



Tishk International University
Faculty of Applied Science
Information Technology Department

Database Connection & Table Creation in PHP

Lecture 8

Fall 2025

Course Code: IT349

Grade 3

Islam Abdulazeez

islam.abdulaziz@tiu.edu.iq

December 2, 2025



Web Programming

Outlines



- Introduction to Databases and SQL
- Overview of MySQL and phpMyAdmin
- Setting Up phpMyAdmin
- Connecting PHP to MySQL
- Executing SQL Queries in PHP
- Creating and Managing Tables in PHP and phpMyAdmin
- Homework

Learning Outcomes



- **At the end of today's session, you will be able to:**
 - ✓ Explain PHP MySQL interaction via phpMyAdmin.
 - ✓ Use PHP to connect and run SQL commands.
 - ✓ Analyze database errors.
 - ✓ Create tables with PHP and phpMyAdmin.

Introduction to Database



- The internet would not be what it is today without databases. In fact, PHP likely wouldn't be as popular or useful without its built-in support for various types of databases.
- A **database** is a collection of tables (made up of columns and rows) that stores information.



Introduction to Database



- Examples:

Gmail Stores user email accounts, inbox contents, contacts, and attachments.



Social Media...





Some Examples!

Introduction to Database



- Most databases are created, updated, and read using **SQL** (Structured Query Language).
- **SQL** is a standard language for interacting with databases.
- Used for creating, reading, updating, and deleting data in databases (CRUD operations).
- We'll use MySQL, a popular open-source database management system



Common SQL Commands



- **CREATE**: Used to create a new database or table.



```
CREATE DATABASE school;
```



```
CREATE TABLE students (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(50),  
    age INT  
);
```


Common SQL Commands



- **ALTER**: Used to modify an existing table (add, remove, or change columns).

Add

```
ALTER TABLE students ADD COLUMN email VARCHAR(100);
```

Remove

```
ALTER TABLE students DROP COLUMN email;
```

Change

```
ALTER TABLE students RENAME COLUMN age TO student_age;
```

Common SQL Commands



- **INSERT**: Used to add new data (records) into a table.

```
INSERT INTO students (id, name, age)  
VALUES (1, 'Ali', 20);
```

- **SELECT**: Used to retrieve data from a table.

```
SELECT * FROM students;
```

```
SELECT * FROM students  
WHERE id = 5;
```

```
SELECT name, age FROM students  
WHERE id = 5;
```

Common SQL Commands



- **UPDATE**: Used to modify existing records in a table.

```
UPDATE students  
SET age = 21  
WHERE id = 1;
```

- **DELETE**: Used to delete specific records (rows) from a table..

```
DELETE FROM students  
WHERE id = 1;
```

Common SQL Commands



- **DROP**: Used to delete a whole database or table (REMOVES structure and data).



DROP TABLE students;



DROP DATABASE university;

Graphical Database Management Tools



- Alternatively, graphical database management tools such as **phpMyAdmin** or **MySQL Workbench** may be utilized to facilitate efficient database administration through an intuitive user interface.
- In this course, **phpMyAdmin** will be employed as the primary tool for managing and interacting with the database.

What is phpMyAdmin?



- **phpMyAdmin** is an open-source software tool, introduced on September 9, 1998, and written in PHP. It is a third-party application used to manage tables and data within databases.
- **phpMyAdmin** supports a wide range of operations on MySQL and MariaDB databases through a graphical interface. Its primary purpose is to facilitate the administration of MySQL databases over the web.



What is phpMyAdmin?



- **Key Features:**
 - Create, modify, and delete databases, tables, and records.
 - Execute SQL queries directly.
 - Import and export data in various formats.
 - Manage user accounts and permissions.



Why Use phpMyAdmin?

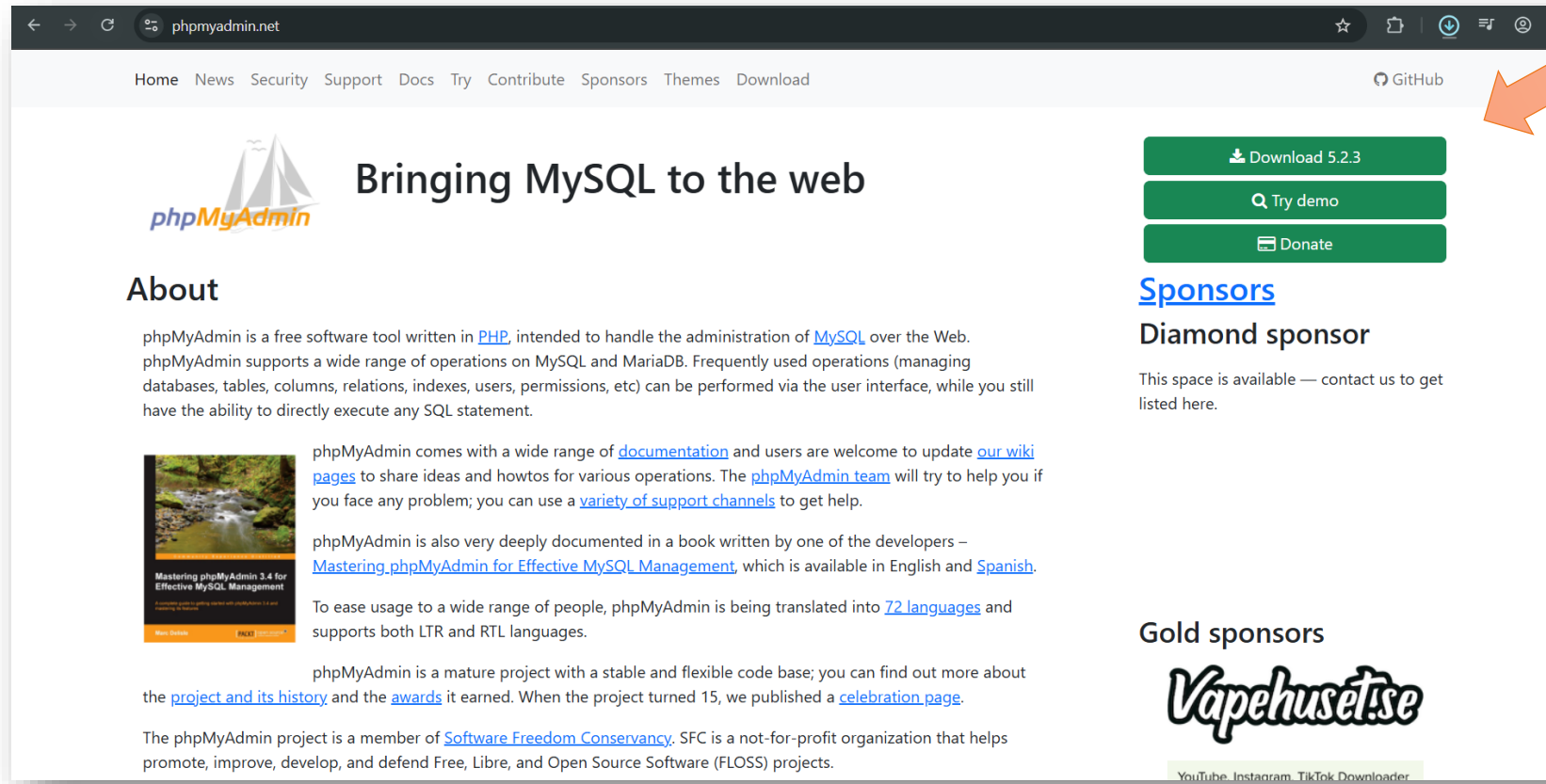


- Provides a user-friendly alternative to command-line database management.
- Supports complex database operations with minimal effort.
- Facilitates database backup and restore.
- Ideal for learning and teaching MySQL/MariaDB.



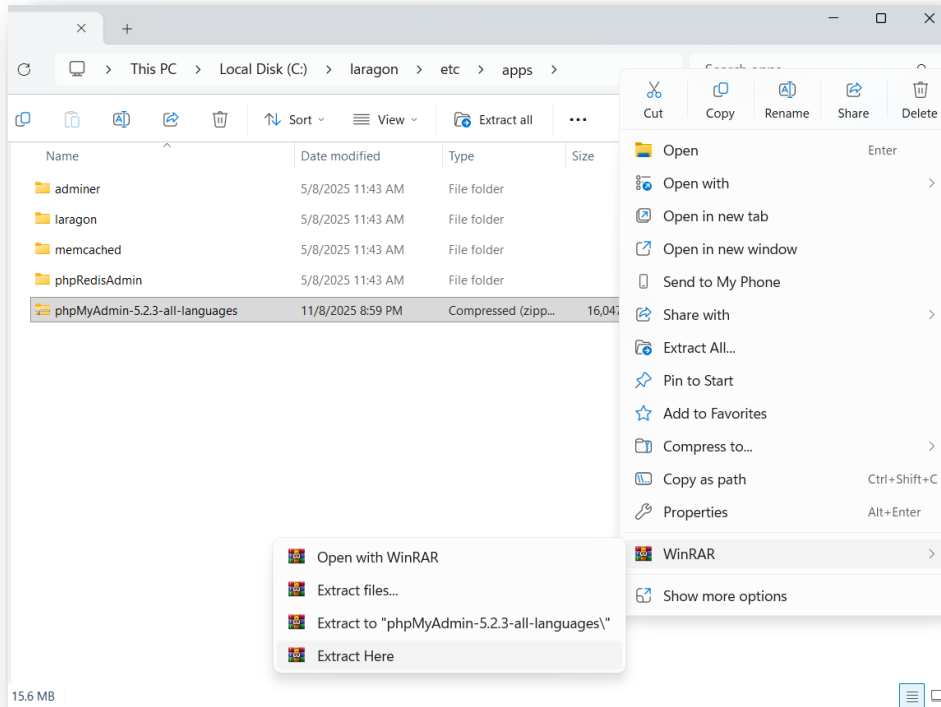
Installing phpMyAdmin

1. Download phpMyAdmin via -> <https://www.phpmyadmin.net>



Installing phpMyAdmin

2. Copy the downloaded .zip or .rar file to `C:\laragon\etc\apps`
3. Extract the file and rename it as phpMyAdmin.
4. Delete the .zip or .rar file.



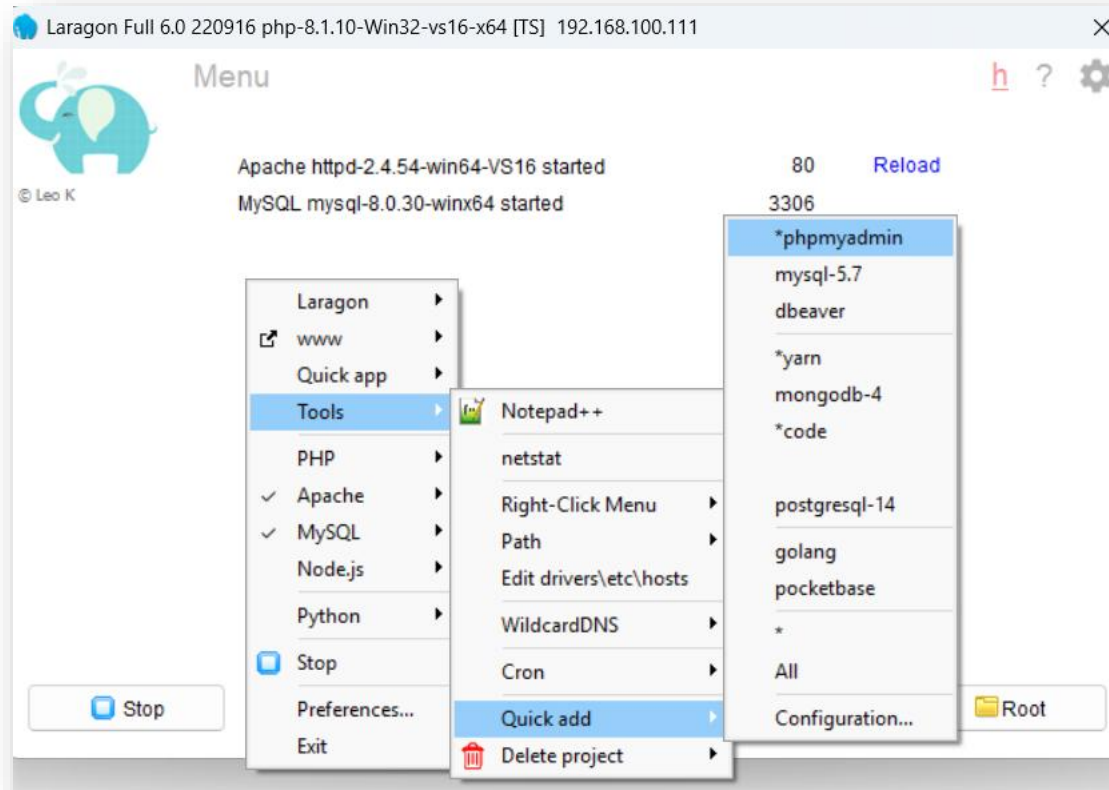
Name	Date modified	Type	Size
adminer	5/8/2025 11:43 AM	File folder	
laragon	5/8/2025 11:43 AM	File folder	
memcached	5/8/2025 11:43 AM	File folder	
phpMyAdmin	10/7/2025 11:40 PM	File folder	
phpRedisAdmin	5/8/2025 11:43 AM	File folder	

Installing phpMyAdmin



- Or you can download and install phpMyAdmin directly from Laragon.

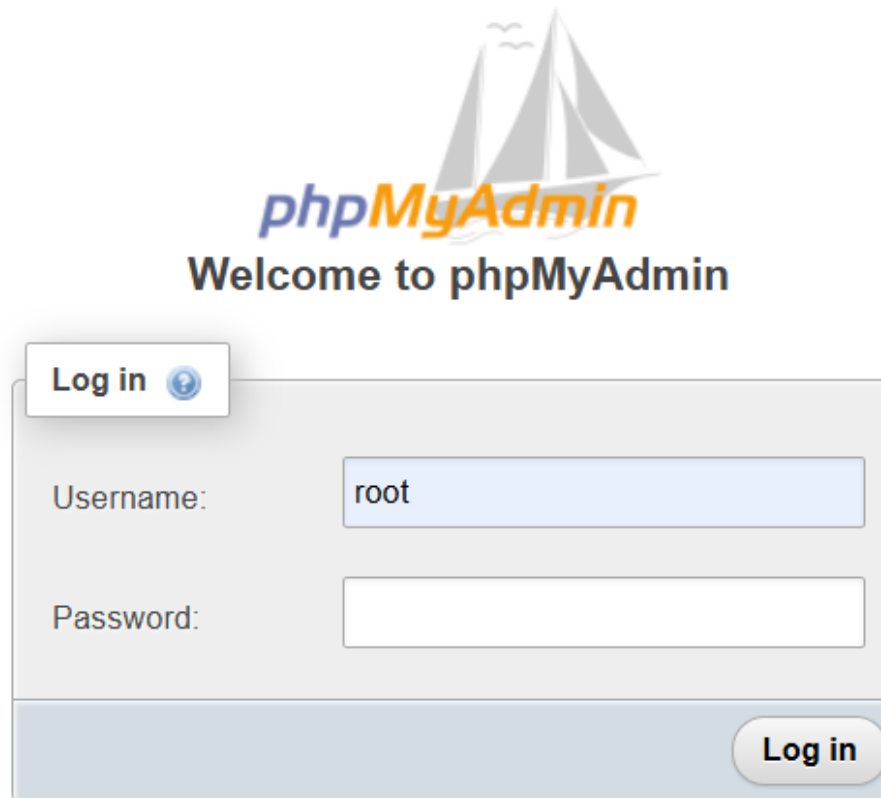
Steps: Right click on Laragon -> Tools -> Quick add -> *phpMyAdmin



phpMyAdmin Interface



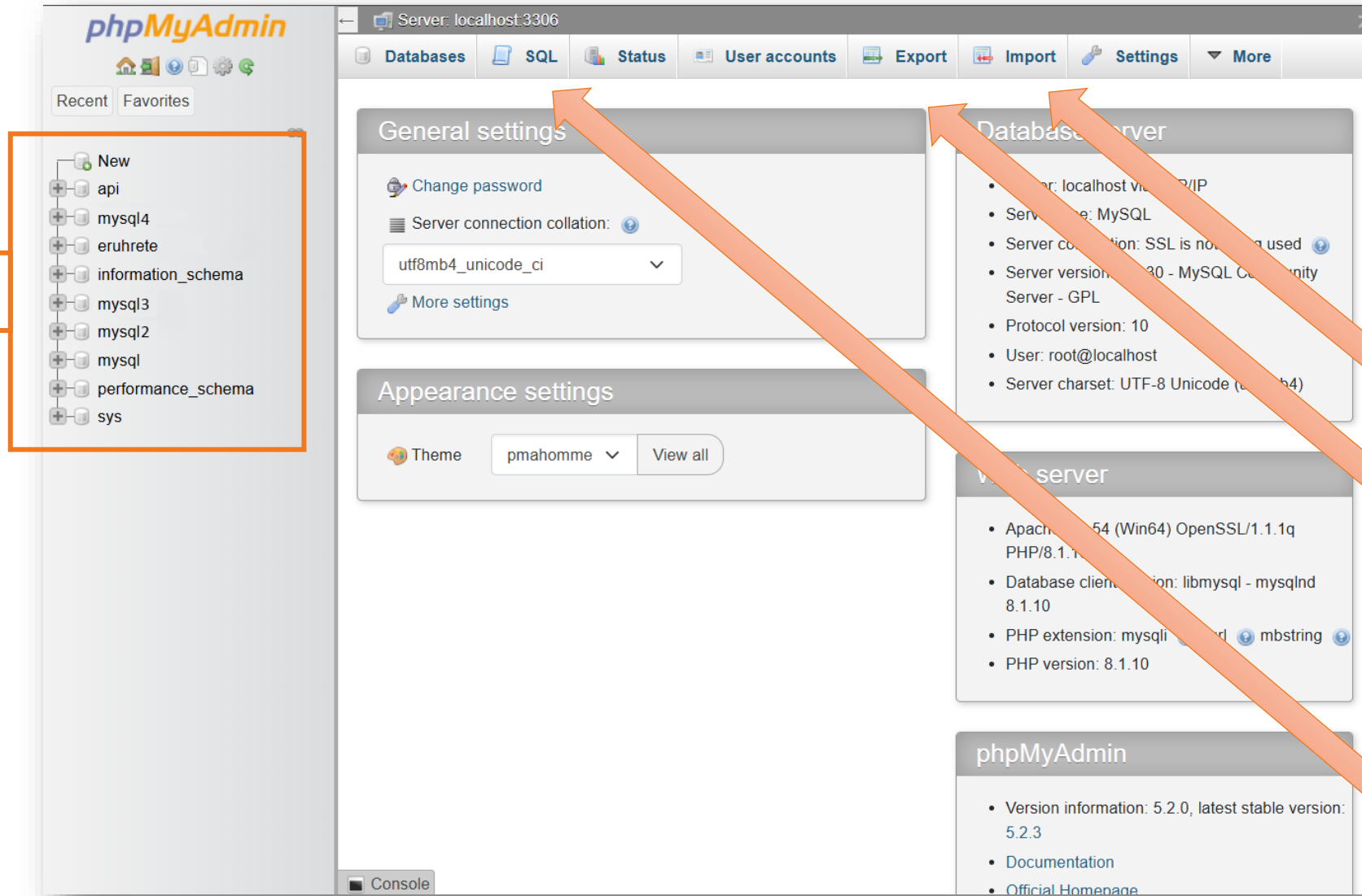
- By default, the username is “root” and has no password.

The image shows the phpMyAdmin login interface. At the top, there is a logo of a sailboat with the text 'phpMyAdmin' and 'Welcome to phpMyAdmin' below it. Below the welcome message is a login form. The form has a 'Log in' button with a help icon. Below this, there are two input fields: 'Username:' with the value 'root' and 'Password:' which is empty. At the bottom right of the form is a 'Log in' button. An orange arrow points from the right towards the 'Log in' button.

phpMyAdmin Interface



List of DBs

A screenshot of the phpMyAdmin web interface. The interface is divided into several sections. On the left, there is a sidebar with a 'List of DBs' section containing a tree view of databases: 'New', 'api', 'mysql4', 'eruhrete', 'information_schema', 'mysql3', 'mysql2', 'mysql', 'performance_schema', and 'sys'. The main area is titled 'Server: localhost:3306' and contains several tabs: 'Databases', 'SQL', 'Status', 'User accounts', 'Export', 'Import', 'Settings', and 'More'. The 'General settings' section is active, showing 'Change password', 'Server connection collation' (set to 'utf8mb4_unicode_ci'), and 'More settings'. Below this is the 'Appearance settings' section with a 'Theme' dropdown set to 'pmahomme' and a 'View all' button. On the right, there are sections for 'Database server' and 'Web server' details. The 'Database server' section lists: 'Server: localhost via TCP/IP', 'Server type: MySQL', 'Server connection: SSL is not being used', 'Server version: 5.6.23 - MySQL Community Server - GPL', 'Protocol version: 10', 'User: root@localhost', and 'Server charset: UTF-8 Unicode (utf8mb4)'. The 'Web server' section lists: 'Apache/2.4.18 (Ubuntu) OpenSSL/1.0.1.1q', 'PHP/5.6.12', 'Database client library: libmysql - mysqlnd 5.0.9', 'PHP extension: mysqli', 'mbstring', and 'PHP version: 5.6.12'. At the bottom, there is a 'phpMyAdmin' section with 'Version information: 5.2.0, latest stable version: 5.2.3', 'Documentation', and 'Official Homepage'. A 'Console' tab is visible at the bottom left.

Import DB

Export DB

Write SQL Query



Demonstration

- To establish a connection between PHP and the MySQL database server, we use the `mysqli_connect()` function. This function takes the `server's name`, `username`, and `password` as arguments, and attempts to create a connection to the database.
- If the connection is successful, PHP will be able to interact with the MySQL server; otherwise, it will return an error.

```
<?php
    $dbc = mysqli_connect('localhost', 'root', ' ');
?>
```

- After establishing a connection to the database server, we check whether the connection was successful. If `mysqli_connect_error()` (or `$conn->connect_error` in object-oriented style) detects an error, the script stops and displays an error message using `die()`. Otherwise, it prints a message confirming that the connection was successful.

```
<?php
    $dbc = mysqli_connect('localhost', 'root', '');
    if (!$dbc) {
        die("Connection failed: " . mysqli_connect_error());
    }
    else{
        echo "Connected to database server successfully";
    }
?>
```


Database Connection

- At this stage, the database name is specified within the `mysqli_connect()` function to establish a connection with the database in phpMyAdmin

```
<?php
    $dbc = mysqli_connect('localhost', 'root', '', 'my_database');
    if (!$dbc) {
        die("Connection failed: " . mysqli_connect_error());
    }
    else{
        echo "Connected to database successfully";
    }
?>
```

Close Connection

- Once you're finished working with a database, you can close the connection, just as you'd close an open file.

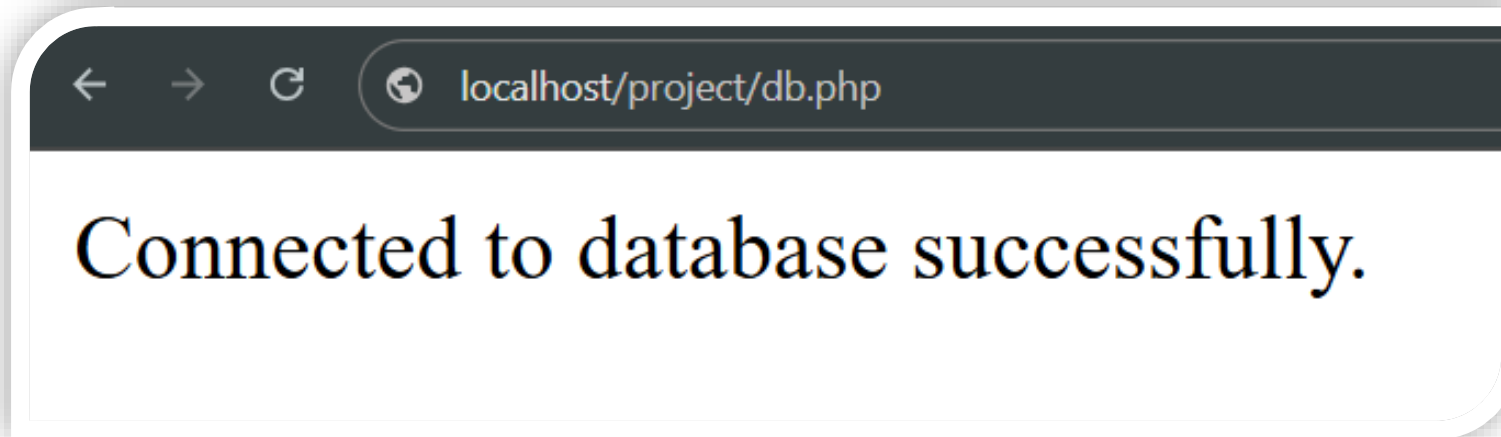
```
<?php mysqli_close($dbc); ?>
```

```
<?php
| $dbc = mysqli_connect('localhost', 'root', '', 'my_database');
| if (!$dbc) {
|     die("Connection failed: " . mysqli_connect_error());
| }
| echo "Connected to database successfully";
| mysqli_close($dbc);
?>
```

Database Connection Output



- If PHP is properly configured with MySQL support and the provided username, password, host, and database credentials are valid, the expected confirmation message will be displayed.



Debug SQL Errors



- **mysqli_error()**: It's used to retrieve the most recent MySQL error message that occurred while running a query on a given MySQL connection.

```
<?php  
    echo mysqli_error($dbc);  
?>
```

Create Table



- After creating and selecting the initial database, the next step is to define the tables within it. Although a database may contain multiple tables, this introductory example focuses on creating a single table to store all relevant data.

1

```
$query = "CREATE TABLE students (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    student_name VARCHAR(100),  
    email VARCHAR(100),  
    department VARCHAR(50),  
    age INT  
);"
```

Executing The Query



- **mysqli_query()**: It's used to send an SQL query to a MySQL database and execute it.

```
<?php  
    mysqli_query($dbc, $query);  
?>
```

Executing The Query

2

```
<?php
$dbc = mysqli_connect('localhost', 'root', '', 'uni');

if (!$dbc) {
    die("Connection failed: " . mysqli_connect_error());
} else {

    $query = "CREATE TABLE students (
        id INT AUTO_INCREMENT PRIMARY KEY,
        student_name VARCHAR(100),
        email VARCHAR(100),
        department VARCHAR(50),
        age INT
    )";

    if (mysqli_query($dbc, $query)) {
        echo "The students table was created successfully.";
    } else {
        echo "Error creating table: " . mysqli_error($dbc) . "<br> The query being executed was: $query";
    }
}
mysqli_close($dbc);
?>
```

Table Created Output



- If the table is created successfully without any errors, the following message will be displayed.

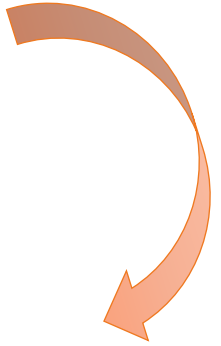




Let's do it together



Creating Table in



Homework



- Create a database in phpMyAdmin and name it “departments”. Then, design a form with four fields: one for entering a table name and three for entering department names. Upon submitting the form, a table should be automatically created in the database using the provided table name and department information.

Create Table and Add Departments

Table Name:

Department 1:

Department 2:

Department 3:

← → ↻ 🌐 localhost/project/create_table.php

Table departments created successfully.

Lab Assessments and Next Session's Topic



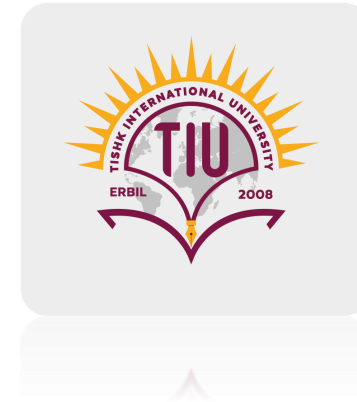
Lab Assessments

- Lab Exercises.

Next Session's Topic

- Create and Read in PHP

References



- Tatroe, K., & MacIntyre, P. (2020). Programming PHP: Creating dynamic web pages (4th ed.). O'Reilly Media.
- Ullman, L. (2016). PHP for the web: Visual QuickStart guide (5th ed.). Peachpit Press.
- PHP Documentation. (n.d.). PHP.net. Retrieved November 8, 2025, from <https://www.php.net/docs.php>



Thank You!