Faculty of Science and Technology

Department of Computer Science

# WEB TECHNOLOGIES (CSC 3222)

Lecture Note 6

Week 6

Sazzad Hossain
sazzad@aiub.edu

## Contents

## PHP MySQL Introduction

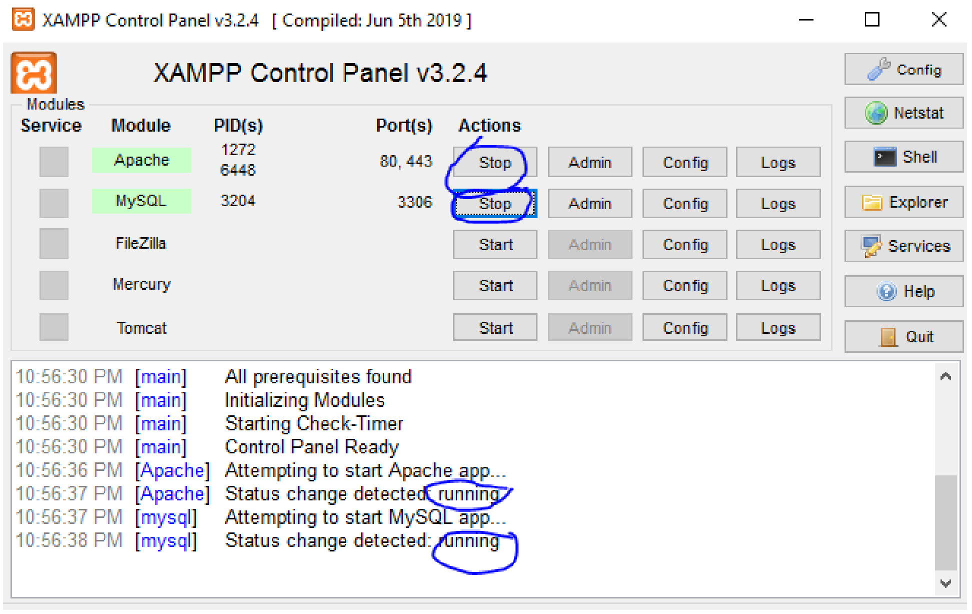
MySQL is one of the most popular relational database system being used on the Web today. It is freely available and easy to install, however if you have installed Xampp it already there on your machine. MySQL database server offers several advantages:

- MySQL is easy to use, yet extremely powerful, fast, secure, and scalable.
- MySQL runs on a wide range of operating systems, including UNIX or Linux, Microsoft Windows, Apple Mac OS X, and others.
- MySQL supports standard SQL (Structured Query Language).
- MySQL is ideal database solution for both small and large applications.MySQL is developed, and distributed by Oracle Corporation.
- MySQL includes data security layers that protect sensitive data from intruders.
- MySQL database stores data into tables like other relational database. A table is a collection of related data, and it is divided into rows and columns.
- Each row in a table represents a data record that are inherently connected to each other such as information related to a particular person, whereas each column represents a specific field such as id, first_name, last_name, email, etc.

## Installing MySQL and Using phpMyAdmin

### MySQL Installation

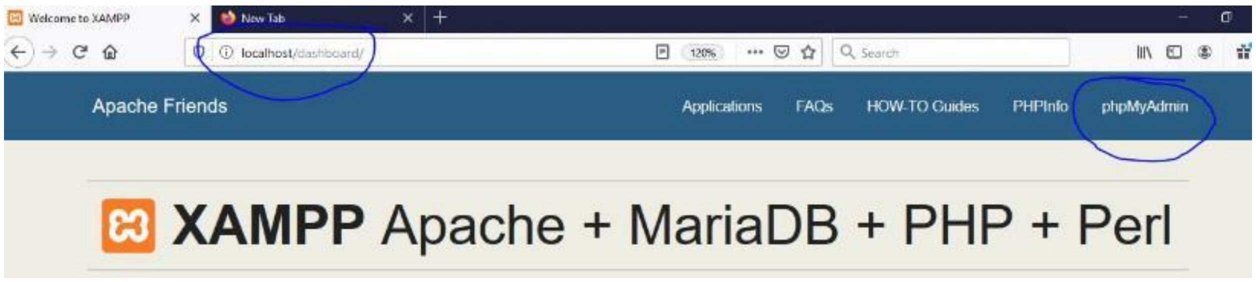
Open XAMPP control panel and make sure that Apache and MySQL service is running


XAMPP Control Panel v3.2.4 [ Compiled: Jun 5th 2019 ]
XAMPP Control Panel v3.2.4
Config
Netstat
Shell
Explorer
Services
Help
Quit
Modules

| Service | Module | PID(s) | Port(s) | Actions | | | |
|---|---|---|---|---|---|---|---|
| | Apache | 1272 6448 | 80, 443 | Stop | Admin | Config | Logs |
| | MySQL | 3204 | 3306 | Stop | Admin | Config | Logs |
| | FileZilla | | | Start | Admin | Config | Logs |
| | Mercury | | | Start | Admin | Config | Logs |
| | Tomcat | | | Start | Admin | Config | Logs |

```
10:56:30 PM  [main]    All prerequisites found
10:56:30 PM  [main]    Initializing Modules
10:56:30 PM  [main]    Starting Check-Timer
10:56:30 PM  [main]    Control Panel Ready
10:56:36 PM  [Apache]  Attempting to start Apache app...
10:56:37 PM  [Apache]  Status change detected: running
10:56:37 PM  [mysql]   Attempting to start MySQL app...
10:56:38 PM  [mysql]   Status change detected: running
```

Open a browser then type 'localhost' or 'localhost:8080' or 'localhost:8082'. Then navigate to 'phpMyAdmin'.



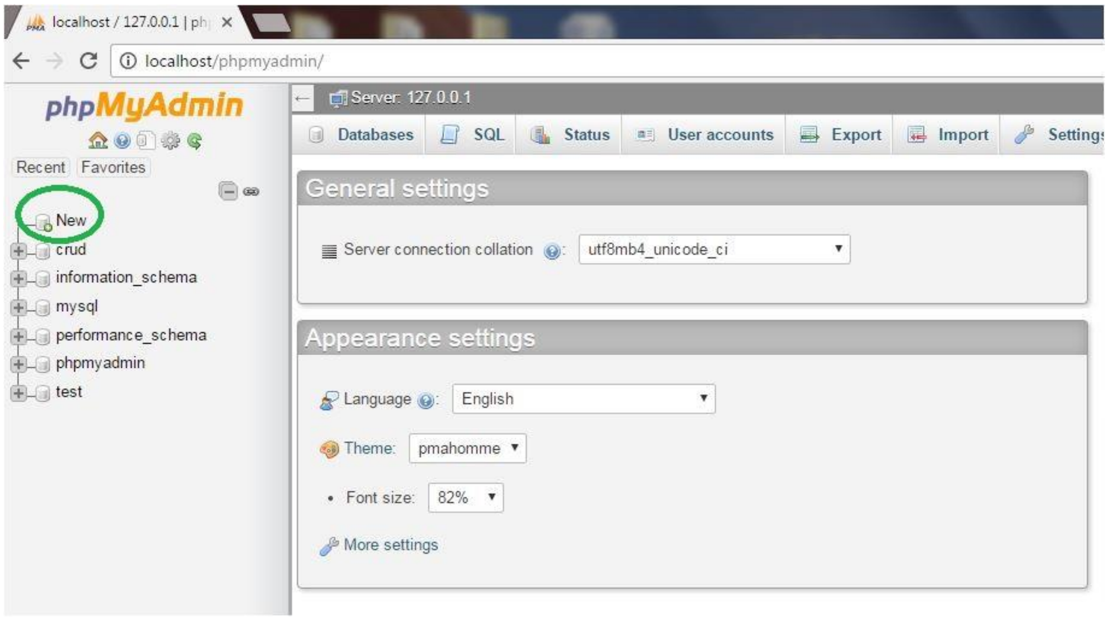
## What is phpMyAdmin

phpMyAdmin is a free tool to support the administrative operations of  MySQL. You may directly execute any SQL statement but phpMyAdmin provide an easy GUI to just execute the same statement with just a click. The administrative operations may include management of databases, columns, tables, indexes, users etc which can be performed easily through GUI of phpMyAdmin.

phpMyAdmin has a great documentation to help its users and the team of developers are always ready assist if users face any problem.

## Using phpMyAdmin

### Creating database using phpMyAdmin



Server connection collation: utf8mb4_unicode_ci

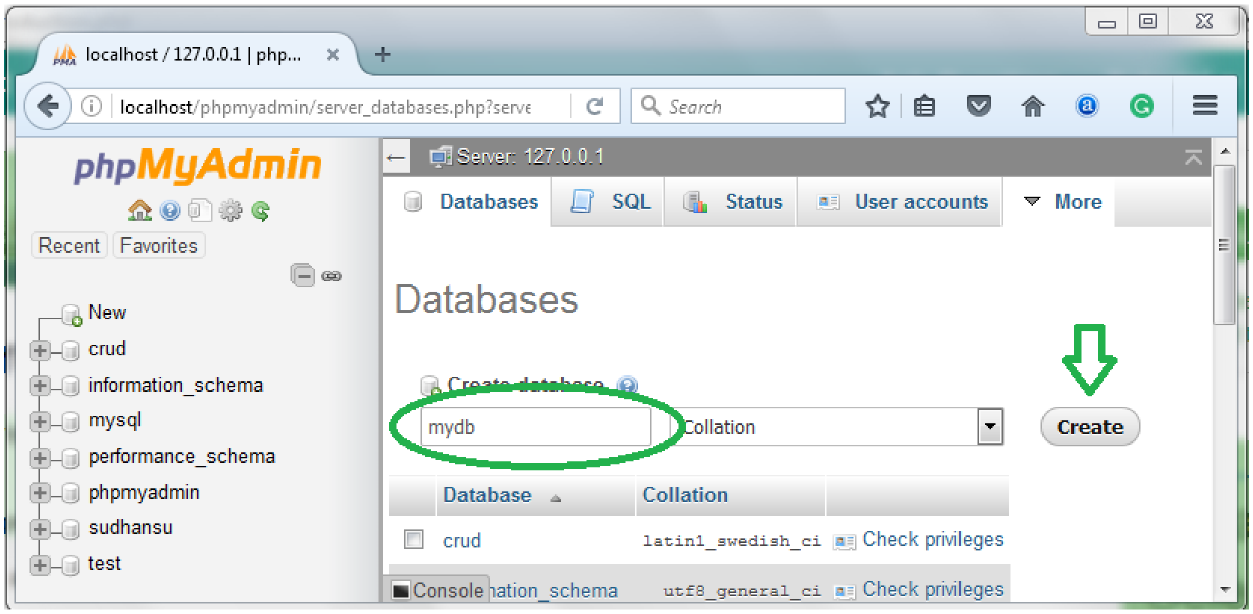
Language: English

Theme: pmahomme

Font size: 82%

Then click on the new button that is available on the sidebar of that page.

After click on the new button you find a page like that

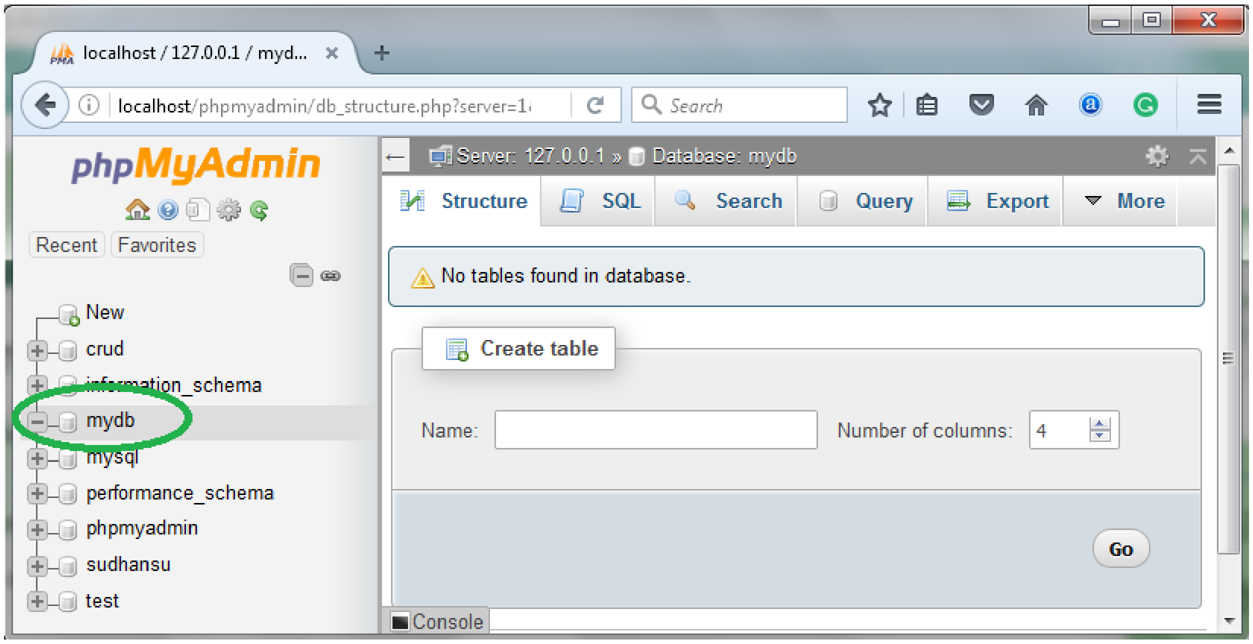


Here put a name of your own choice on the create database field and click on the **create** button and you get a successful message on the screen that you create database successfully and will appear in the database list.

*Note: You can't create table if you not have a database so first create database.*

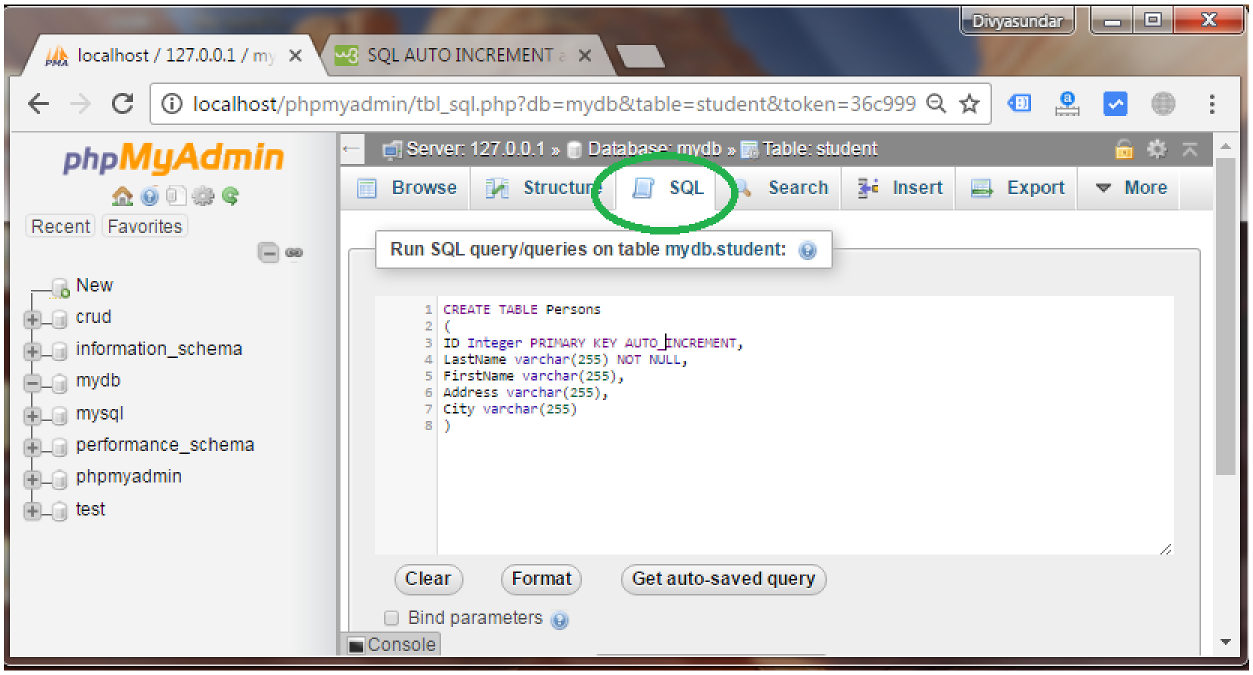
## Create table using query

Click on the database name in which under you create a table. After click on the database name you find a page like that.



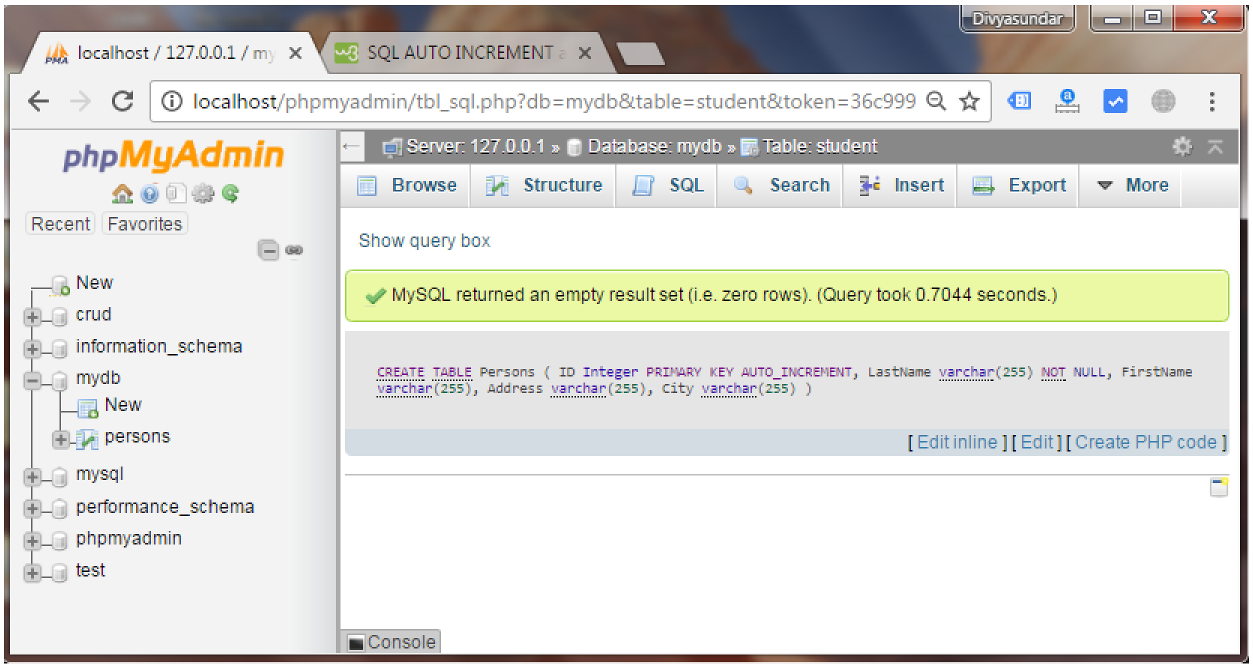
Here you have two option to create table first one is using **structure** and second one is using **SQL**.If you want to create table in structure option then put your table name on the create table name field and choose columns and click on the go button.

If you want to create a table by writing SQL Query simply click on the SQL button on the page and write your query and click on the go button.



```
CREATE TABLE Persons
(
ID Integer PRIMARY KEY AUTO_INCREMENT,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

If your SQL query is correct you find a page like that.

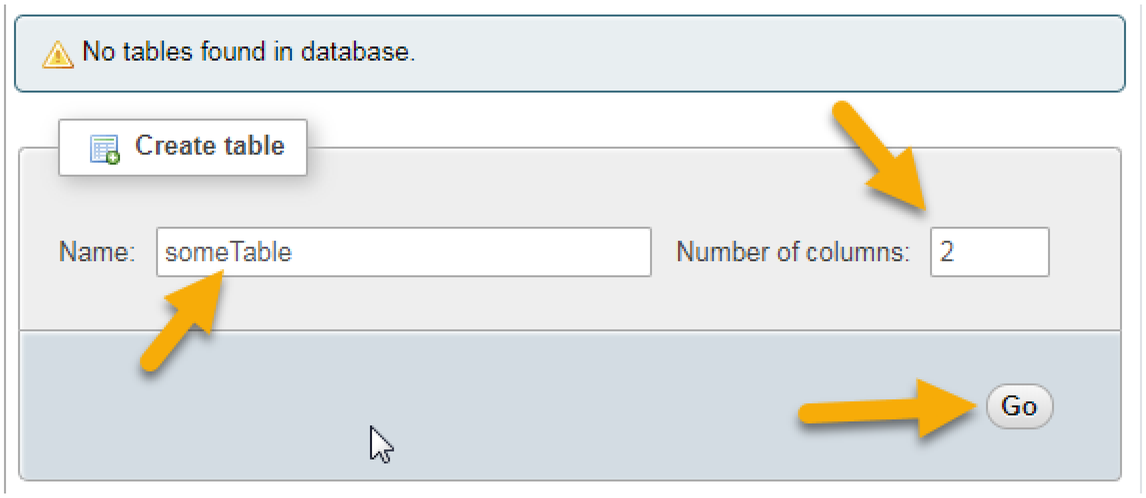

localhost / 127.0.0.1 / my
SQL AUTO INCREMENT
localhost/phpmyadmin/tbl_sql.php?db=mydb&table=student&token=36c999
phpMyAdmin
Recent
Favorites
New
crud
information_schema
mydb
New
persons
mysql
performance_schema
phpmyadmin
test
Server: 127.0.0.1 » Database: mydb » Table: student
Browse
Structure
SQL
Search
Insert
Export
More
Show query box
[ Edit inline ] [ Edit ] [ Create PHP code ]
Console

MySQL returned an empty result set (i.e. zero rows). (Query took 0.7044 seconds.)

```
CREATE TABLE Persons ( ID Integer PRIMARY KEY AUTO_INCREMENT, LastName varchar(255) NOT NULL, FirstName varchar(255), Address varchar(255), City varchar(255) )
```

## Create table using GUI

After creating table below image will pop up. To create a table with however many columns you want. Once you've decided that, click "Go" again:


No tables found in database.

Create table

Name: someTable

Number of columns: 2

This will require that you preplan your database a bit. You'll need to know exactly what you will be storing in it. For this example, we will be storing the user's name and age. So on the next screen we will put in a names and ages column. We will also need to give them a type, age will be INT (integer/numbers) and names will be VARCHAR (characters/letters). Finally we will need to say how many characters can be in each column. Age will be 3 since I don't think anyone would live to be older than 999 and we will give 100 characters for their name which should be enough. Once we filled in those fields, we would click "Save":

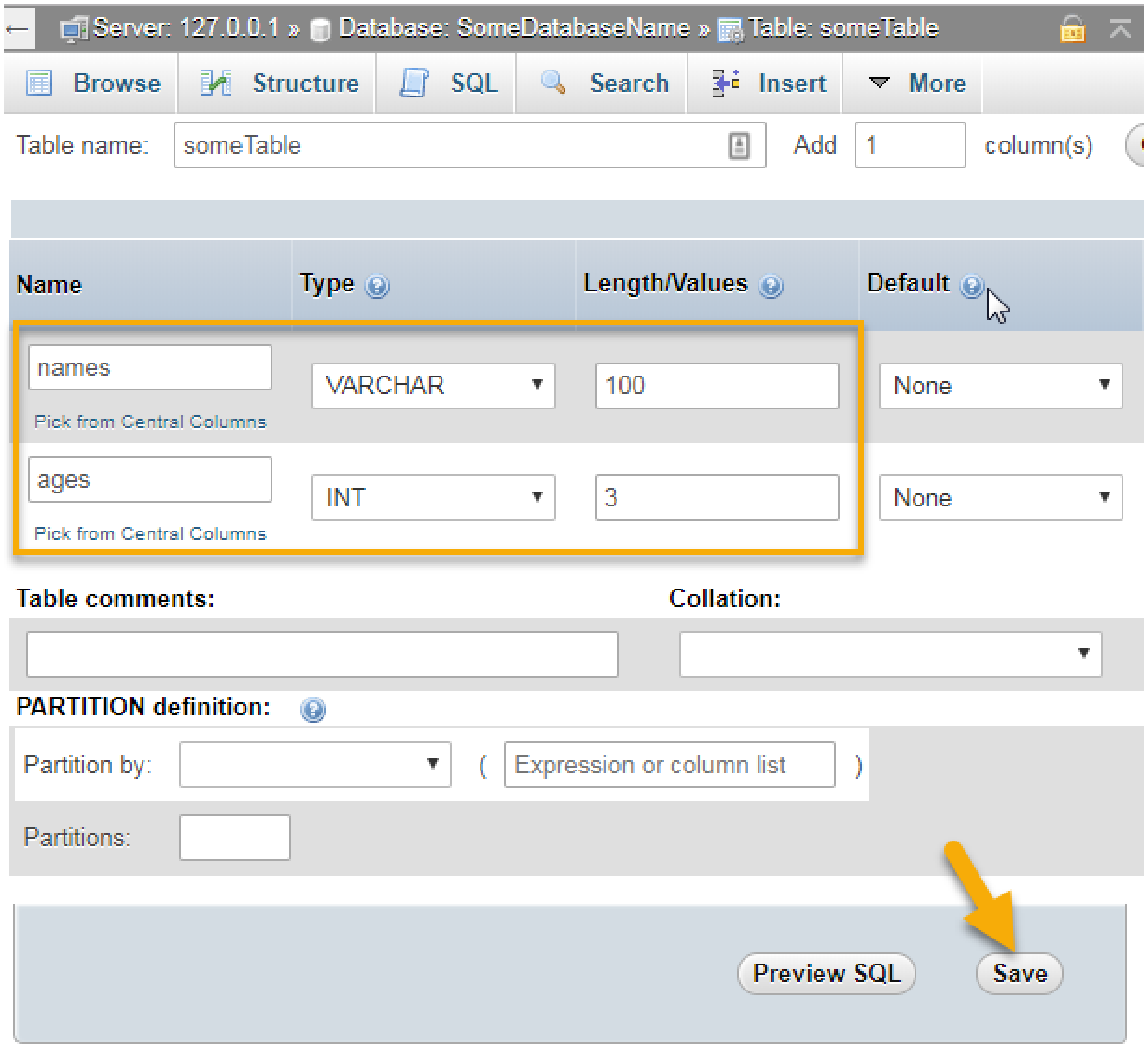

Server: 127.0.0.1 » Database: SomeDatabaseName » Table: someTable
Browse
Structure
SQL
Search
Insert
More
Pick from Central Columns
Preview SQL
Save

Table name: someTable Add 1 column(s)

| Name | Type | Length/Values | Default |
|---|---|---|---|
| names | VARCHAR | 100 | None |
| ages | INT | 3 | None |

Table comments:

Collation:

PARTITION definition:

Partition by: ( Expression or column list )

Partitions:

And there you have it! A new database with a table with two columns in it ready to be filled.

## PHP MySQL Connection

PHP 5 and later can work with a MySQL database using:

**MySQLi extension** (the "i" stands for improved)

**PDO (PHP Data Objects)**

Earlier versions of PHP used the MySQL extension. However, this extension was deprecated in 2012.

**MySQLi or PDO**

Both MySQLi and PDO have their advantages:

- PDO will work on 12 different database systems, whereas MySQLi will only work with MySQL databases.
- So, if you have to switch your project to use another database, PDO makes the process easy. You only have to change the connection string and a few queries. With MySQLi, you will need to rewrite the entire code - queries included.
- Both are object-oriented, but MySQLi also offers a procedural API.
- Both support Prepared Statements. Prepared Statements protect from SQL injection, and are very important for web application security.

**MySQL Examples in Both MySQLi and PDO Syntax**

In this, and in the following chapters we demonstrate three ways of working with PHP and MySQL:

- MySQLi (object-oriented)
- MySQLi (procedural)
- PDO

## Open a Connection to MySQL

Before we can access data in the MySQL database, we need to be able to connect to the server:

**Example (MySQLi Object-Oriented)**

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

**Example (MySQLi Procedural)**

```
<?php
$servername = "localhost";
```

```
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

**Example (PDO)**

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null


try {
  $conn = new PDO("mysql:host=$servername;dbname=myDB", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  echo "Connected successfully";
} catch(PDOException $e) {
  echo "Connection failed: " . $e->getMessage();
}
?>
```

**Note:** In the PDO example above we have also **specified a database (myDB)**. PDO require a valid database to connect to. If no database is specified, an exception is thrown.

**Tip:** A great benefit of PDO is that it has an exception class to handle any problems that may occur in our database queries. If an exception is thrown within the try{ } block, the script stops executing and flows directly to the first catch(){ } block.

### Close the Connection

The connection will be closed automatically when the script ends. To close the connection before, use the following:

**MySQLi Object-Oriented:**

```
$conn->close();
```

**MySQLi Procedural:**

```
mysqli_close($conn);
```

**PDO:**

```
$conn = null;
```

## PHP Create a MySQL Database

The CREATE DATABASE statement is used to create a database in MySQL.

The following examples create a database named "myDB":

### Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null

// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
  echo "Database created successfully";
} else {
  echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```

*Note: When you create a new database, you must only specify the first three arguments to the mysqli object (servername, username and password).*

## Example (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null

// Create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}

// Create database
$sql = "CREATE DATABASE myDB";
if (mysqli_query($conn, $sql)) {
  echo "Database created successfully";
} else {
  echo "Error creating database: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

## Example (PDO)

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
```

```
$password = "";//DB default password in empty/null

try {
  $conn = new PDO("mysql:host=$servername", $username, $password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "CREATE DATABASE myDBPDO";
  // use exec() because no results are returned
  $conn->exec($sql);
  echo "Database created successfully<br>";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}

$conn = null;
?>
```

*Note: The following PDO example create a database named "myDBPDO".*

## PHP MySQL Create Table

The CREATE TABLE statement is used to create a table in MySQL.

We will create a table named "MyGuests", with five columns: "id", "firstname", "lastname", "email" and "reg_date":

```
CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
)
```

**Notes on the table above:**

After the data type, you can specify other optional attributes for each column:

NOT NULL - Each row must contain a value for that column, null values are not allowed

DEFAULT value - Set a default value that is added when no other value is passed

UNSIGNED - Used for number types, limits the stored data to positive numbers and zero

AUTO INCREMENT - MySQL automatically increases the value of the field by 1 each time a new record is added

PRIMARY KEY - Used to uniquely identify the rows in a table. The column with PRIMARY KEY setting is often an ID number, and is often used with AUTO_INCREMENT

Each table should have a primary key column (in this case: the "id" column). Its value must be unique for each record in the table.

The following examples shows how to create the table in PHP **MySQLi**:

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)";

if ($conn->query($sql) === TRUE) {
  echo "Table MyGuests created successfully";
} else {
  echo "Error creating table: " . $conn->error;
}

$conn->close();
?>
```

## PHP MySQL Insert Data

After a database and a table have been created, we can start adding data in them.

Here are some syntax rules to follow:

- The SQL query must be quoted in PHP
- String values inside the SQL query must be quoted
- Numeric values must not be quoted
- The word NULL must not be quoted

The INSERT INTO statement is used to add new records to a MySQL table:

INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)

In the previous chapter we created an empty table named "MyGuests" with five columns: "id", "firstname", "lastname", "email" and "reg_date". Now, let us fill the table with data.

The following examples add a new record to the "MyGuests" table:

**Example (MySQLi Object-oriented)**

```php
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

## PHP MySQL Get Last Inserted ID

INSERT or UPDATE operation on a table with an AUTO_INCREMENT field, we can get the ID of the last inserted/updated record immediately.

**Example (MySQLi Object-oriented)**

```
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
  $last_id = $conn->insert_id;
  echo "New record created successfully. Last inserted ID is: " .
$last_id;
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

## PHP MySQL Insert Multiple Records

The following examples add three new records to the "MyGuests" table:

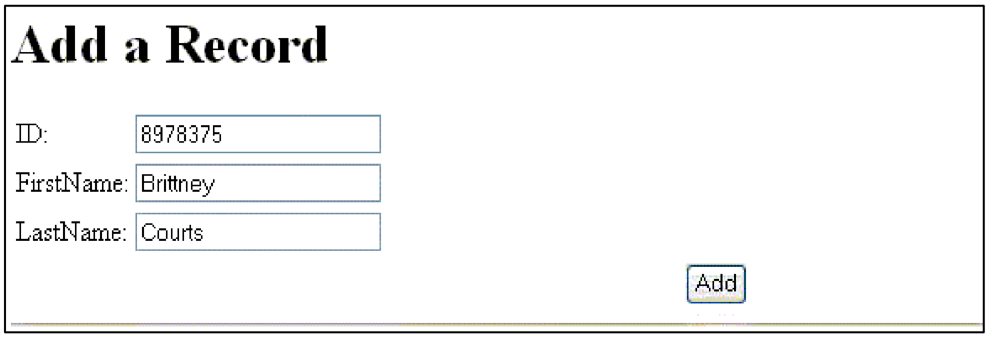
**Example (MySQLi Object-oriented)**

```php
<?php
$servername = "localhost";
$username = "root";//DB default username is root
$password = "";//DB default password in empty/null
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";

if ($conn->multi_query($sql) === TRUE) {
  echo "New records created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

## Exercises

1. Write a program to create a table named jobs including columns job_id, job_title, min_salary, max_salary and check whether the max_salary amount exceeding the upper limit 25000.

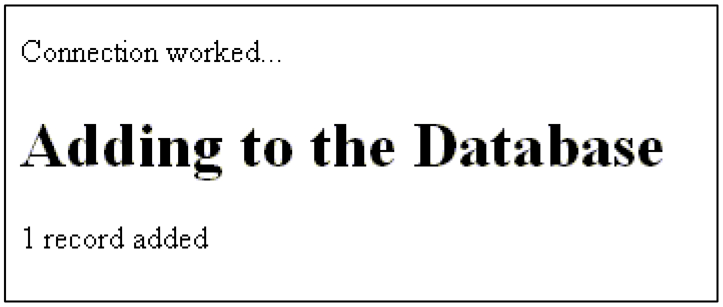
2. Create a form that looks similar to the following (values have been typed in for later demonstration)



Add a Record

| | |
|---|---|
| ID: | 8978375 |
| FirstName: | Brittney |
| LastName: | Courts |

When you click on the "Add" button, below window will show.



```
Connection worked...

Adding to the Database

1 record added
```

## References

- MySQL - www.mysql.com
- W3Schools Online Web Tutorials- www.w3schools.com
- PHP Manual - www.php.net