503073

WEB PROGRAMMING & APPLICATIONS

PHP - MySQL

Instructor: Mai Van Manh

OUTLINE

1. Introduction
2. SQL
3. MySQL
4. PHP vs MySQL

INTRODUCTION

- Why Use a Database?
- Structured Query Language

SQL: STRUCTURED QUERY LANGUAGE

- SQL is the most popular language for adding, accessing and managing content in a database.
- Pronounciation: either S - Q - L (one character at a time) or 'sequel'.
- SQL Statements: Most of the actions you need to perform on a database are done with SQL statements:
  ```
  SELECT * FROM Customers
  ```
- SQL keywords are NOT case sensitive.
- Some database systems require a semicolon at the end of each SQL statement.

Important SQL Commands

- SELECT - extracts data from a database
- UPDATE - updates data in a database
- DELETE - deletes data from a database
- INSERT INTO - inserts new data into a database
- CREATE DATABASE - creates a new database
- ALTER DATABASE - modifies a database
- CREATE TABLE - creates a new table
- ALTER TABLE - modifies a table
- DROP TABLE - deletes a table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

MYSQL

- MySQL is a freely available open source RDBMS that uses Structured Query Language.
- MySQL is an essential part of almost every open source PHP application.
- MySQL is developed, and distributed by Oracle Corporation.

MYSQL 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 includes data security layers that protect sensitive data from intruders.

WORKING WITH MYSQL

1. Download & install MySQL
2. Connect to MySQL Database.
3. Interact with a database.
   - Create a database and tables
   - Insert data into tables
   - Load data from tables
   - Update and delete data from tables
4. Close database connection.

MySQL Server Default Information

1. Hostname: e.g localhost, 127.0.0.1
2. Username: root
3. Password: (empty)
4. Port: 3306
5. Database name: optional

PHP Connect to Database Server

- PHP offers two different ways to connect to MySQL server:
  1. **MySQLi**: Improved version of MySQL.
  2. **PDO**: PHP Data Objects extensions.

PHP Data Object

- More portable and supports more than 12 different databases.
- Only Object-oriented API is supported.

MySQLi

- Only supports MySQL database.
- Provides an easier way to connect to, and execute queries on, a MySQL database server.
- MySQLi also offers a procedural API which is relatively easy for beginners to understand.
- Faster than PDO.

Open Connection: MySQLi

- Connect to server without selecting a database.

```php
$conn = new mysqli("127.0.0.1", "root", "123456");

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

echo "Connect success";
$conn->close();
```

Open Connection: MySQLi

- Connect to server then select a database.

```php
$conn = new mysqli("127.0.0.1", "root", "123456");
if ($conn->connect_error) {
    die("Kết nối thất bại: " . $conn->connect_error);
}

// 1. create database 'shopping'
// 2. then, select 'shopping' database
$conn->select_db("shopping");

// 3. insert or query data from database tables
```

Open Connection: MySQLi

- Connect to server and database at the same time.

```php
$conn = new mysqli("127.0.0.1", "root", "123456","shopping");

if ($conn->connect_error) {
    die("Kết nối thất bại: " . $conn->connect_error);
}

echo "Kết nối thành công";
$conn->close();
```

Close Connection

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

```
1 $conn->close();
```

Create Database

- Before saving or accessing the data, we need to create a database first.
- The CREATE DATABASE statement is used to create a new database in MySQL.
- The MySQLi query() method is used to execute the create database statement.

Create Database

```php
... // Khởi tạo kết nối lưu vào biến $conn

$sql = "CREATE DATABASE shopping";

if ($conn->query($sql) === TRUE) {
    echo "Tạo database thành công";
} else {
    echo "Lỗi tạo database: " . $conn->error;
}

$conn->close();
```

MySQLi query() method

- Returns FALSE on failure.
- Returns a mysqli_result object for successful SELECT, SHOW, DESCRIBE or EXPLAIN queries.
- Return TRUE for other successful queries.

Create Table

```php
$sql = "CREATE DATABASE shopping";

if ($conn->query($sql) === FALSE) {
    die ("Lỗi tạo database: " . $conn->error);
}

$conn->select_db('shopping');

// Viết lệnh tạo bảng
```

Create Table

- Use $conn->query('sql command') to create a table.

```php
$sql = "CREATE TABLE PRODUCT(ID INT PRIMARY KEY,
        NAME VARCHAR(64), PRICE INT)";

if ($conn->query($sql) === true) {
    echo "Tạo bảng thành công";
}else {
    echo "Tạo bảng thất bại" . $conn->error;
}

$conn->close();
```

Insert Data

- The INSERT INTO statement is used to insert new rows in a database table.

```
$sql = "insert into product values(1, 'iPhone 12', 1200)";
```

- Multiple records can be inserted at the same time:

```
$sql = "insert into product values
        (1, 'iPhone 12', 1200),
        (2, 'iPhone 13', 2300),
        (3, 'iPhone 14', 3500)";
```

Insert Data

```php
$sql = "insert into product values (1, 'iPhone 12', 1200)";

if ($conn->query($sql) === true) {
    echo 'Chèn dữ liệu thành công';
}else {
    echo 'Chèn thất bại: ' . $conn->error;
}
```

Insert Data

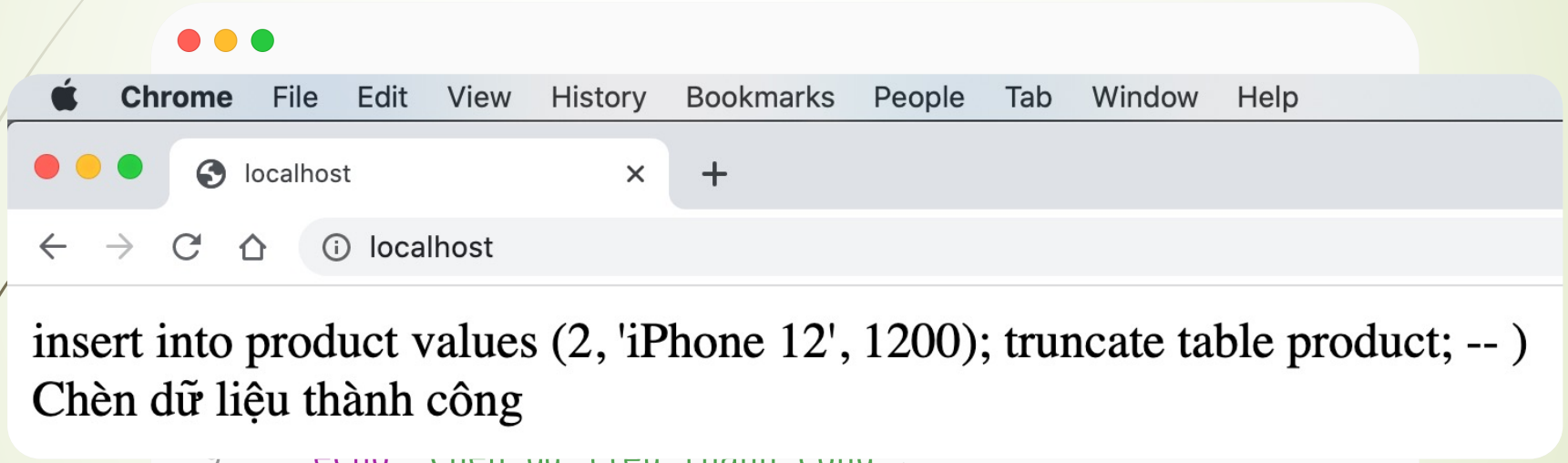
- When receiving data from the user, we put them in the insert statement.

```php
$id = 1;
$name = 'iPhone 12';
$price = 1200;
$sql = "insert into product values ($id, '$name', $price)";

if ($conn->query($sql) === true) {
    echo 'Chèn dữ liệu thành công';
}else {
    echo 'Chèn thất bại: ' . $conn->error;
}
```

Insert Data

- What if the user input is not safe?




```
10 }else {
11     echo 'Chèn thất bại: ' . $conn->error;
12 }
```

Insert Data using Prepared Statement

- A prepared statement is simply a SQL query template containing placeholder instead of the actual parameter values.
- These placeholders will be replaced by the actual values at the time of execution of the statement.

```
$sql = "insert into product values (?,?,?)";
```

Insert Data using Prepared Statement

- The prepared statement execution consists of two stages: prepare and execute.

1. Prepare — At the prepare stage a SQL statement template is created and sent to the database server. The server parses the statement template, performs a syntax check and query optimization, and stores it for later use.
2. Execute — During execute the parameter values are sent to the server. The server creates a statement from the statement template and these values to execute it.

Insert Data using Prepared Statement

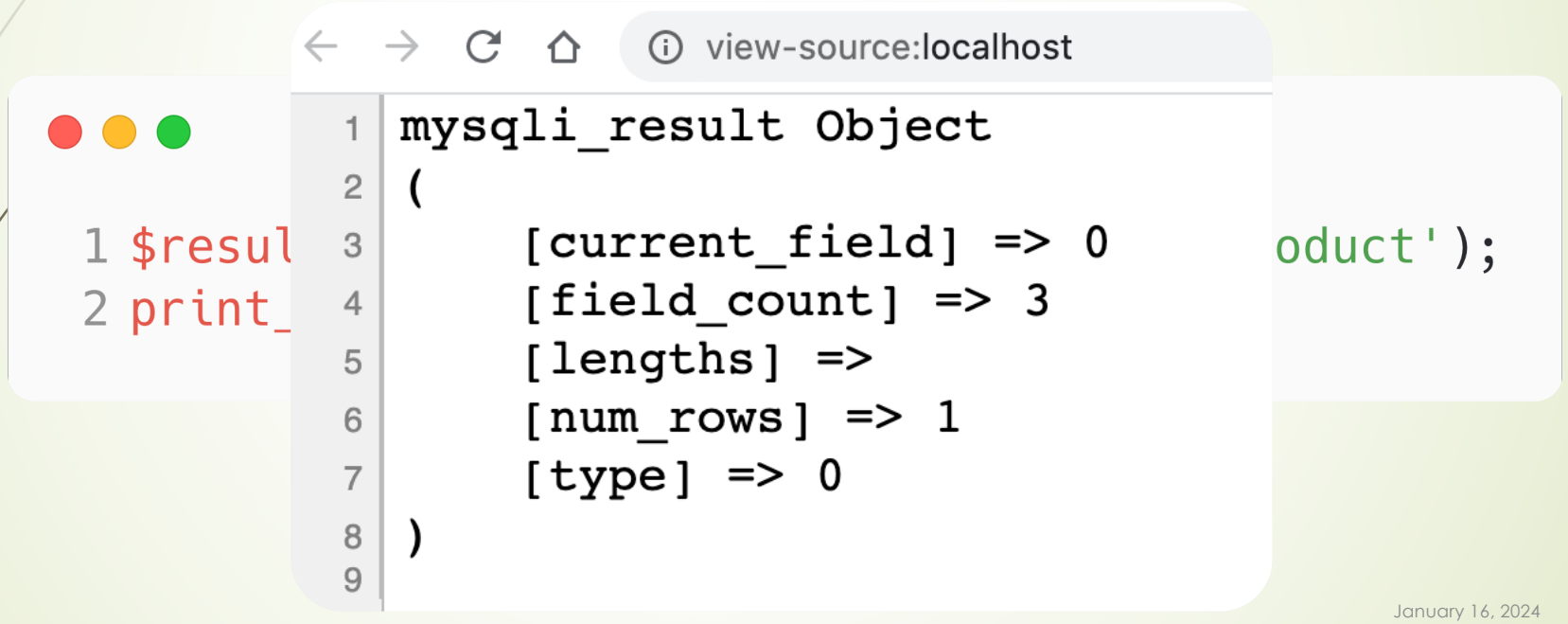
```php
id = 2;
$name = 'iPhone 12';
$price = 1200;

$sql = "insert into product values (?,?,?)";
$stm = $conn->prepare($sql); // Prepare stage
$stm->bind_param('isi', $id, $name, $price);

if ($stm->execute() === true) {
    echo 'Chèn dữ liệu thành công';
}else {
    echo 'Chèn thất bại: ' . $conn->error;
}
```

Load data from a Table

- MySQLi returns a mysqli_result object for successful SELECT command.



Load data from a Table

- If there is data, we use fetch_assoc() to read it line by line.

```
1 $resu                                              duct');
2
3 if ($
4     e
5 }else
6
7     $
8     p
9 }
```

view-source:localhost

```
1  Array
2  (
3      [id] => 1
4      [name] => iPhone 12
5      [price] => 1200
6  )
7
```

Load data from a Table

- call fetch_assoc() repeatedly to get each row:

```
1 $product = $res
2 print_r($produc
3
4 $product = $res
5 print_r($produc
6
7 $product = $res
8 print_r($produc
```

view-source:localhost

```
1  Array
2  (
3      [id] => 1
4      [name] => iPhone 12
5      [price] => 1200
6  )
7  Array
8  (
9      [id] => 2
10     [name] => iPhone XS Max
11     [price] => 900
12 )
13 Array
14 (
15     [id] => 3
16     [name] => iPad Pro
17     [price] => 1500
18 )
19
```

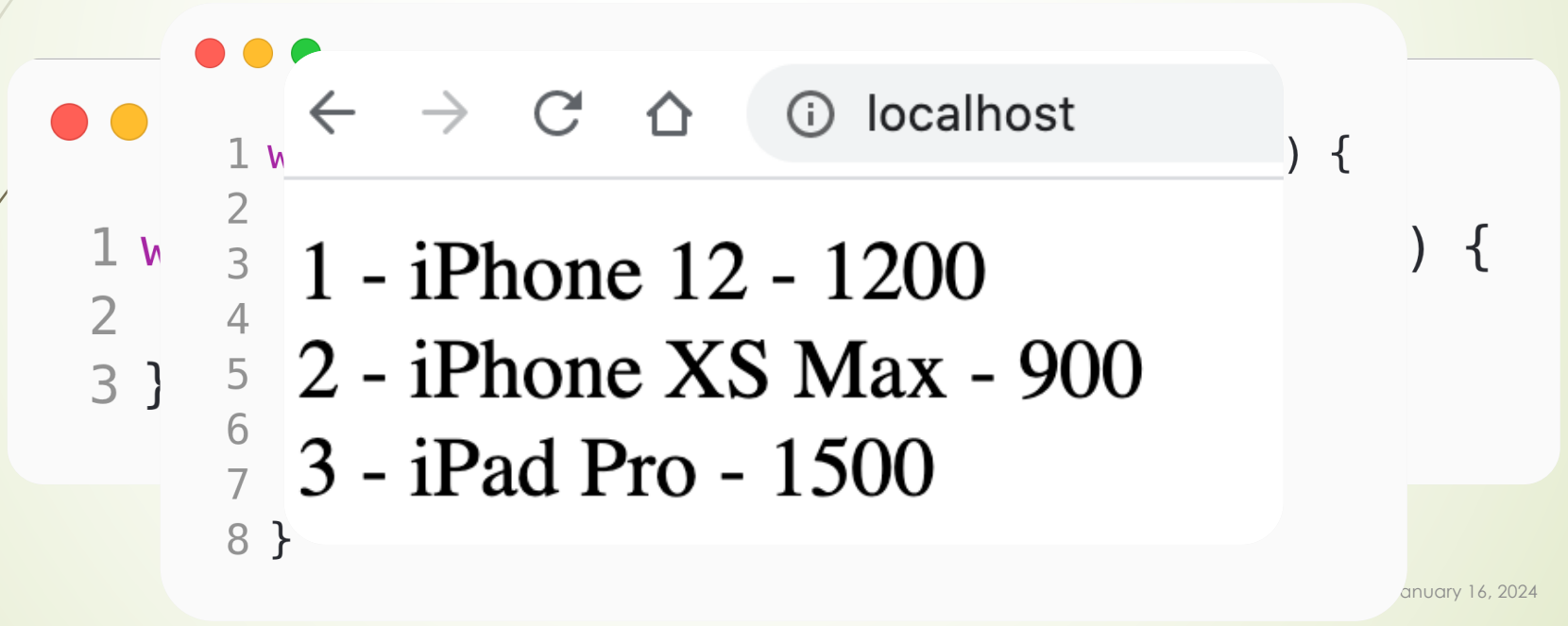
Load data from a Table

- Use while loop when we do not know how many row may be returned.

```php
while ($product = $result->fetch_assoc()) {
    print_r($product);
}
```

Load data from a Table

- Use while loop when we do not know how many row may be returned.



Load data with condition

```php
$result = $conn->query('select * from product where price >= 1500');

if ($result->num_rows == 0) {
    echo 'Không có dữ liệu';
}else {

    while ($product = $result->fetch_assoc()) {

        $id = $product['id'];
        $name = $product['name'];
        $price = $product['price'];

        echo "$id - $name - $price<br>";
    }
}
```

Load data with condition

```php
$price = 1500;
$stm = $conn->prepare('select * from product where price >= ?');
$stm->bind_param('i', $price);
$stm->execute();

$result = $stm->get_result();
if ($result->num_rows == 0) {
    echo 'Không có dữ liệu';
}else {
    // Fetch dữ liệu
}
```

PHP MySQL Demo

- Watch a demo video