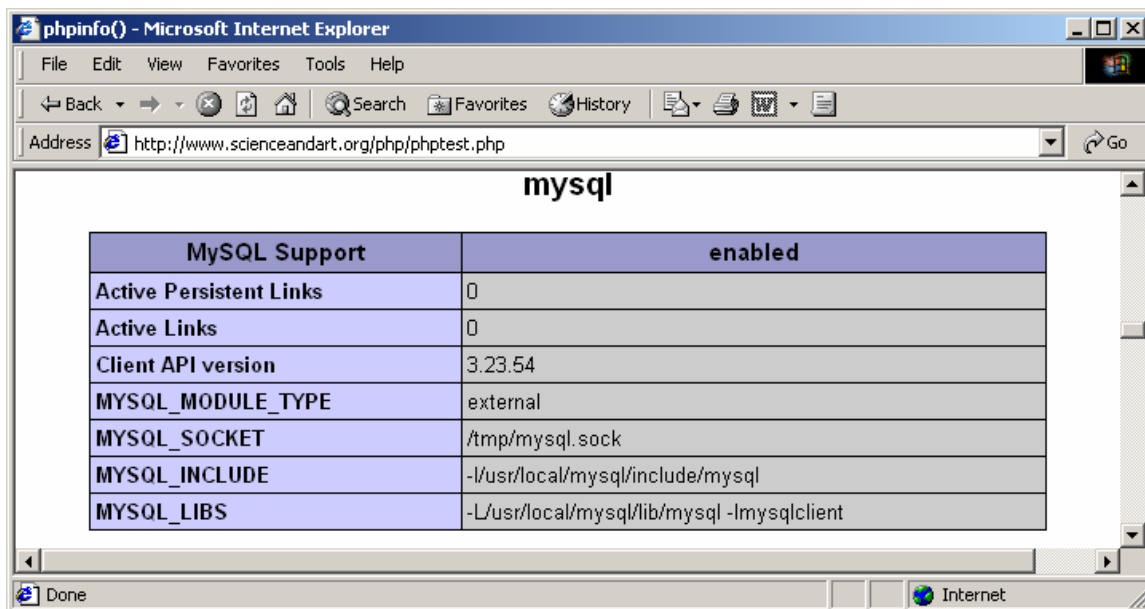
MySQL and PHP – Getting Started 21/1/2003

MySQL is a non-proprietary, cross platform database management system that is suitable for small and medium sized businesses. It is available for Win, Unix and Mac by downloading from www.mysql.com. A database management system is the software that interfaces with the database.

MySQL consists of several pieces of software including: MySQL server (mysqld) which manages the database, the mysql client (mysql) which provides the interface to the server and various utilities.

You can interface with MySQL using a variety of programming languages, including Perl, Java, C, C++ and PHP (called programming APIs). PHP is the most common API. MySQL can handle databases as large as 60,000 tables with more than 5 billion rows. The current version is 3.23.49a – version 4.0 is in the works.

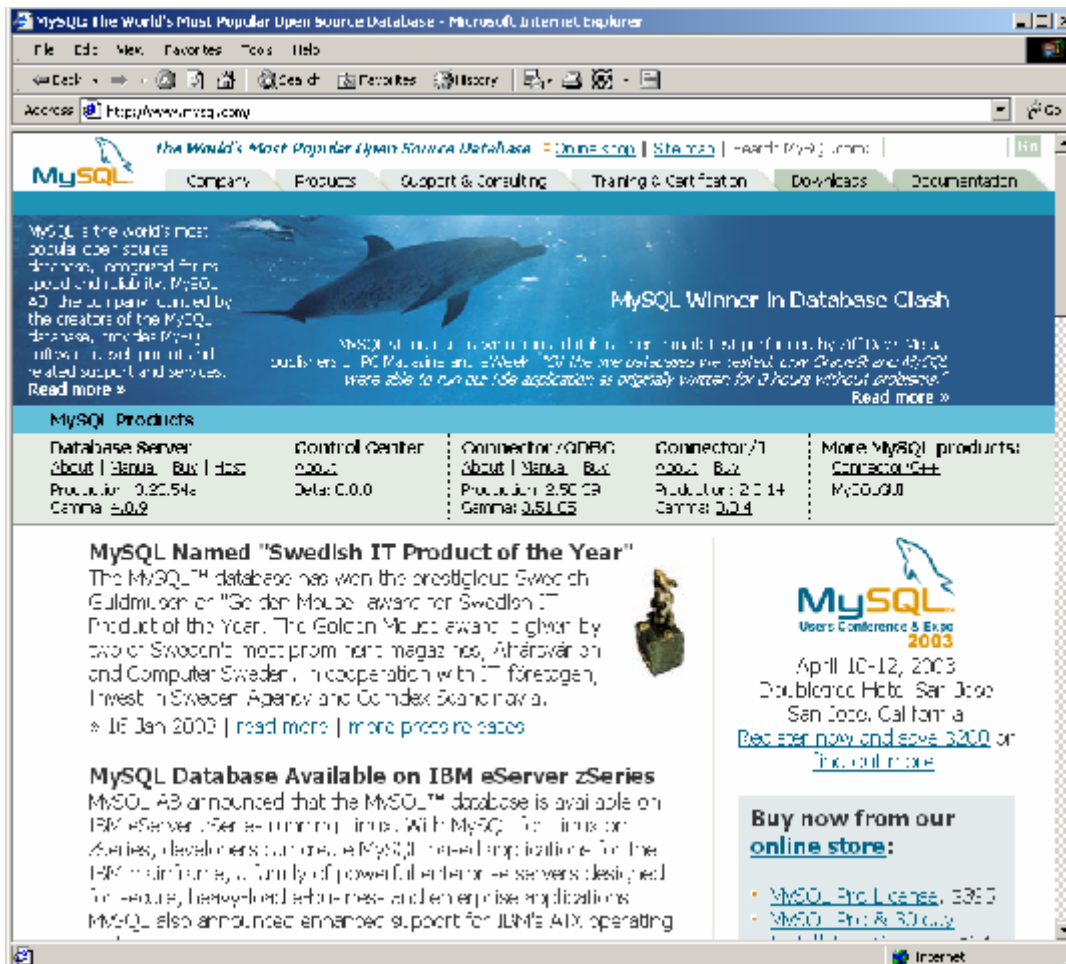
In order to run MySQL you must have local server running (e.g. Apache, MS information server IIS, PWS, or Omnihttpd), if you plan to make your database accessible via the web you will need to have an ISP who supports MySQL – there is usually an additional charge for this service. If you are using TELUS as a host it costs about \$50/year and you can host 2 databases on your server. Also if you are running MySQL off a local server you must make sure the php.ini file is configured so it will permit MySQL access – you can check this by running simple `<?php phpinfo(); ?>` and checking the status.



MySQL Support	enabled
Active Persistent Links	0
Active Links	0
Client API version	3.23.54
MYSQL_MODULE_TYPE	external
MYSQL_SOCKET	/tmp/mysql.sock
MYSQL_INCLUDE	-I/usr/local/mysql/include/mysql
MYSQL_LIBS	-L/usr/local/mysql/lib/mysql -lmysqlclient

Installing MySQL on Win98/2000 computer by R. Berdan 20/1/2003

1. Go to web site www.mysql.com and select the downloads tab.

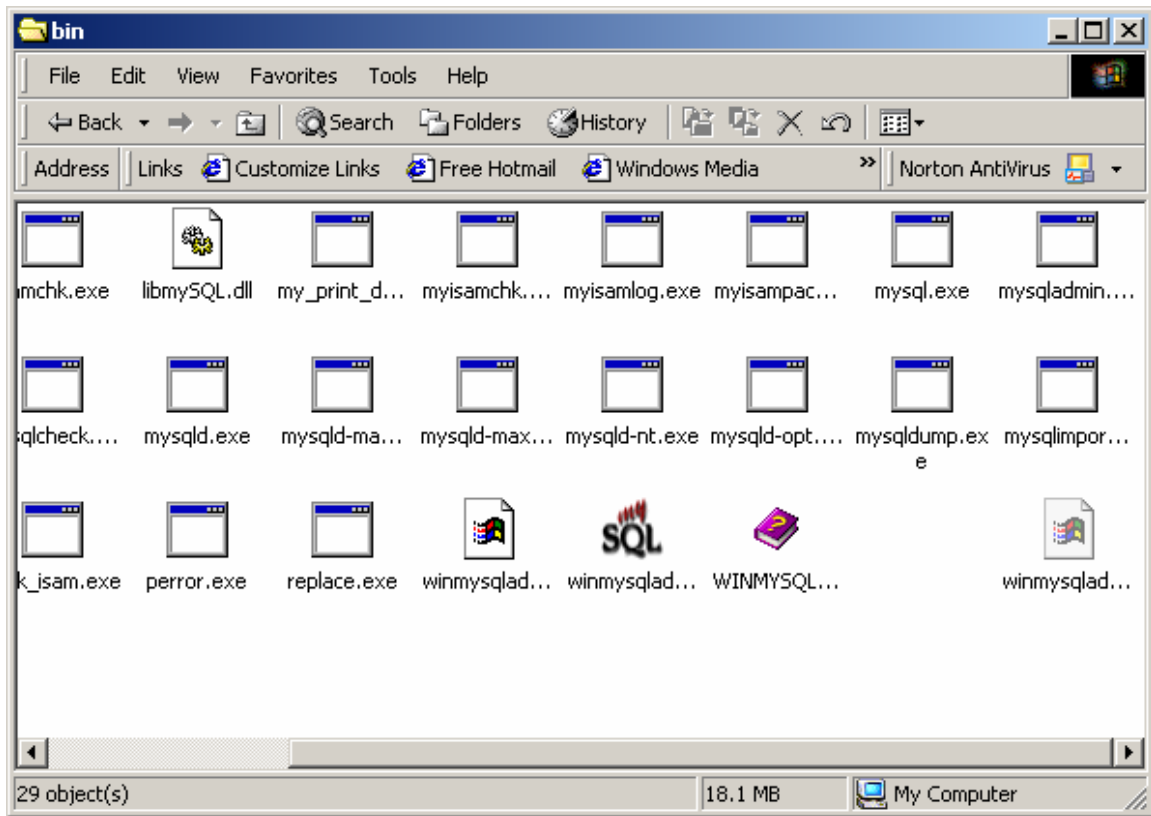


2. On the downloads page select

- **MySQL database server & standard clients:**
 - [MySQL 3.23](#) -- Stable release (recommended)
 - [MySQL 4.0](#) -- Gamma release (use this for new

MySQL 3.23 – stable release version, right click, save target as and copy the zip file to a folder on your computer.

3. Unzip the files using WinZip or other unzip utility, you can place the unzipped files in the same folder you downloaded the compressed file to.
4. Find the setup.exe file and double click on it to begin the installation. Recommended you have any web server software turned off, and any other programs running in your window it is recommended you turn them off as well.

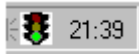


5. Run installer> selecting the default values and if it asks you where you want the files again I recommend the default c:\mysql it will create the folder for you. When its finished go to the directory c:\mysql\bin where the binary (exe files are stored). Find the mysqladmin.exe icon (see above) > double click on the icon and it will configure the MySQL program. You will be asked to enter a username and password.

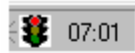


Eg. Username: rob Password xxxxxx

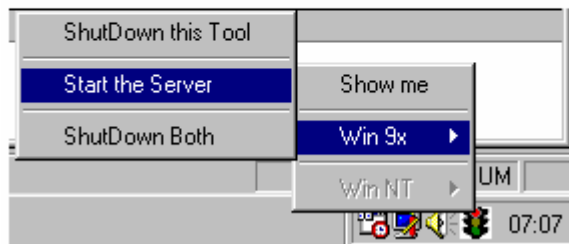
When you are finished – you should see a red a small stop light icon in the lower right corner of your task bar indicating the MySQL is running and ready. I would recommend that you make a short cut to the winmysqladmin.exe icon and the winmysqladmin.hlp file (help reference, book icon with ? on cover) and place the shortcuts on your desktop.



Note stop light icon with green light indicates that MySQL is up and running.

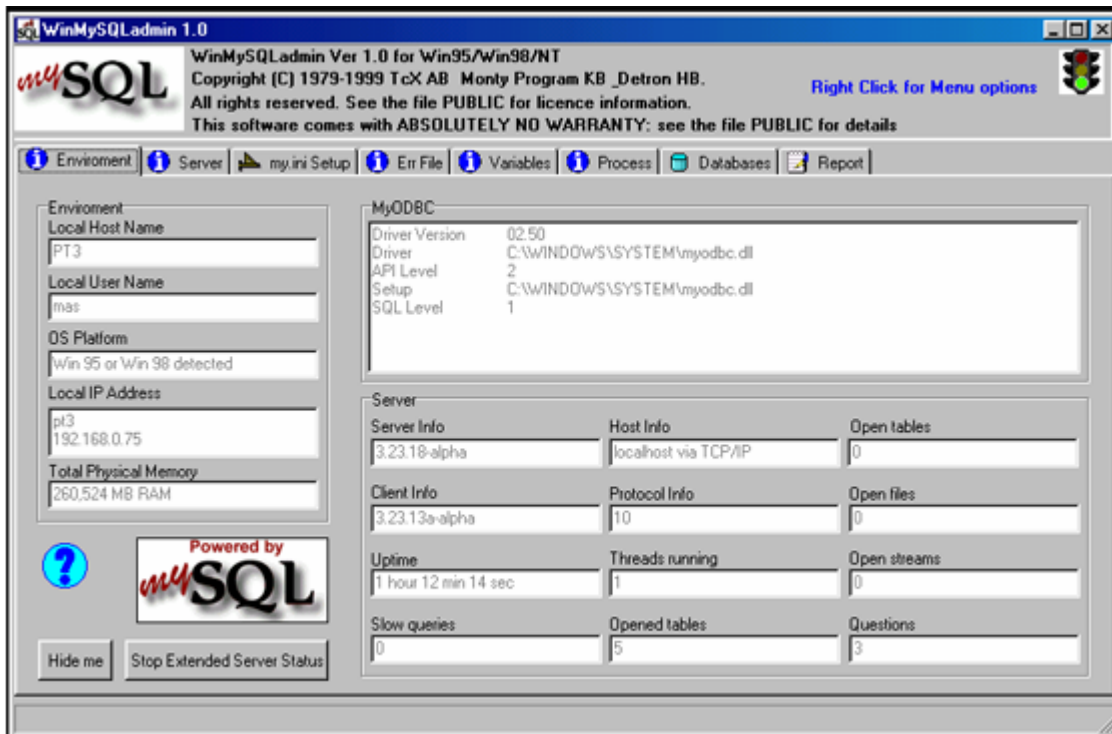


The red light means that the MySQL Server is stopped.



You can turn the program off and on by clicking on the stop light icon.

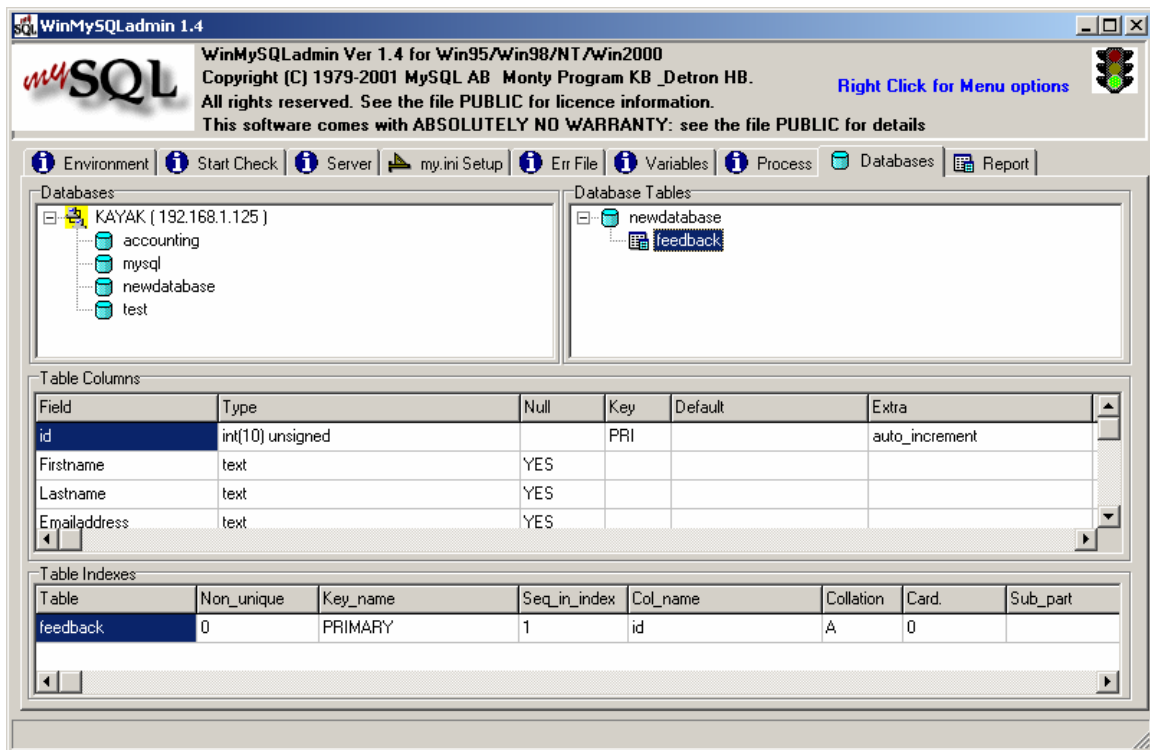
To access the main screen, click on the 'Show Me' menu item.



See the help for more information about the admin features shown above.

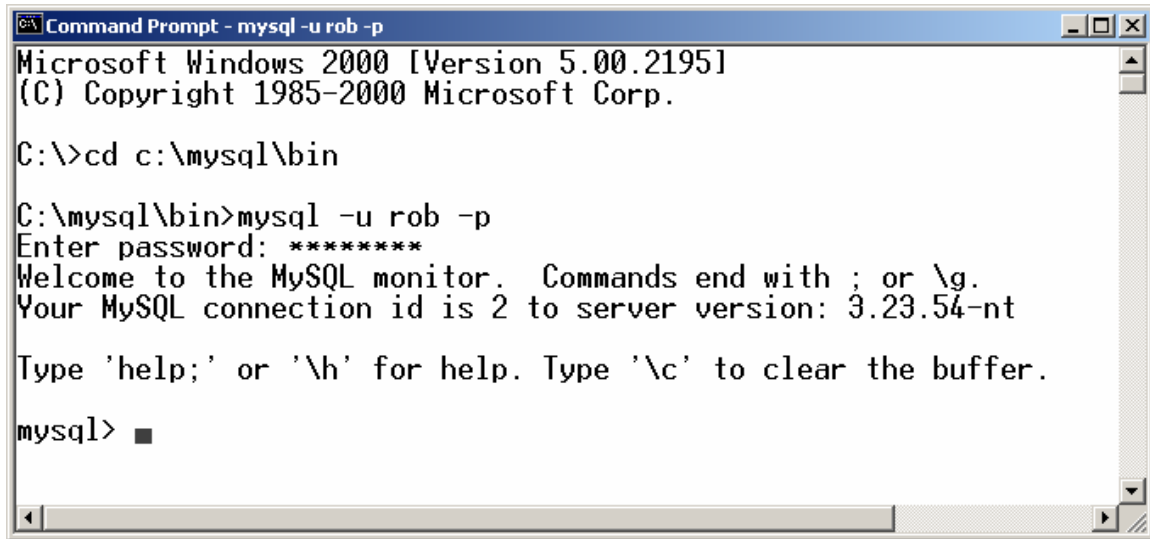
Starting and stopping MySQL

1. You can start MySQL by clicking on the winmysqladmin icon in the c:\mysql\bin directory (I like to make a shortcut and put it on my desktop). You will see the traffic light icon in the lower right task bar with a green stop light turned on - if MySQL is running. See installations instructions. The screen below shows the databases available, their names and table layout – you view winMySQLAdmin by selecting the traffic light icon in the task bar and selecting “show me” from the pop up men.



The tabs at the top can be selected to view databases, and various other forms of information. The window above shows there are 4 databases: accounting, mysql, newdatabase, test on my Kayak computer. Two of the databases: mysql and test are preinstalled with the software.

2. The second way to work with MySQL is run in command line mode. You can start the command line window by Start>Programs>Accessories>Command Prompt or you can select Start>run>cmd>OK - you can customize the appearance of the window, background color, font and window size by clicking on the upper left icon when the window opens.



```
Command Prompt - mysql -u rob -p
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>cd c:\mysql\bin

C:\mysql\bin>mysql -u rob -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 3.23.54-nt

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

mysql> ■
```

You need to change to the directory c:\mysql\bin then log on

```
C:\mysql\bin>mysql -u username -p  
Enter password: password
```

-u option indicates you need to follow with your username that you set up with MySQL when you installed it – you can view this information using the Admin window on the previous page, select my.ini setup and you will see your username and password.

-p option indicates you need to enter your password, you will be prompted on the next line, the password will not be echoed to the screen.

Once your MySQL server is up and running you are now ready to build or modify a database.

Creating & Building a Database with MySQL

You can create a database and modify it is to type in SQL commands directly at the prompt and or you can use PHP scripts to create and modify the database. SQL, short for Structured Query Language, is a group of special words used exclusively for interacting with databases. Every major database uses, SQL including MySQL.

For the full Tutorial on creating a sample Accounting Database see (L.Ullman, 2003, MySQL, Peachpit Series Press, Chapter 4 SQL, page 59).

The following is just to demonstrate how you would get started working in the command mode: (note you will have to have logged in with your username and password). You enter commands at the prompt, commands are terminated with a semicolon. You can also hit the carriage return to continue typing commands so they are easier to read and follow. The following example you will start create a new database called **accounting** then create a simple table. Note the commands are not case sensitive but I will use uppercase for the SQL commands. **mysql>command** is the prompt you will see when you are logged into MySQL in command mode.

General

```
mysql>CREATE DATABASE databasename
```

```
mysql>CREATE TABLE tablename(column1name, description, colum2name, description)
```

Specific Example

```
mysql>CREATE DATABASE accounting;  
USE accounting;  
Query OK, 1 row affected (0.11 sec)
```

```
mysql> CREATE TABLE invoices(  
->invoice_id SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT,  
->client_id SMALLINT(3) UNSIGNED,  
->invoice_date DATE NOT NULL,  
->invoice_amount DECIMAL(10,2) UNSIGNED NOT NULL,  
->invoice_description TINYTEXT,  
->PRIMARY KEY (invoice_id),  
->INDEX (invoice_date)  
->);  
Query OK, 0 rows affected (0.42 sec)
```

```
mysql>
```

Notes: hit your enter key to add the carriage returns -> so code is easier to read
Code is not case sensitive, SQL commands are in UPPERCASE for presentation purposes only.
Every SQL command must be terminated with a semicolon i.e. see last ;
MySQL will report if a command worked using the **Query OK statement**
The **USE** statement allows you to select the specific database you want to work on. If you are working on an existing database you must **USE databasename** before you can modify or view it.

```
Command Prompt - mysql -u rob -p

mysql> use accounting;
Database changed
mysql> show tables;
+-----+
| Tables_in_accounting |
+-----+
| invoices              |
+-----+
1 row in set (0.00 sec)

mysql> █
```

USE command to select a database,
SHOW command to view tables.

mysql>SHOW columns from invoices;

```
Command Prompt - mysql -u rob -p

mysql> show tables;
+-----+
| Tables_in_accounting |
+-----+
| invoices              |
+-----+
1 row in set (0.01 sec)

mysql> show columns from invoices;
+-----+-----+-----+-----+-----+-----+
| Field          | Type                | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| invoice_id    | smallint(4) unsigned |      | PRI | NULL    | auto_ |
| increment     |                      |      |     |         |       |
| client_id     | smallint(3) unsigned | YES  |     | NULL    |       |
| invoice_date  | date                |      | MUL | 0000-00-00 |      |
| invoice_amount | decimal(10,2) unsigned |      |     | 0.00    |       |
| invoice_description | tinytext           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

COMMONLY USED MySQL COMMANDS used from the cmd prompt

Inserting DATA

Once your database and tables have been created you can start to populate them with the INSERT command.

```
mysql> INSERT INTO tablename(column1, column2 _)  
->('value1', 'value2');
```

Using this structure you can add rows of records, populating only the columns that matter. Any columns not given a value will be treated as NULL.

E.g.

```
mysql> INSERT INTO expense_categories  
->(expense_category) VALUES  
->('Travel-Hotel');
```

Note: values containing a single quotation should be escaped e.g. O\'Malley
Terms INTO and INSERT are optional in current versions of MySQL

Selecting DATA

```
mysql>SELECT * FROM tablename
```

e.g.

```
mysql>SELECT user_id, first_name, last_name FROM users;
```

```
mysql>SELECT * FROM expense_categories
```

SELECT is used to return all rows of records based on certain criteria.

Sorting Query Results

```
mysql>SELECT * FROM tablename ORDER BY column
```

e.g.

```
mysql>SELECT invoice_amount, invoice_date FROM  
->invoices ORDER BY invoice_date;
```

Updating Data

To change existing records for example if the data entered was incorrect or a client moved to a new location and you need to change their contact information.

```
mysql>UPDATE tablename SET column=value
```

You can adjust multiple columns at one time, separating each by a comma

```
mysql>UPDATE tablename SET column1='value', column2='value'....
```

Deleting Data

General

```
mysql>DELETE FROM tablename WHERE column=value
```

e.g.

```
mysql>DELETE FROM expense_categories WHERE  
-->expense_category_id = 1 OR expense_category_id=2;
```

Modifying Tables

The ALTER SQL keyword is used to modify the structure of the table, add, delete or change columns therein. It also applies to renaming the table as a whole and altering the keys and indexes.

General

```
mysql>ALTER TABLE tablename CLAUSE
```

e.g.

```
mysql>ALTER TABLE clients ADD COLUMN contact_last_name VARCHAR(25) AFTER  
-->contact_firstname;
```

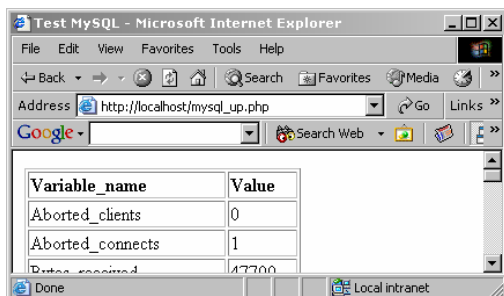
Although you can create and edit databases in command mode – you can also use PHP to create, modify and read from a database. We will do this in an another tutorial.

Testing whether MySQL is working on your machine

Create the file below save it into your wwwroot folder and name it mysql_up.php call the file if MySQL is working you will see a table generated on your web page. If this does not work you may have to substitute \$host \$user \$password variables with those on your server.

```
<html>
<head>
<title>Test MySQL</title>
<body>
<!-- mysql_up.php -->
<?php
$host="localhost";
$user="root";
$password="";
mysql_connect($host,$user,$password);
$sql="show status";
$result = mysql_query($sql);
if ($result == 0)
    echo("<b>Error " . mysql_errno() . ": " . mysql_error() .
"</b>");
elseif (mysql_num_rows($result) == 0)
    echo("<b>Query executed successfully!</b>");
else
{
?>
<!-- Table that displays the results -->
<table border="1">
<tr><td><b>Variable_name</b></td><td><b>Value</b></td>

></tr>
<?php
    for ($i = 0; $i < mysql_num_rows($result); $i++) {
        echo("<TR>");
        $row_array = mysql_fetch_row($result);
        for ($j = 0; $j < mysql_num_fields($result); $j++) {
            echo("<TD>" . $row_array[$j] . "</td>");
        }
        echo("</tr>");
    }
?>
</table>
<?php } ?>
</body>
</html>
```



The screenshot shows a web browser window titled "Test MySQL - Microsoft Internet Explorer". The address bar shows "http://localhost/mysql_up.php". The main content area displays a table with two columns: "Variable_name" and "Value". The table contains the following data:

Variable_name	Value
Aborted_clients	0
Aborted_connects	1
Bytes_received	47700

Screen shot showing part of the table.

Creating a Simple Database with PHP and MySQL that stores information input into an HTML form.

This tutorial provides you with a series of steps required to create a simple MySQL database that accepts information from a form and stores it in a table (MySQL database). In order to complete this tutorial you will need to have a local server (e.g. IIS), PHP 4+ version and MySQL installed on your machine or have access to a server that supports these features. This tutorial is based on L. Ullmans's tutorial in his book PHP page 183-200, by Peachpit press.

There are 5 steps

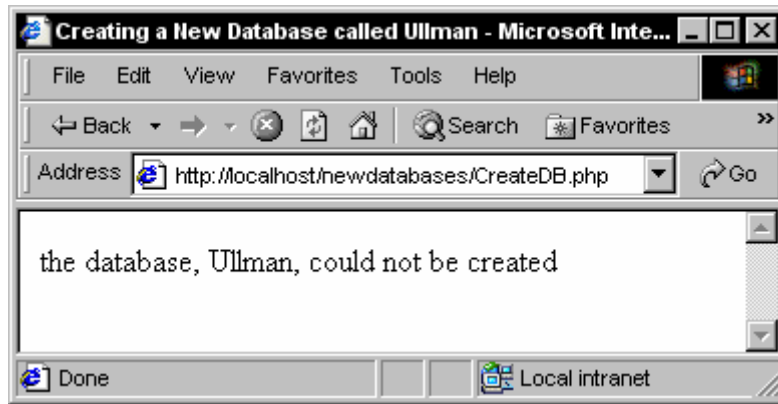
- 1) Create a database using PHP (php document)
- 2) Create a table to insert the data (php document)
- 3) Create the form to collect the data (html document)
- 4) Create a form handler (php document)
- 5) Create a file that calls and displays the data in the database

1) Create a database using PHP

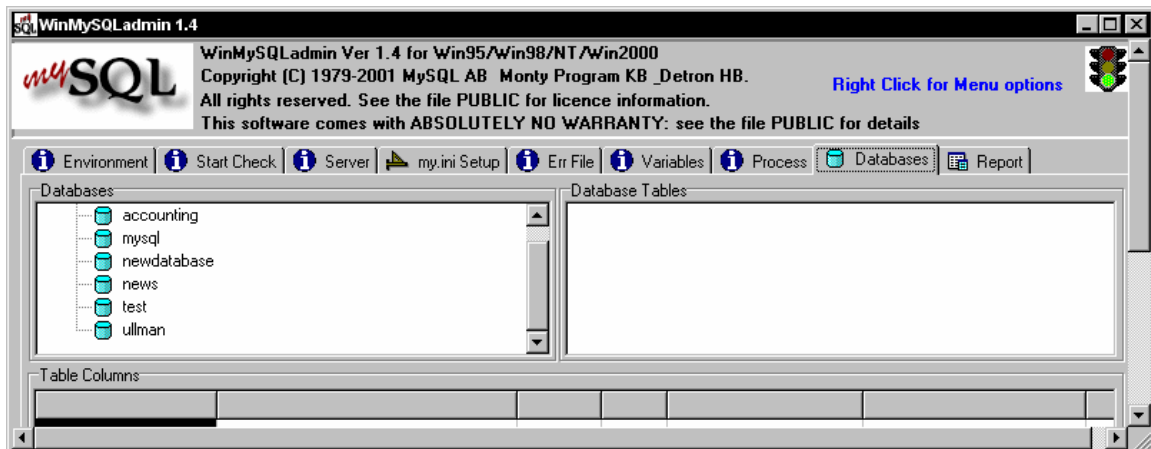
```
<html>
<head>
<title>Creating a New Database called Ullman</title>
</head>
<body>
<?php
$Host = "localhost"; // account if on local host alternatively use url or IP address
$User = "rob"; // user name
$Password = "helisoma"; // MySQL password
$DBname = "Ullman"; // database name can be anything
$link = mysql_connect($Host, $User, $Password); //
if (mysql_create_db($DBname, $link))
{
    print "The database, $DBname, was successfully created <br>\n";
}
else
{
    print "the database, $DBname, could not be created <br>\n";
}
mysql_close($link); // close the database
?>
</body>
</html>
```

save file as **CreateDB.php** (you can use all lower case if you prefer)

Creating a database consists of 3 steps 1) linking to the database server 2) create database 3) close database. Host, user and password were assigned to variables so it is easy to change them. Save all files to the same location in a folder inside the root of your local host server.



The first time you create the database it will say The database Ullman was successfully created. If you try to create it again you will get the message above. You can verify the database exists by using the MySQL manager.



Available databases that have been created are shown in the window when the databases tab is selected in the MySQL manager.

2) Create a Table that will store your data in the database you created

```
<html>
<head>
<title>Creating a Table</title>
</head>
<body>

<?php
// Set the variables for the database access;

$Host = "localhost";
$User = "rob";
$Password = "helisoma";
$DBname = "Ullman";
$TableName = "Feedback";

$link= mysql_connect ($Host, $User, $Password);
$query = "CREATE table $TableName (id INT UNSIGNED NOT NULL AUTO_INCREMENT
PRIMARY KEY, FirstName TEXT, LastName TEXT, EmailAddress TEXT, Comments TEXT)";
if (mysql_db_query($DBname, $query, $link))
{
print "The query was successfully executed!<br>\n";
}
else
{
print "The query could not be executed!<br>\n";
}
mysql_close($link);
?>

</body>
</html>
```

Save as **CreateTable.php**

Primary Key	FirstName	LastName	EmailAddress	Comments
1				
2				
3				

The organization of the table will look like that above. The Primary key will auto_increment as you enter new data. When you load the page the first time it should say "query was successfully executed". If you load the page a second time it will say query could not be executed since the table has already been created.

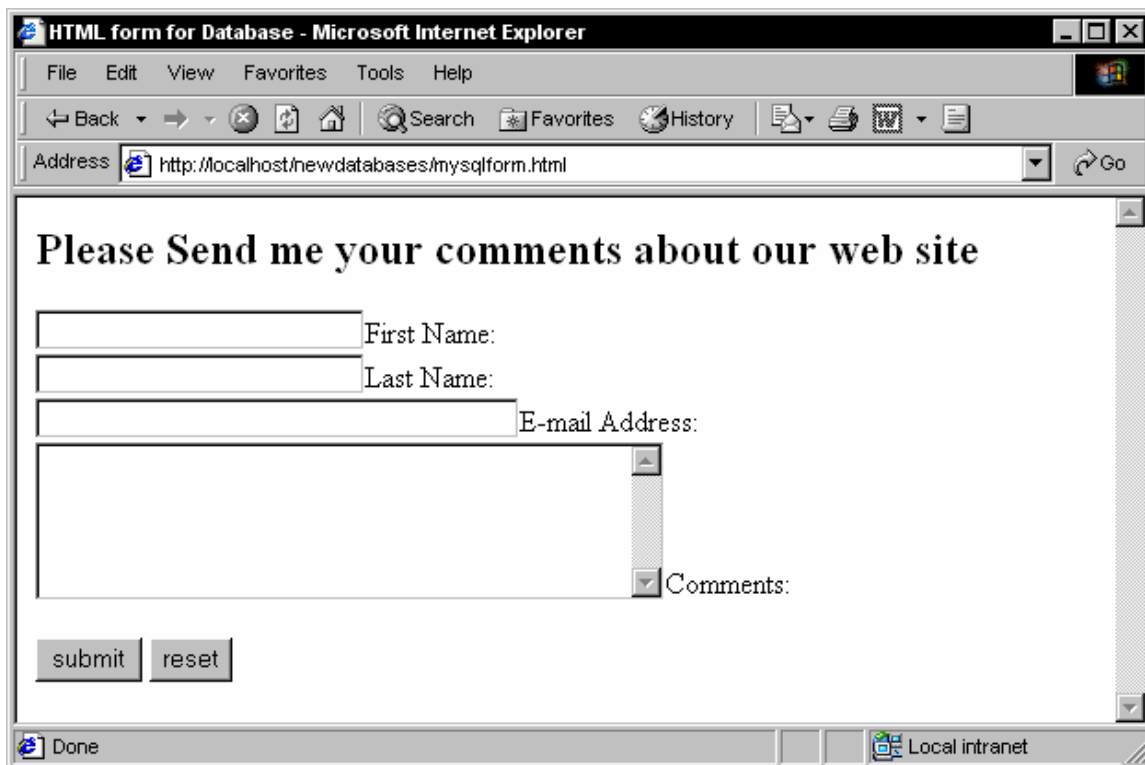
3. Create your form used to enter the data.

```
<html>
<head>
<title>HTML form for Database</title>
</head>
<body>

<h2>Please Send me your comments about our web site</h2>
<form action="handleform.php" method="post">
<input type="text" name="Array[FirstName]" size="26">First Name:<br>
<input type="text" name="Array[LastName]" size="26">Last Name: <br>
<input type="text" name="Array[Email]" size="40">E-mail Address:<br>
<textarea name="Array[Comments]" rows=5 cols=40></textarea>Comments:<br><br>
<input type="submit" name="submit" value="submit">
<input type="reset" name="reset" value="reset">
</form>

</body>
</html>
```

In this form the data is collected with each name element assigned to an array. Save the file as **mysqlform.html**



Above screen shot showing the form

4. Create a script to process the form data and send the data to the database

```
<html>
<head>
<title>Form handler</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<?php
// this page processes the form data and sends it to a MySQL database called Ullman
// trim function removes white space before and after entered text
$Array["FirstName"] = trim($Array["FirstName"]);
$Array["LastName"] = trim($Array["LastName"]);
$Array["Email"] = trim($Array["Email"]);
$Array["Comments"] = trim($Array["Comments"]);
//set the variables for the database
$Host = "localhost";
$User = "rob";
$Password = "helisoma";
$DBname = "Ullman";
$TableName = "Feedback";

$Link = mysql_connect($Host, $User, $Password);
$query = "INSERT into $TableName values ('0', '$Array[FirstName]', '$Array[LastName]',
'$Array[Email]', '$Array[Comments]')";
print "The query is: <br>$query<p>\n";
if (mysql_db_query($DBname, $query,$Link))
{
print "The query was successfully executed!";
}
else
{
print "The query could not be executed";
}
mysql_close($Link);
?>
</body>
</html>
```

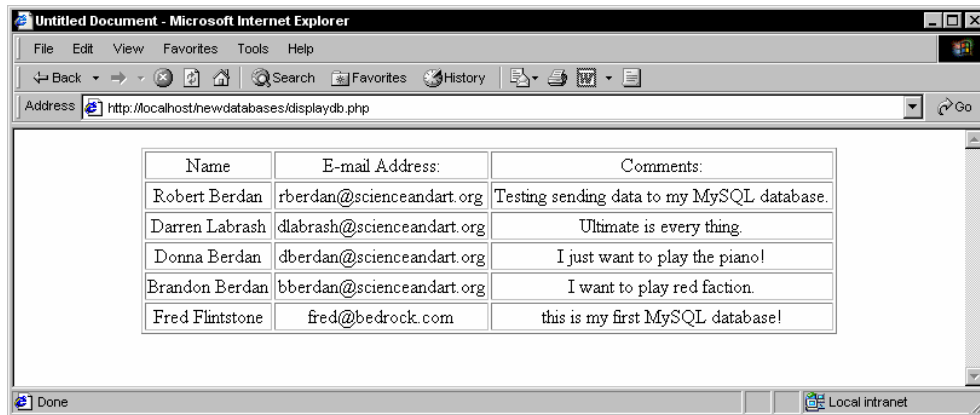
Save the file as **handleform.php**

5. **Finally you will create a page that selects all the data submitted by your form into the database.** Before you run this this page you should fill out your form and submit data using the a couple off times. Save the file as **displaydb.php**

```

<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<?php
// set the variables for teh database access;
$Host = "localhost";
$User = "rob";
$Password = "helisoma";
$DBname = "Ullman";
$TableName = "Feedback";
$Link = mysql_connect($Host,$User, $Password);
$query = "SELECT * from $TableName";
$result = mysql_db_query($DBname,$Query,$Link);
//Create a table
print "<table border=1 width='75%' align='center'>";
print "<tr align='center' valign='top'>\n";
print "<td align='center' valign='top'>Name</td>\n";
print "<td align='center' valign='top'>E-mail Address:</td>\n";
print "<td align='center' valign='top'>Comments: </td>\n";
print "</tr>\n";
//Fetch the results from tha database
while ($Row=mysql_fetch_array($Result))
{
print "<tr align=center valign=top>\n";
print "<td align=center valign=top>$Row[FirstName] $Row[LastName]</td>\n";
print "<td align=center valign=top>$Row[EmailAddress]</td>\n";
print "<td align=center valign=top>$Row[Comments]</td>\n";
print "</tr>";
}
mysql_close($Link);
print "</table>";
?>
</body>
</html>

```



Screen shot displaying the data submitted to the database.

Note when creating forms dynamically with php you can either escape double quotes " , or use a single ' quote.

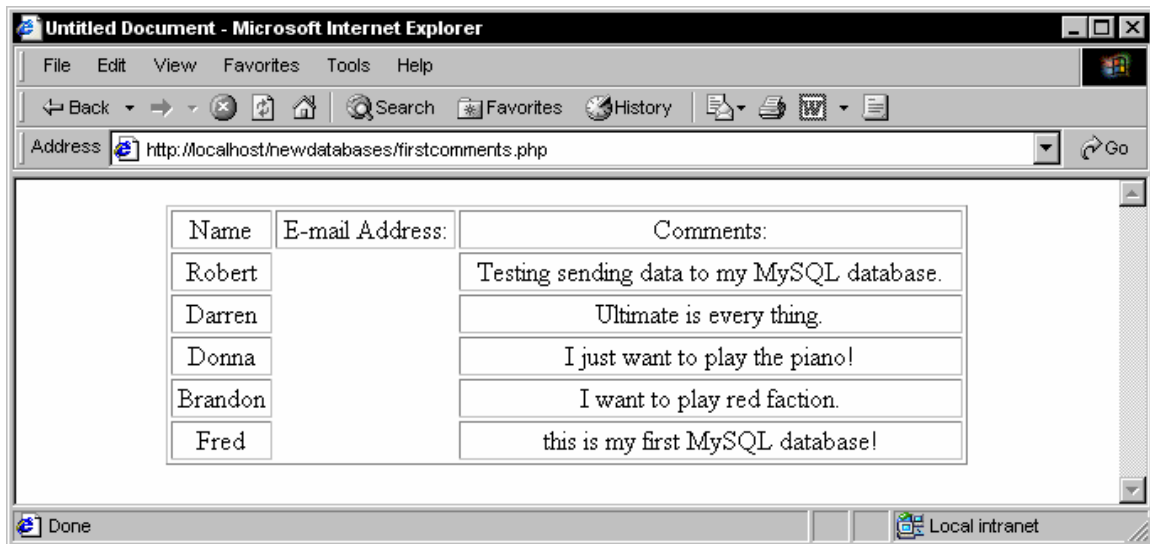
e.g. name="firstname" or name='firstname' both work but you can not use "" alone.

In the last script we used:

```
$Query = "SELECT * from $TableName";
```

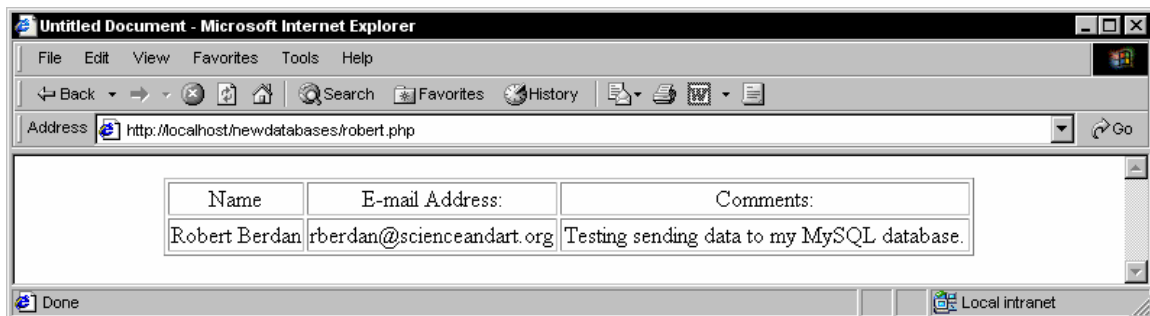
This statement says select all data (*) from the \$TableName. You can modify this statement to select only parts of the data for example

```
$Query ="SELECT FirstName, Comments from $TableName
```



Screen shot after Selecting Firstname and Comments fields. File saved as **firstcomments.php**

```
$Query = "SELECT * from $TableName where (FirstName='Robert')";
```



The Next Step in the process is to build a form that submits a query to the database to allow users to select the specific information they wish to display i.e. Name, E-mail Address and Comments of an individual or display all the data and print it to the screen.

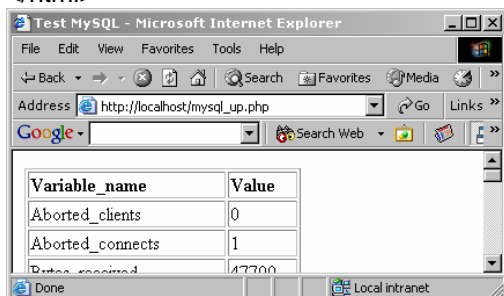
APPENDIX – script to test if MySQL is working properly

Create the file below save it into your wwwroot folder and name it mysql_up.php call the file if MySQL is working you will see a table generated on your web page. If this does not work you may have to substitute \$host \$user \$password variables with those on your server.

```
<html>
<head>
<title>Test MySQL</title>
<body>
<!-- mysql_up.php -->
<?php
$host="localhost";
$user="root";
$password="";
mysql_connect($host,$user,$password);
$sql="show status";
$result = mysql_query($sql);
if ($result == 0)
    echo("<b>Error " . mysql_errno() . ": " . mysql_error() .
" </b>");
elseif (mysql_num_rows($result) == 0)
    echo("<b>Query executed successfully!</b>");
else
{
?>
<!-- Table that displays the results -->
<table border="1">

<tr><td><b>Variable_name</b></td><td><b>Value</b></td>

></tr>
<?php
for ($i = 0; $i < mysql_num_rows($result); $i++) {
    echo("<TR>");
    $row_array = mysql_fetch_row($result);
    for ($j = 0; $j < mysql_num_fields($result); $j++) {
        echo("<TD>" . $row_array[$j] . "</td>");
    }
    echo("</tr>");
}
?>
</table>
<?php } ?>
</body>
</html>
```



The screenshot shows a web browser window titled "Test MySQL - Microsoft Internet Explorer". The address bar shows "http://localhost/mysql_up.php". The main content area displays a table with two columns: "Variable_name" and "Value". The table contains the following data:

Variable_name	Value
Aborted_clients	0
Aborted_connects	1
Bytes_received	47700

Screen shot showing part of the table.