University of Tissemsilt
Faculty of Science & Technology
Departement of Math and Computer Science

Application Web Development
PHP & MySQL Database

2024-05-12

Lecturer

Dr. HAMDANI M

Speciality : Computer Science (ISIL)
Semester: S4

Plan

PHP and MySQL

PHP can work with a MySQL database using:

- MySQLi extension (the "i" stands for improved) :only work with MySQL databases
  - MySQLi Procedural: In the procedural approach, you use functions to execute queries and manage connections
  - MySQLi Object-oriented (OO): offers better code organization and reusability,
- PDO (PHP Data Objects) : work on 12 different database systems

Choose MySQLi for dedicated MySQL projects if performance is crucial, or PDO for portability across different databases.

Create a new database

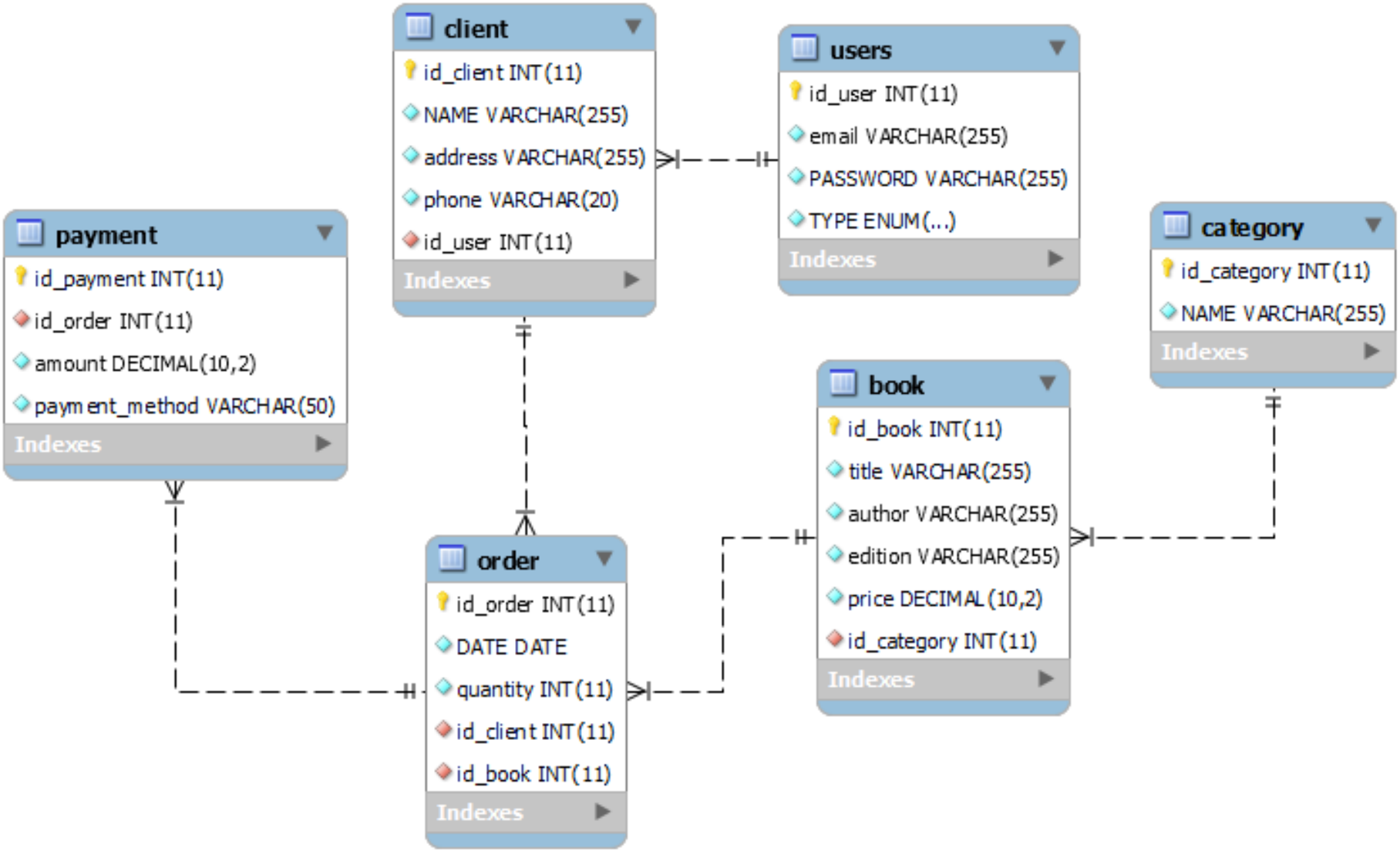
- Start XAMPP: Locate the XAMPP control panel. This is usually an icon in your system tray or a shortcut on your desktop.

  Click the "Start" buttons for both "Apache" and "MySQL" modules. The buttons will turn green when the modules are running.

- Access phpMyAdmin: Open your web browser and navigate to the following URL: http://localhost/phpmyadmin/

- Create a new database (you can create a Database /table using SQL)

Conceptual Treatment Model



```sql
-- Create the database if it doesn't exist
CREATE DATABASE IF NOT EXISTS bookdb;
-- Use the bookdb database
USE bookdb;

CREATE TABLE IF NOT EXISTS Users (
    id_user INT AUTO_INCREMENT PRIMARY KEY,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    type ENUM('client', 'vendor') NOT NULL
);

CREATE TABLE IF NOT EXISTS Client (
    id_client INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address VARCHAR(255) NOT NULL,
    phone VARCHAR(20) NOT NULL,
    id_user INT NOT NULL
);
```

```sql
CREATE TABLE IF NOT EXISTS Book (
     id_book INT AUTO_INCREMENT PRIMARY KEY,
     title VARCHAR(255) NOT NULL,
     author VARCHAR(255) NOT NULL,
     edition VARCHAR(255) NOT NULL,
     price DECIMAL(10,2) NOT NULL,
     id_category INT NOT NULL
 );
CREATE TABLE IF NOT EXISTS Category (
     id_category INT AUTO_INCREMENT PRIMARY KEY,
     name VARCHAR(255) NOT NULL
 );
CREATE TABLE IF NOT EXISTS `Order` (
     id_order INT AUTO_INCREMENT PRIMARY KEY,
     date DATE NOT NULL,
     quantity INT NOT NULL,
     id_client INT NOT NULL,
     id_book INT NOT NULL,
     id_vendor INT NOT NULL
 );
```

```
  CREATE TABLE IF NOT EXISTS Payment (
         id_payment INT AUTO_INCREMENT PRIMARY KEY,
         id_order INT NOT NULL,
         amount DECIMAL(10,2) NOT NULL,
         payment_method VARCHAR(50) NOT NULL
  );
 ALTER TABLE Client
  ADD FOREIGN KEY (id_user) REFERENCES Users(id_user);

 ALTER TABLE Book
  ADD FOREIGN KEY (id_category) REFERENCES
        Category(id_category);

ALTER TABLE `Order`
  ADD FOREIGN KEY (id_client) REFERENCES Client(id_client),
  ADD FOREIGN KEY (id_book) REFERENCES Book(id_book),
  ADD FOREIGN KEY (id_vendor) REFERENCES Users(id_user);

ALTER TABLE Payment
  ADD FOREIGN KEY (id_order) REFERENCES `Order`(id_order);
```

Connect to MySQL (MySQLi Object-Oriented)

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

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

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

Connection using Class

```php
<?php     //File : config.php
class Database
{   private $servername = "localhost";   private $username = "username";
    private  $password = "password";     private   $dbname = "dbname";
    private $conn = null;
  public function connect()  // Connect to the database
  {    if ($this->conn == null) {
       $this->conn = new mysqli($this->servername, $this->username,
                $this->password, $this->dbname);

       if ($this->conn->connect_error) // Check connection
         { die("Connection failed: " . $this->conn->connect_error); }
       echo "Connected successfully <br><br>";
    }
    return $this->conn;
  }
  public function disconnect() // Disconnect from the database
  {   if ($this->conn != null) {
       $this->conn->close();
       $this->conn = null;
       echo "<br><br>Disconnected successfully";
    }
  }
}
```

Connect to MySQL : Calling the Class

```php
<?php
require_once 'config.php';

$db = new Database();
$conn = $db->connect(); // Connect to the database

$sql = "SELECT userName, password FROM user";
$result = $conn->query($sql);

//...

$db->disconnect(); // Close the database connection
```

Persistent Database Connections(1)

- Persistent database connections maintain an open connection between the application and the database even after executing a query or a series of queries.
- Unlike regular connections, persistent connections remain open for a defined period or until explicitly closed by the application.
- Benefits include reduced server overhead by avoiding repeated connection establishment and reusability of connections for multiple successive queries.

Persistent Database Connections(2)

- Careful resource management is crucial to prevent exhausting server resources.
- Server-side configuration is necessary to enable and optimize persistent connections.
- Persistent connections are suitable for high-traffic applications where connection time is critical and numerous queries need to be executed.
- However, their usage should be judiciously evaluated based on specific application needs and overall performance considerations.

Example

```php
<?php
$server = 'localhost';
$username = 'root';
$password = '';
$database = 'test';

// Establishing a persistent database connection with MySQLi
$conn = new mysqli("p:$server", $username, $password,
       $database);

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

  // Queries and processing with the persistent connection

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

Questions ?