HiLCoE School of Computer Science & Technology

Chapter Two: PHP MySQL Database

Course Title : Web Technologies II

Instructor name: Yitayew Solomon
E-mail address: yitayewsolomon3@gmail.com

PHP MySQL Database

PHP MySQL Database: Introduction and Explanation

PHP provides robust support for interacting with MySQL databases, allowing developers to perform database operations such as connecting to a database, running SQL queries, and handling the results using PHP scripts. MySQL is a popular open-source relational database management system (RDBMS) commonly used in conjunction with PHP to build dynamic, data-driven web applications.

There are different ways to connect PHP with MySQL:

- **MySQLi (MySQL Improved)**: A relational database driver for MySQL, which offers both procedural and object-oriented ways to interact with the database.
- **PDO (PHP Data Objects)**: A database access layer that supports multiple database types, including MySQL, through a uniform API.

phpMyAdmin

phpMyAdmin: Explanation

phpMyAdmin is a free, open-source web-based tool used to manage MySQL and MariaDB databases. It provides an easy-to-use graphical interface that allows users to interact with their databases without needing to write SQL queries manually. phpMyAdmin simplifies database management tasks such as creating databases, tables, running queries, importing/exporting data, and managing user permissions.

Cont. ...

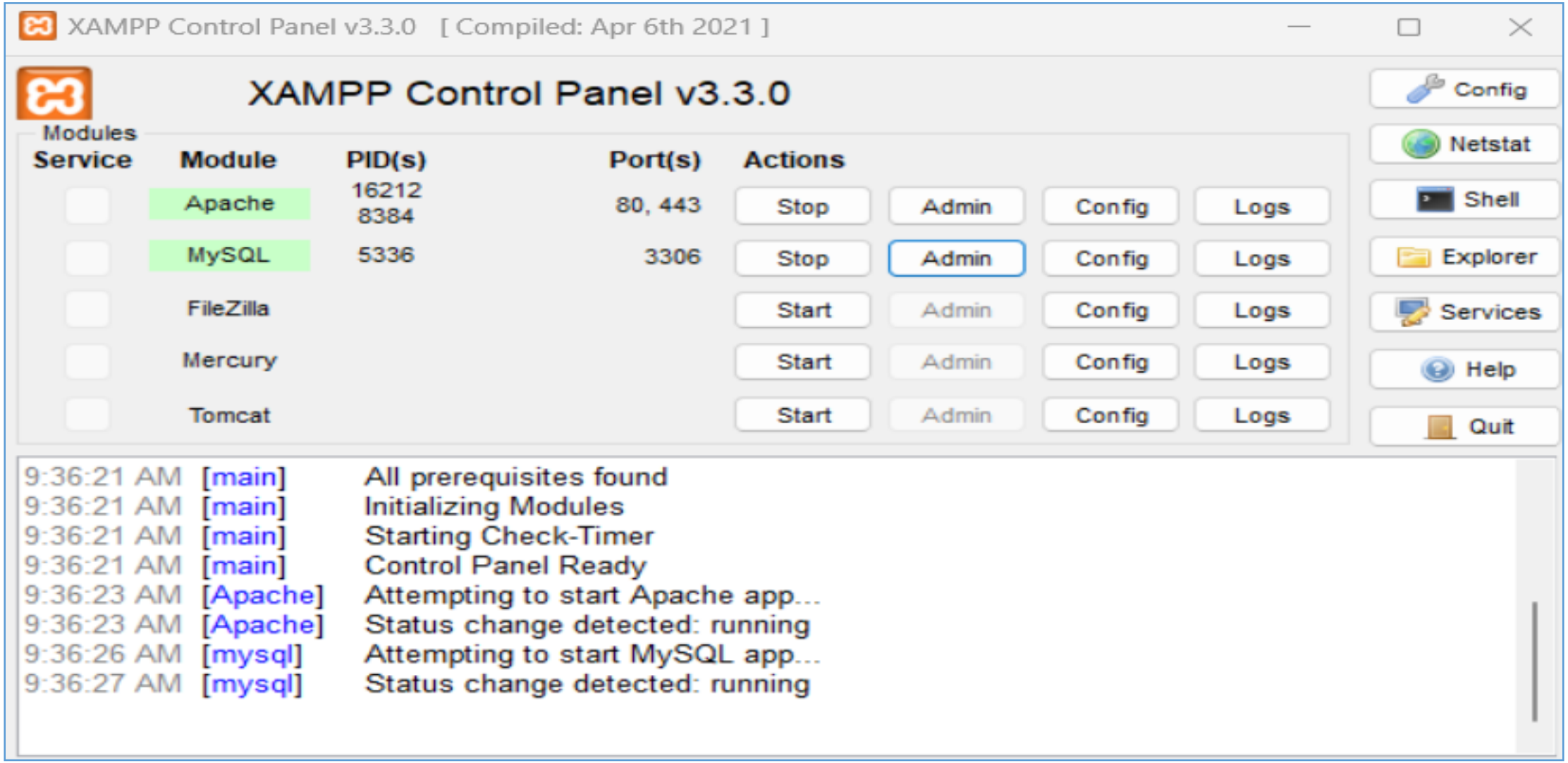
1. Features of phpMyAdmin

- **Database Management**: phpMyAdmin allows you to create, modify, and delete databases, tables, columns, indexes, and constraints.
- **SQL Execution**: You can run SQL queries directly through the interface, allowing for complex operations and custom queries.
- **Data Import/Export**: It supports importing data from CSV, SQL, and other formats and exporting data in various formats such as CSV, SQL, Excel, etc.
- **User Management**: phpMyAdmin allows the creation of new database users, assignment of privileges, and password management.
- **Backup and Restore**: Databases and tables can be backed up and restored via the interface, simplifying data recovery processes.

Cont. ...

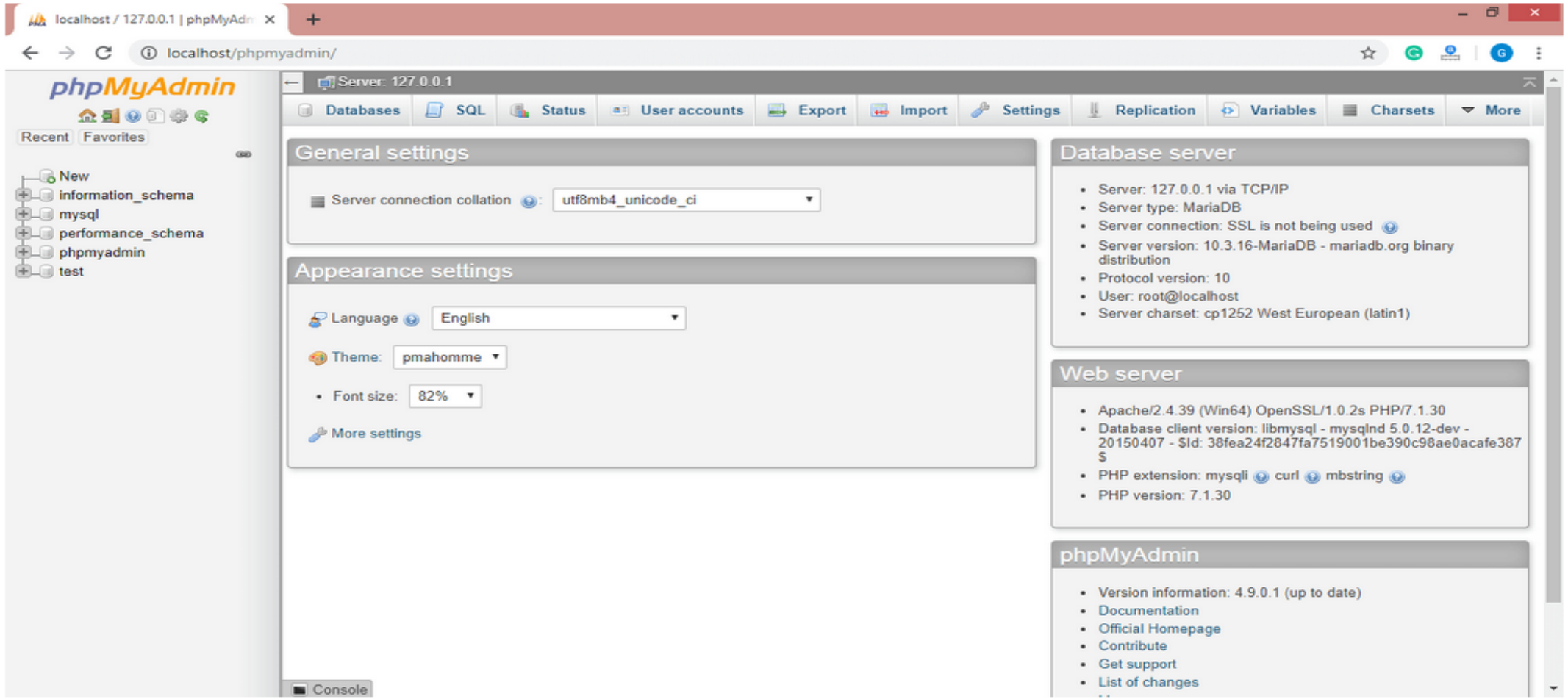
- **User Management**: phpMyAdmin allows the creation of new database users, assignment of privileges, and password management.
- **Backup and Restore**: Databases and tables can be backed up and restored via the interface, simplifying data recovery processes.
- **Database Optimization**: You can perform maintenance tasks such as optimizing tables, repairing corrupted data, and checking for errors.
- **Security**: phpMyAdmin supports SSL and HTTPS for secure access, and you can configure user authentication for better security.

How to open phpMyadmin



Cont. ...

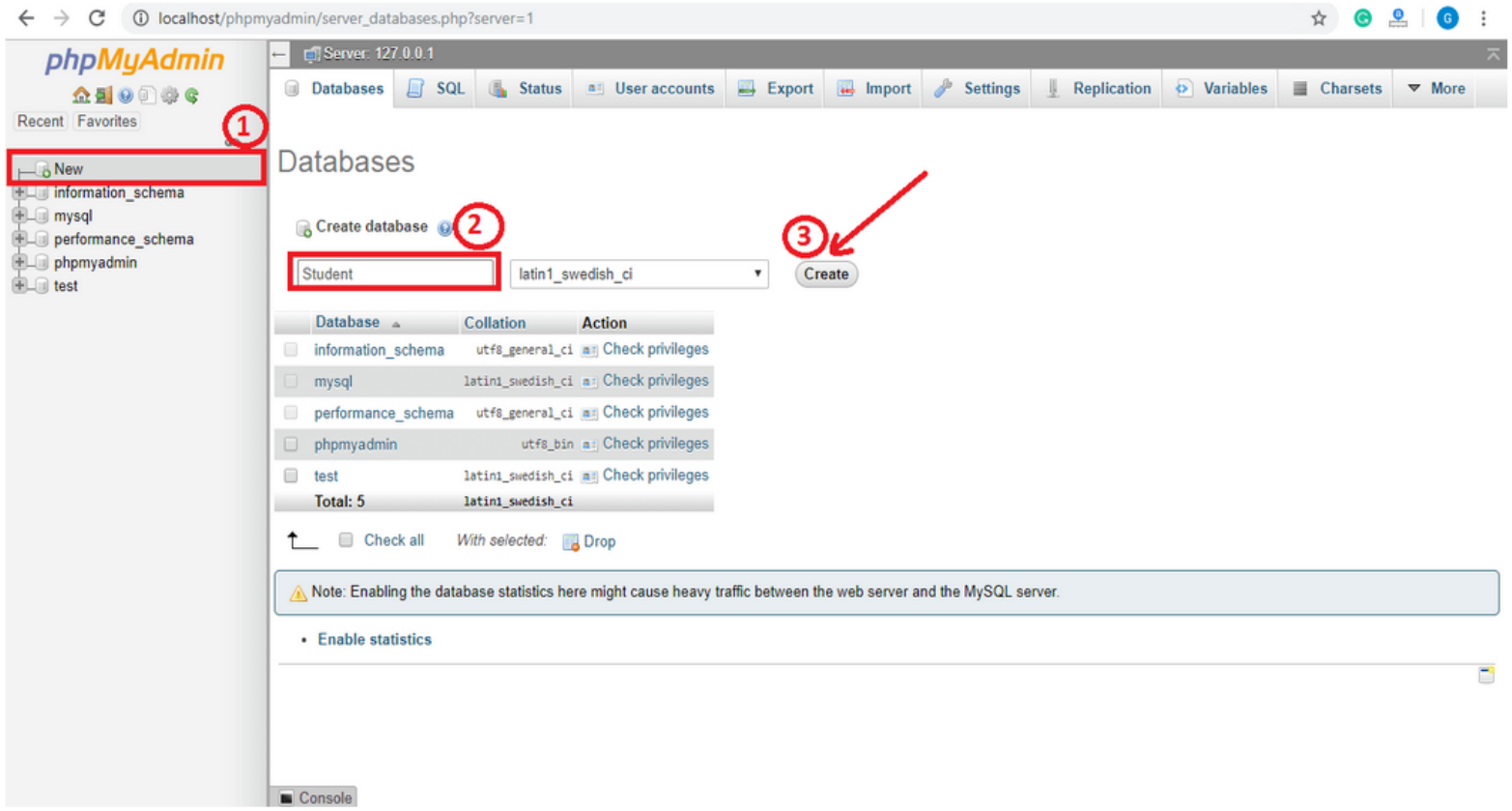
Now open the browser and type http://localhost/phpmyadmin/. phpMyAdmin will start running in the browser.



Cont. ...

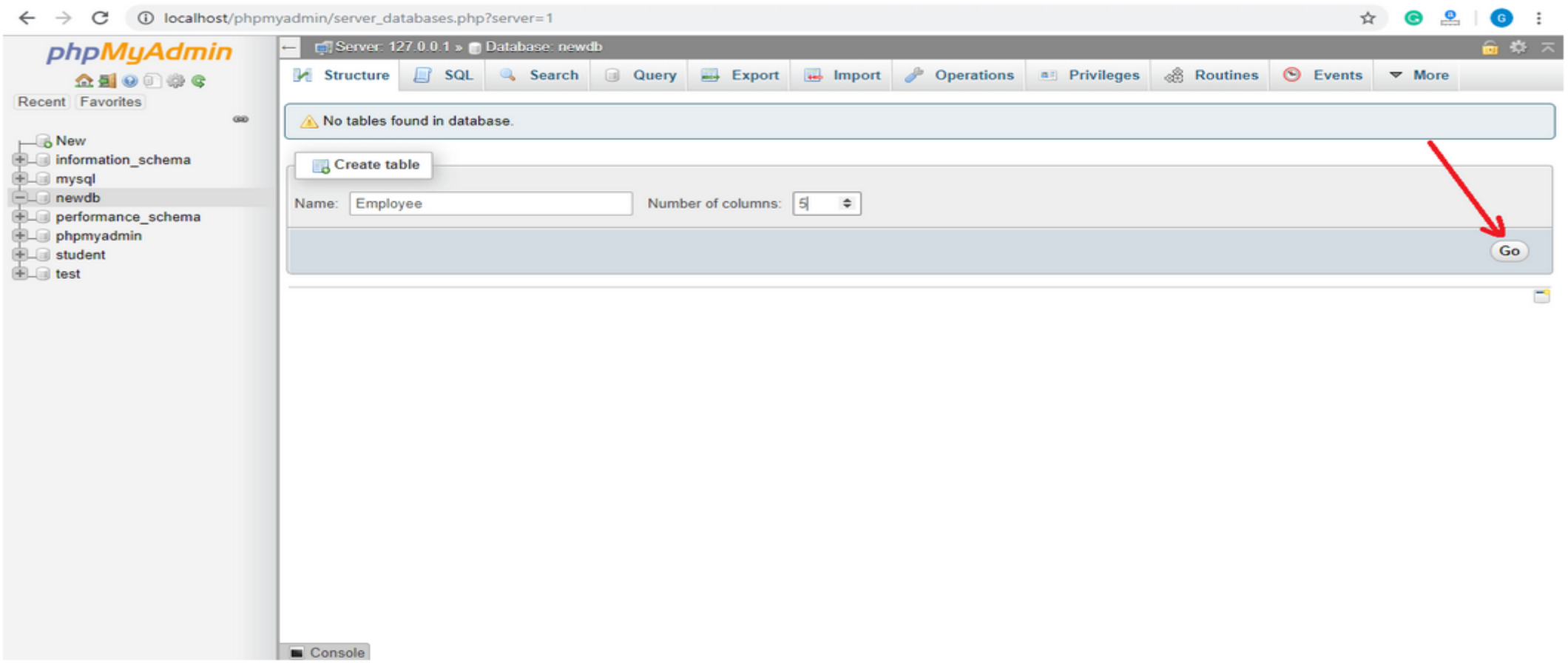
How to work with phpMyAdmin?

Click on **New (1)** to create a database and enter the database name in **Create database (2)** field and then click on **Create (3)** button. We can create any number of databases.



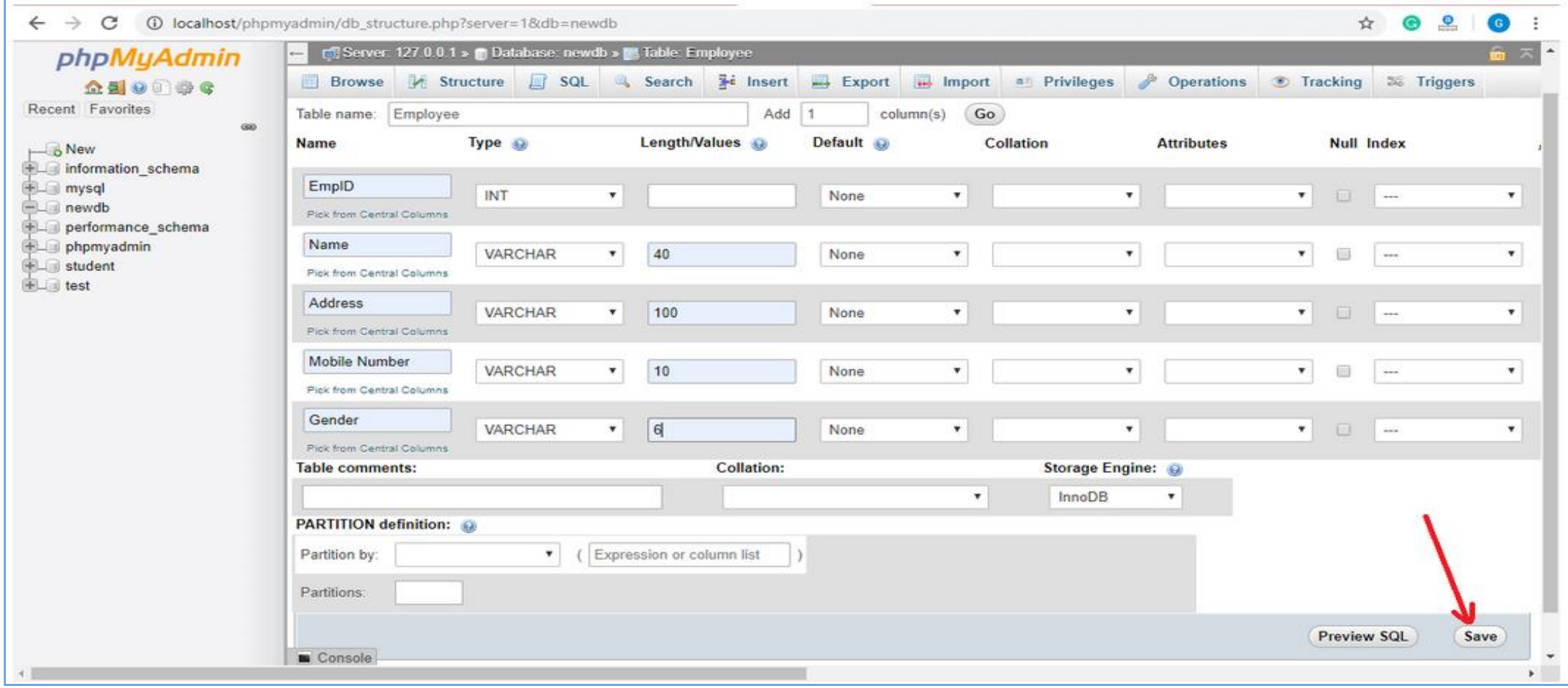
Cont. ...

Enter the table name, number of columns, and click on Go. A message will show that the table is created successfully.



Cont. ...

Now enter the field name, type, their size, and any constraint here and save it.



Cont. ...

The table is created successfully. We can make changes in the table from here.

localhost/phpmyadmin/tbl_structure.php?server=1&db=newdb&table=employee

phpMyAdmin

Recent Favorites

- New
- information_schema
- mysql
- newdb
  - New
  - employee
    - Columns
    - Indexes
- performance_schema
- phpmyadmin
- student
- test

Server: 127.0.0.1 » Database: newdb » Table: employee

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	EmpID	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	Name	varchar(40)	latin1_swedish_ci		No	None			Change Drop More
3	Address	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
4	Mobile Number	varchar(10)	latin1_swedish_ci		No	None			Change Drop More
5	Gender	varchar(6)	latin1_swedish_ci		No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns Remove from central columns

Print Propose table structure Track table Move columns Normalize

Add 1 column(s) after Gender Go

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	EmpID	0	A	No	

Create an index on 1 columns Go

Partitions

No partitioning defined!

Console

Creating User

phpMyAdmin

Recent Favorites

- New
- college
  - New
  - student
- information_schema
- mysql
- performance_schema
- phpmyadmin
- student
- test

Server: 127.0.0.1

Databases | SQL | Status | User accounts | Export | Import | Settings | Replication | Variables | Charsets

Add user account

Login Information

User name: Use text field — yitayew

An account already exists with the same username but possibly a different hostname.

Host name: Any host — %

Password: Use text field — •••••••• Strength:

Re-type:

Authentication plugin: Native MySQL authentication

Generate password: Generate

Connecting PHP to MySQL Using MySQLi

C:\xampp\htdocs\web\Database\Database_Connection_Mysqli.php • - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Database_Connection_Mysqli.php

```php
<?php
// Database credentials
$servername = "localhost";
$username = "yitayew";
$password = "test@123";
$dbname = "student";

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

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

// Close connection
mysqli_close($conn);
?>
```

Connected successfully

PHP: Create a MySQL Database

PHP: Create a MySQL Database

To create a MySQL database using PHP, you typically need to use the following steps:

1. **Establish a connection** to the MySQL server using PHP.
2. **Write a SQL query** to create a new database.
3. **Execute the query** using PHP.
4. **Close the connection** after the query execution.

Cont. ...

C:\xampp\htdocs\web\Database\Create_Database.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Create_Database.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";  // Change if necessary
$username = "yitayew";      // MySQL username (root by default)
$password = "test@123";     // MySQL password (blank by default)

// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// SQL query to create a new database
$sql = "CREATE DATABASE myNewDatabase";

// Execute query and check if database was created successfully
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully!";
} else {
    echo "Error creating database: " . $conn->error;
}
// Close the connection
$conn->close();
?>
```

Database created successfully!

Cont. ...

2. Explanation of the Code

- **Connection Details:**
  - `$servername`: This is the name of the server where MySQL is hosted. Usually, it's `localhost` if running on your own machine.
  - `$username`: The MySQL user that has privileges to create databases. Often this is `root` for local development.
  - `$password`: The password for the MySQL user (it may be empty for local environments).
- **Creating a Connection:**
  - The `new mysqli()` function is used to connect to the MySQL server.
  - If the connection fails, the script terminates with the `die()` function, printing the error message.

Cont. ...

- **SQL Query to Create Database:**
  - The `CREATE DATABASE` statement is used to define the SQL query.
  - The name `myNewDatabase` can be replaced with any name you want for your database.
- **Executing the SQL Query:**
  - The `query()` method is used to send the SQL statement to MySQL.
  - If the query is successful, a success message is printed. Otherwise, an error message is shown.
- **Closing the Connection:**
  - It's important to close the MySQL connection once the task is complete to free up resources.

Cont. ...

3. Points to Remember

- **Privileges:** The MySQL user must have the necessary privileges to create a database. If not, the script will fail.
- **Database Name Rules:** The name of the database should be unique and follow MySQL's naming conventions (e.g., no spaces, should start with a letter, etc.).
- **Error Handling:** For better error handling, you can implement more robust error checking, such as logging errors into a file instead of displaying them to the user.

Cont. ...

4. Testing the Script

1. Save the above PHP code to a file, e.g., `create_database.php`.
2. Run the script by navigating to the file in your web browser (e.g., `http://localhost/create_database.php`).
3. If the connection to the MySQL server is successful and the user has the necessary permissions, a new database called `myNewDatabase` will be created.
4. You can verify this by checking your MySQL server (either through phpMyAdmin or directly using the MySQL command line).

PHP: Create a MySQL Table Using MySQLi

PHP: Create a MySQL Table Using MySQLi

To create a MySQL table using PHP and the MySQLi extension, you need to:

1. **Connect to the database** using MySQLi.
2. **Write an SQL query** that defines the table's structure (columns, data types, etc.).
3. **Execute the query** to create the table.
4. **Close the connection** once the task is done.

Cont. ...

C:\xampp\htdocs\web\Database\Create_Table.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Create_Table.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";  // Server name (usually 'localhost')
$username = "yitayew";         // MySQL username (typically 'root' for local environments)
$password = "test@123";             // MySQL password (blank if not set)
$dbname = "myNewDatabase";  // Name of the database where the table will be created

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

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

```

Cont. ...

C:\xampp\htdocs\web\Database\Create_Table.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Create_Table.php

```php
// SQL query to create a new 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
)";

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

// Close the connection
$conn->close();
?>
```

Table MyGuests created successfully!

Cont. ...

2. Explanation of the Code

- Connection Details:
  - `$servername`, `$username`, `$password`: These hold the credentials for connecting to the MySQL server.
  - `$dbname`: The name of the database where the table will be created.
- Creating a Connection:
  - The `new mysqli()` function is used to establish a connection to the MySQL server. If the connection fails, the script stops, and an error message is displayed using `die()`.

Cont. ...

- **SQL Query to Create the Table:**
  - The `CREATE TABLE` statement defines the structure of the table:
    - `id`: An auto-incremented, unsigned integer column that serves as the primary key.
    - `firstname`: A `VARCHAR` column to store the guest's first name, with a maximum length of 30 characters.
    - `lastname`: A `VARCHAR` column to store the guest's last name, also with a maximum length of 30 characters.
    - `email`: A `VARCHAR` column to store the guest's email, with a maximum length of 50 characters.
    - `reg_date`: A `TIMESTAMP` column that automatically stores the current timestamp when a new record is added or updated.

Cont. ...

- **Executing the Query:**
  - The `query()` method is used to run the SQL command.
  - If the table is created successfully, a success message is printed; otherwise, an error message is displayed.
- **Closing the Connection:**
  - It's important to close the MySQL connection with `$conn->close()` after the task is completed to release resources.

Cont. ...

3. Points to Remember

- **Auto-incremented Primary Key:** In this example, the `id` column is auto-incremented and serves as the primary key. This means that every time a new row is added, the `id` value will automatically increase by 1.
- **Data Types:** The data types for the columns must be chosen based on the kind of data they will store. For instance, `VARCHAR(30)` for short strings and `TIMESTAMP` for date and time.
- **Error Handling:** You can implement more advanced error handling by logging errors or showing more user-friendly messages in a production environment.

Cont. ...

4. Testing the Script

1. Save the above PHP code to a file, e.g., `create_table.php`.
2. Ensure that the database `myNewDatabase` exists. You can create it using the earlier example.
3. Run the script by navigating to the file in your web browser (e.g., `http://localhost/create_table.php`).
4. If successful, a new table `MyGuests` will be created in the `myNewDatabase` database.

Cont. ...

5. Verifying the Table

You can verify that the table was created by either:

1. **Using phpMyAdmin:**
   - Navigate to phpMyAdmin, select your database, and check if the table `MyGuests` appears under the list of tables.
2. **Using MySQL CLI:**
   - Run the following commands:

sql Copy code

```sql
USE myNewDatabase;
SHOW TABLES;
DESCRIBE MyGuests;
```

PHP: Insert Data into MySQL Table Using MySQLi

PHP: Insert Data into MySQL Table Using MySQLi

To insert data into a MySQL database table using PHP and the MySQLi extension, the steps are:

1. **Connect to the MySQL database** using MySQLi.
2. **Write an SQL query** to insert the data into a specific table.
3. **Execute the query** to perform the insertion.
4. **Close the connection** after completing the operation.

Cont. ...

C:\xampp\htdocs\web\Database\Insert_Data.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Insert_Data.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";
$username = "yitayew";
$password = "test@123";
$dbname = "myNewDatabase"; // Ensure this database exists
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// SQL query to insert data into the MyGuests table
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('yitayew', 'Solomon', 'yitayewsolomon3@gmail.com')";
// Execute query and check if data was inserted successfully
if ($conn->query($sql) === TRUE) {
    echo "New record created successfully!";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
// Close the connection
$conn->close();
?>
```

←T→			id	firstname	lastname	email	reg_date
☐	Edit	Copy	Delete	1	yitayew	Solomon	yitayewsolomon3@gmail.com	2024-10-07 11:04:24

Cont. ...

2. Explanation of the Code

- **Connection Details:**
  - `$servername`, `$username`, `$password`: These hold the credentials for connecting to the MySQL server.
  - `$dbname`: The name of the database where the table is located.
- **Creating a Connection:**
  - The `new mysqli()` function is used to establish a connection to the MySQL server. If the connection fails, the script terminates with `die()` and displays an error message.
- **SQL Query to Insert Data:**
  - The `INSERT INTO` SQL statement specifies which table (`MyGuests`) to insert data into and the columns to populate: `firstname`, `lastname`, and `email`.
  - The `VALUES` clause specifies the actual data to insert for those columns.

Cont. ...

- **SQL Query to Insert Data:**
  - The `INSERT INTO` SQL statement specifies which table (`MyGuests`) to insert data into and the columns to populate: `firstname`, `lastname`, and `email`.
  - The `VALUES` clause specifies the actual data to insert for those columns.
- **Executing the Query:**
  - The `query()` method runs the SQL command.
  - If the data is inserted successfully, a success message is displayed. Otherwise, an error message is shown.
- **Closing the Connection:**
  - It's important to close the MySQL connection once the insertion is complete using `$conn->close()`.

Cont. ...

3. Points to Remember

- **Table Structure**: Ensure the table (`MyGuests`) already exists in the database, with the columns (`firstname`, `lastname`, `email`) defined as in previous examples.
- **Prepared Statements**: For security reasons, especially to avoid SQL injection, use prepared statements (discussed below) when inserting user-provided data into the database.
- **Auto-Incremented IDs**: If your table includes an `AUTO_INCREMENT` column (e.g., `id`), you do not need to specify a value for it during the insertion; MySQL will automatically generate a unique value.

Inserting Multiple Records

C:\xampp\htdocs\web\Database\Inserting_Multiple_Records.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Inserting_Multiple_Records.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "myNewDatabase"; // Ensure this database exists

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

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

// SQL queries to insert multiple records
$sql1 = "INSERT INTO MyGuests (firstname, lastname, email) VALUES ('Haileab', 'Solomon', 'haileabsolomon@gmail.com')";
$sql2 = "INSERT INTO MyGuests (firstname, lastname, email) VALUES ('Yared', 'Solomon', 'yaredsolomon@gmail.com')";

```

Cont. ...

C:\xampp\htdocs\web\Database\Inserting_Multiple_Records.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Inserting_Multiple_Records.php

```php
// Execute first query
if ($conn->query($sql1) === TRUE) {
    echo "First record inserted successfully!";
} else {
    echo "Error: " . $sql1 . "<br>" . $conn->error;
}

// Execute second query
if ($conn->query($sql2) === TRUE) {
    echo "Second record inserted successfully!";
} else {
    echo "Error: " . $sql2 . "<br>" . $conn->error;
}

// Close the connection
$conn->close();
?>
```

←T→			id	firstname	lastname	email	reg_date
Edit	Copy	Delete	1	yitayew	Solomon	yitayewsolomon3@gmail.com	2024-10-07 11:04:24
Edit	Copy	Delete	2	Haileab	Solomon	haileabsolomon@gmail.com	2024-10-07 11:12:31
Edit	Copy	Delete	3	Yared	Solomon	yaredsolomon@gmail.com	2024-10-07 11:12:31

Inserting User Input Data (Prepared Statements)

C:\xampp\htdocs\web\Database\Insert_Data_Prepared_Statement.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Inserting_Multiple_Records.php | Insert_Data_Prepared_Statement.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";
$username = "yitayew";
$password = "test@123";
$dbname = "myNewDatabase";

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

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

```

Cont. …

C:\xampp\htdocs\web\Database\Insert_Data_Prepared_Statement.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Insert_Data_Prepared_Statement.php

```php
// Prepare and bind
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);

// Set parameters and execute
$firstname = "Natanim";
$lastname = "Yitayew";
$email = "natanimyitayew@gmail.com";
$stmt->execute();

$firstname = "Dureti";
$lastname = "Guye";
$email = "duretiguye@gmail.com";
$stmt->execute();

echo "Records inserted successfully!";

// Close the statement and connection
$stmt->close();
$conn->close();
```

←T→	id	firstname	lastname	email	reg_date
Edit Copy Delete	1	yitayew	Solomon	yitayewsolomon3@gmail.com	2024-10-07 11:04:24
Edit Copy Delete	2	Haileab	Solomon	haileabsolomon@gmail.com	2024-10-07 11:12:31
Edit Copy Delete	3	Yared	Solomon	yaredsolomon@gmail.com	2024-10-07 11:12:31
Edit Copy Delete	4	Natanim	Yitayew	natanimyitayew@gmail.com	2024-10-07 11:20:57
Edit Copy Delete	5	Dureti	Guye	duretiguye@gmail.com	2024-10-07 11:20:57

Cont. ...

- `prepare()`: Prepares an SQL statement for execution.
- `bind_param()`: Binds the variables to the prepared statement (`sss` stands for string, string, string – the types of values being inserted).
- `execute()`: Executes the prepared statement with the bound variables.

---

6. Testing the Script

1. Save the PHP script as `insert_data.php`.
2. Ensure your table exists in the `myNewDatabase` database.
3. Run the script by navigating to `http://localhost/insert_data.php`.
4. Check if the data has been inserted successfully into the MySQL table (e.g., through phpMyAdmin or MySQL CLI).

Inserting Data into MySQL Using an HTML Form and PHP

Inserting Data into MySQL Using an HTML Form and PHP

To insert data into a MySQL database using an HTML form, you'll follow these steps:

1. **Create an HTML form** for users to input data.
2. **Capture form data using PHP.**
3. **Insert the form data into a MySQL table** using PHP and MySQLi.

Create an HTML Form

C:\xampp\htdocs\web\Database\form1.html - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

form1.html | insert_data_form.php

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Insert Data Using Form</title>
</head>
<body>
    <h2>Submit Your Information</h2>
    <form action="insert_data_form.php" method="post">
        <label for="firstname">First Name:</label><br>
        <input type="text" id="firstname" name="firstname" required><br><br>

        <label for="lastname">Last Name:</label><br>
        <input type="text" id="lastname" name="lastname" required><br><br>

        <label for="email">Email:</label><br>
        <input type="email" id="email" name="email" required><br><br>

        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

Cont. ...

- **Action**: The form's `action` attribute points to `insert_data.php`, where the data will be processed using PHP.
- **Method**: We are using the `POST` method to send form data to the server.

---

2. PHP Script to Handle Form Submission

Create a file named `insert_data.php` to process the form data and insert it into the database.

php Copy code

Cont. ...

C:\xampp\htdocs\web\Database\insert_data_form.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

form1.html | insert_data_form.php

```php
<?php
// MySQL server connection credentials
$servername = "localhost";
$username = "yitayew";
$password = "test@123";
$dbname = "myNewDatabase"; // Ensure this database exists

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

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

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Prepare and bind
    $stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
    $stmt->bind_param("sss", $firstname, $lastname, $email);

```

Cont. ...

C:\xampp\htdocs\web\Database\insert_data_form.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

form1.html | insert_data_form.php

```php
    // Retrieve form data
    $firstname = htmlspecialchars($_POST['firstname']);
    $lastname = htmlspecialchars($_POST['lastname']);
    $email = htmlspecialchars($_POST['email']);

    // Execute the prepared statement
    if ($stmt->execute()) {
        echo "New record created successfully!";
    } else {
        echo "Error: " . $stmt->error;
    }

    // Close the statement and connection
    $stmt->close();
}

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

Cont. ...

←T→			id	firstname	lastname	email	reg_date
Edit	Copy	Delete	1	yitayew	Solomon	yitayewsolomon3@gmail.com	2024-10-07 11:04:24
Edit	Copy	Delete	2	Haileab	Solomon	haileabsolomon@gmail.com	2024-10-07 11:12:31
Edit	Copy	Delete	3	Yared	Solomon	yaredsolomon@gmail.com	2024-10-07 11:12:31
Edit	Copy	Delete	4	Natanim	Yitayew	natanimyitayew@gmail.com	2024-10-07 11:20:57
Edit	Copy	Delete	5	Dureti	Guye	duretiguye@gmail.com	2024-10-07 11:20:57
Edit	Copy	Delete	6	yitayew	Solomon	yitayewsolomon3@gmail.com	2024-10-07 11:44:14
Edit	Copy	Delete	7	Mihiret	Gashaw	yitayewsolomon3@gmail.com	2024-10-07 11:50:57
Edit	Copy	Delete	8	Abebe	Kebede	abebe@gmail.com	2024-10-07 11:51:52

Cont. ...

3. Explanation of the Code

- HTML Form:
  - The form contains three fields: first name, last name, and email.
  - The `action` attribute specifies that the form will be submitted to `insert_data.php`.
  - The `method="post"` specifies that the form data will be sent via the POST method.
- PHP Script (`insert_data.php`):
  - **Form Data Retrieval**: PHP retrieves the submitted form data using the `$_POST` superglobal array.
  - **Data Sanitization**: We use `htmlspecialchars()` to prevent XSS attacks by converting special characters into HTML entities.

Cont. ...

- **PHP Script (`insert_data.php`):**
  - **Form Data Retrieval:** PHP retrieves the submitted form data using the `$_POST` superglobal array.
  - **Data Sanitization:** We use `htmlspecialchars()` to prevent XSS attacks by converting special characters into HTML entities.
  - **SQL Query:** The `INSERT INTO` statement is used to insert the form data into the `MyGuests` table.
  - **Execute the Query:** The PHP script attempts to execute the SQL query. If successful, a success message is displayed; otherwise, an error message is shown.

Cont. ...

4. Example Database Table Structure

The `MyGuests` table in MySQL should have the following structure for this example to work:

sql Copy code

```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
);
```

Cont. ...

5. Testing the Script

1. Save the **HTML form** as `form.html`.
2. Save the **PHP script** as `insert_data.php`.
3. Ensure that the database and the table `MyGuests` exist.
4. Open the `form.html` file in a browser (e.g., `http://localhost/form.html`).
5. Fill in the form with sample data and submit it.
6. If the form is submitted successfully, check the database (using phpMyAdmin or MySQL CLI) to see if the data has been inserted:

sql Copy code

```sql
SELECT * FROM MyGuests;
```

All in One(HTML + PHP)

C:\xampp\htdocs\web\Database\Insert_Data_Form_All.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Insert_Data_Form_All.php

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Insert Data Using Form</title>
</head>
<body>

    <!-- HTML FORM TO COLLECT DATA -->
    <h2>Submit Your Information</h2>
    <form action="" method="post">
        <label for="firstname">First Name:</label><br>
        <input type="text" id="firstname" name="firstname" required><br><br>

        <label for="lastname">Last Name:</label><br>
        <input type="text" id="lastname" name="lastname" required><br><br>

        <label for="email">Email:</label><br>
        <input type="email" id="email" name="email" required><br><br>

        <input type="submit" value="Submit">
    </form>

```

Cont. ...

C:\xampp\htdocs\web\Database\Insert_Data_Form_All.php • - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Insert_Data_Form_All.php

```php
<?php
// PHP Script to handle form submission and insert data into MySQL database

// MySQL server connection credentials
$servername = "localhost"; // Server name (change if necessary)
$username = "root";        // Username (default is 'root')
$password = "";            // Password (leave empty if not set)
$dbname = "myNewDatabase"; // Name of the database

// Create a connection to the MySQL database
$conn = new mysqli($servername, $username, $password, $dbname);
// Check if the connection was successful
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

Cont. ...

C:\xampp\htdocs\web\Database\Insert_Data_Form_All.php • - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

Insert_Data_Form_All.php

```php
    // Check if the form has been submitted
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        // Prepare the SQL statement with placeholders to avoid SQL injection
        $stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
        $stmt->bind_param("sss", $firstname, $lastname, $email);
        // Get the form data and sanitize it to prevent XSS attacks
        $firstname = htmlspecialchars($_POST['firstname']);
        $lastname = htmlspecialchars($_POST['lastname']);
        $email = htmlspecialchars($_POST['email']);
        // Execute the prepared statement
        if ($stmt->execute()) {
            echo "<p>New record created successfully!</p>";
        } else {
            echo "<p>Error: " . $stmt->error . "</p>";
        }
        // Close the statement
        $stmt->close();
    }
    // Close the database connection
    $conn->close();
    ?>
</body>
</html>
```

Submit Your Information

First Name:
Abebe

Last Name:
Kebede

Email:
abebe@gmail.com

Submit

New record created successfully!

Thank you!

Appreciate your action.