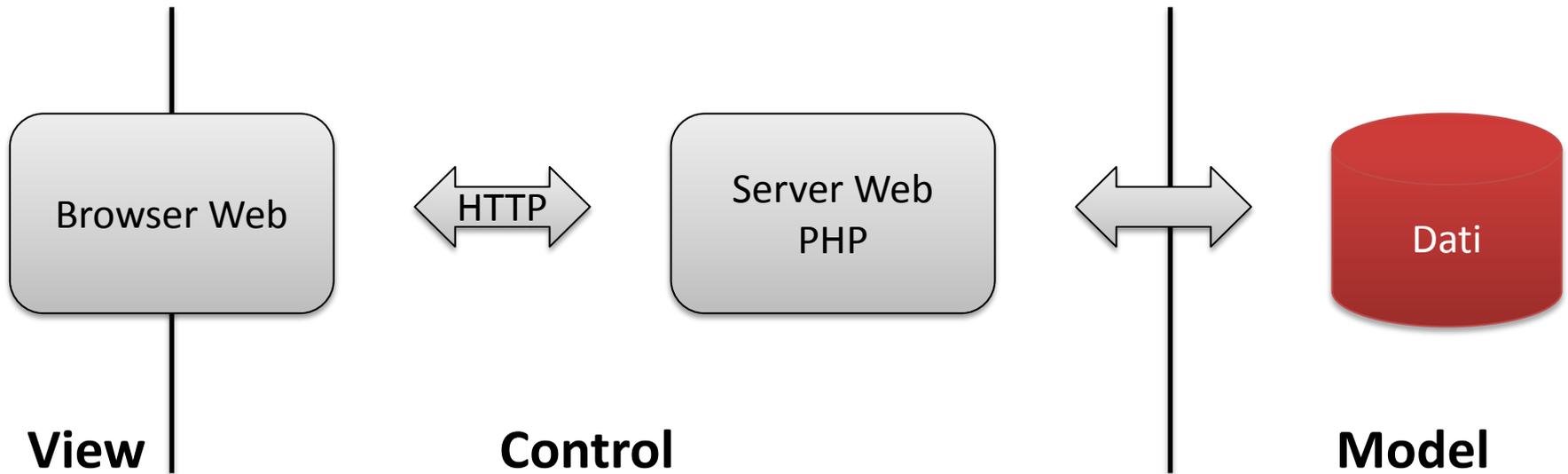


Laboratorio Progettazione Web

PHP e MySQL - Lezione 9

Andrea Marchetti – IIT-CNR
andrea.marchetti@iit.cnr.it
2011/2012

Architettura di una **applicazione Web**



PHP e Database

- Quando i dati sono molti e salvare i dati su filesystem risulta inefficiente conviene usare il supporto di una **base di dati**
- PHP fornisce delle funzioni per **accedere ai database**, ma essenzialmente si usa lo script SQL per lavorare sui DB.
- PHP è tipicamente usato con MySQL, un database free opensource (community edition), liberamente scaricabile da www.mysql.org
- E' possibile accedere anche ad altri database (Postgres, access, oracle etc)

MySQL

- MySQL è disponibile su tutte le piattaforme ed è realizzato come un server (quindi un servizio attivo che risponde su una porta) .
- Ogni **server MySQL** può essere configurato per gestire **database**, ognuno dei quali conterrà **tabelle** le quali potranno essere popolate con i **dati (record)**
- L'interazione con il server MySQL può avvenire da riga di comando (shell) digitando i comandi, o le query SQL, per creare database, tabelle, inserire dati, fare ricerche etc.
- L'interazione può avvenire anche tramite interfaccia grafica, ad esempio PhpMyAdmin

PHPMyAdmin

- Un client molto usato basato su una grafica user-friendly è PHPMyAdmin, una applicazione web fatta in PHP che permette di gestire MYSQL server via interfaccia web
- Si può liberamente scaricare da <http://www.phpmyadmin.net/> si installa come applicazione PHP (quindi php deve essere installato e il web server deve essere attivo).
- E' già incluso in EasyPHP, MAMP e XAMMP
- Se è installato sulla cartella di default del server web è tipicamente disponibile alla URL:
- <http://localhost/phpmyadmin/>
- Oppure <http://localhost/mysql> su Easyphp

PHPMyAdmin

- MySQL viene configurato di default con un utente “root” amministratore. Può essere configurato aggiungendo altri utenti e concedendo diritti sui database.
- Con il login di root possiamo accedere all’interfaccia di *phpmyadmin*, da dove possiamo creare nuovi database, nuove tabelle, inserire dati, eseguire query e tante altre funzionalità più avanzate
- Occorre ricordarsi la password di "root". Se non è stata impostata sarà null

PHPMyAdmin GUI

phpMyAdmin



(Recent tables) ...

- information_schema
- mysql
- opendata

localhost

- Databases
- SQL
- Status
- Users
- Export
- Import
- Settings
- Synchronize
- Replication
- Variables
- Charsets
- Engines

General Settings

- Change password
- Server connection collation : Collation

Appearance Settings

- Language : English
- Theme: pmahomme
- Font size: 82%
- More settings

Database server

- Server: Localhost via UNIX socket
- Software: MySQL
- Software version: 5.1.41-3ubuntu12.10 - (Ubuntu)
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8)

Web server

- Apache/2.2.14 (Ubuntu)
- Database client version: libmysql - 5.1.41
- PHP extension: mysqli

phpMyAdmin

- Version information: 4.0.0-dev, latest stable version: 3.5.0
- Documentation
- Wiki
- Official Homepage
- Contribute
- Get support
- List of changes

The phpMyAdmin configuration storage is not completely configured, some extended features have been deactivated. To find out why click [here](#).

The configuration file now needs a secret passphrase (blowfish_secret).

The `mcrypt` extension is missing. Please check your PHP configuration.

MySQL Workbench GUI

The screenshot displays the MySQL Workbench GUI. At the top, there is a menu bar with options: File, Edit, View, Database, Plugins, Scripting, Community, and Help. Below the menu bar is a toolbar with icons for Home, SQL Editor (Locale), and other functions. The main workspace is divided into several sections:

- Workbench Central:** A header section with a welcome message "Welcome to MySQL Workbench" and a link to "What's New in This Release?". It also features a row of icons for MySQL Doc Library, MySQL Bug Reporter, Workbench Team Blog, Planet MySQL, Workbench Forums, and Scripting Shell.
- Workspace:** The main area is divided into three columns:
 - SQL Development:** Includes a "New Connection" button and a list of connections: "Locale" (User: root, Host: 127.0.0.1:3306) and "localhost" (User: root, Host: localhost:3306).
 - Data Modeling:** Includes an "Open Existing EER Model" button and a list of models: "schemaDB" (Last modified Mon Nov 15 12:21:34 2010).
 - Server Administration:** Includes a "New Server Instance" button and a list of server instances: "@localhost" (Local, Type: Windows).
- Bottom Panels:** A row of buttons for various tasks:
 - Edit Table Data:** Select a connection and schema table to edit.
 - Edit SQL Script:** Open an existing SQL Script file for editing.
 - Manage Connections:** Modify connection settings or add connections.
 - Create New EER Model:** Create a new EER Model from scratch.
 - Create EER Model From Existing Database:** Create by connecting and reverse engineering.
 - Create EER Model From SQL Script:** Import an existing SQL file.
 - Manage Import / Export:** Create a dump file or restore data from a file.
 - Manage Security:** Manage user accounts and assign privileges.
 - Manage Server Instances:** Add, delete and update server instance settings.

The status bar at the bottom left shows "Ready".

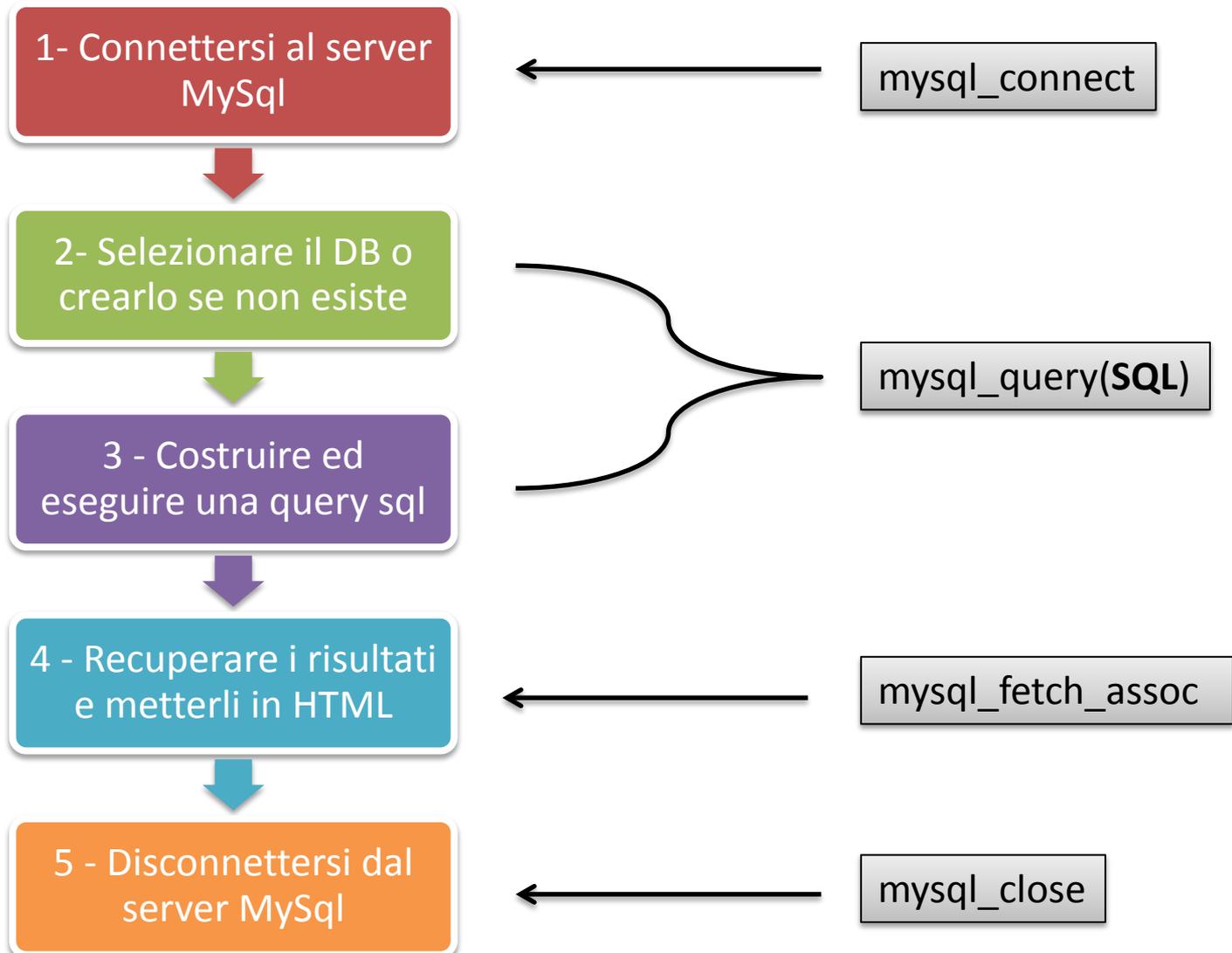
PHP e MySQL

- PHPMyAdmin è una **applicazione web** facile e intuitiva per effettuare alcune operazioni di gestione del server MySQL.
- Tutte le operazioni possibili da PHPMyAdmin si possono comunque fare anche da script PHP tramite opportuni comandi.
- In PHPMyAdmin è possibile visualizzare il codice PHP di ogni operazione
- Noi lo useremo per creare il DB, le tabelle e per popolarle di dati

PHP e MySQL

- L'accesso a MySQL server da PHP si può effettuare con semplici funzioni che PHP mette a disposizione.
- La lista delle funzioni MySQL si può trovare al seguente link
 - <http://www.php.net/manual/en/ref.mysql.php>

Sequenza dei passi per interagire con MySQL da PHP



1. Connessione al server MySql

```
<?php
// Imposto i parametri della connessione
$dbhost="localhost"; // server su cui risiede MySql
$dbuser="root";
$dbpass="XXXXXXXX"; // definita in fase di install.
```

```
//connessione al server
```

```
$conn = mysql_connect($dbhost,$dbuser,$dbpass);
if (!$conn) die("Errore mysql: ".mysql_error());
```

\$conn sarà la variabile che inserirò in tutte le successive funzioni utilizzate per accedere al server MySql. In questo modo posso accedere a più server MySQL dallo stesso programma!

die=morire stampa il msg quindi interrompe l'esecuzione del programma. Se non voglio interrompere uso **echo** o **print**

mysql_error stampa l'errore dell'ultima chiamata ad una funzione mysql

2. Selezione del DB

...

```
// Creazione del Database
```

```
$sql = "CREATE DATABASE lpwDB";
```

```
$ok = mysql_query($sql, $conn);
```

```
if (!$ok) print("impossibile creare DB: ".mysql_error());
```

```
//selezioniamo il database su cui lavorare
```

```
$sql = "USE lpwDB";
```

```
$ok = mysql_query($sql, $conn);
```

```
if (!$ok)
```

```
    die("imposs. selezionare DB: ".mysql_error());
```

...

Nota: non è richiesto il ;

`mysql_query` è la funzione che mi permette di usare direttamente il linguaggio SQL da PHP!!!

3a. Creazione Tabella

```
$sql="CREATE TABLE agenda(  
    id          int PRIMARY KEY,  
    nome       varchar(32),  
    cognome    varchar(32),  
    telefono   varchar(16));  
  
// stampa di controllo  
echo "query SQL: $sql";  
  
//esecuzione della query  
$ok=mysql_query($sql,$conn);  
if (!$ok) die("Errore query: ".mysql_error());
```

3b. Inserimento dati

```
$sql="INSERT INTO agenda(Nome, Cognome, Telefono)  
VALUES ('andrea','marchetti','050-3152649)";
```

```
// stampa di controllo
```

```
echo "query SQL: $sql";
```

```
//esecuzione della query
```

```
$ok=mysql_query($sql,$conn);
```

```
if (!$ok) die("Errore query: ".mysql_error());
```

3c. Modifica dati

```
$sql="UPDATE agenda SET nome='antonio' WHERE  
    cognome='marchetti';  
  
// stampa di controllo  
echo "query SQL: $sql";  
  
//esecuzione della query  
$ok=mysql_query($sql,$conn);  
if (!$ok) die("Errore query: ".mysql_error());
```

3d. Cancellazione dati

```
$sql="DELETE FROM agenda WHERE cognome='marchetti';  
  
// stampa di controllo  
echo "query SQL: $sql";  
  
//esecuzione della query  
$ok=mysql_query($sql,$conn);  
if (!$ok) print("Errore query: ".mysql_error());
```

3e. Selezione dati

```
$sql="SELECT * FROM agenda";

// stampa di controllo
echo "query SQL: $sql";

//esecuzione della query
$res=mysql_query($sql,$conn);
if (!$res) die("Errore query: ".mysql_error());
```

\$res contiene il risultato della SELECT.

Vedremo come estrarre i risultati nelle prossime slides.

Se c'è stato un errore, conterrà FALSE.

4. Recupero dei dati - Records

Il risultato di una SELECT in php è un array multidimensionale ovvero un array di records dove ogni record è a sua volta un array.

Abbiamo tre costrutti per accedere ai singoli record e quindi ai campi

- `mysql_fetch_row`
- `mysql_fetch_assoc`
- `mysql_fetch_array`

```
$res=mysql_query($sql,$conn);
```

Id	Nome	Cognome	Telefono
1	Mario	Rossi	
2	Giuseppe	Bianchi	
3	Marco	Verdi	
4	Luigi	Gialli	

record →



4. fetch_row

```
$sql="SELECT * FROM agenda";  
$res=mysql_query($sql,$conn);
```

Fintanto che c'è un record
il risultato dell'assegnamento
è un valore TRUE

```
while($records=mysql_fetch_row($res)) {  
    echo "ID:          $records[0]";  
    echo "Nome:       $records[1]";  
    echo "Cognome:    $records[2]";  
}
```

L'indice dell'array corrisponde
alla posizione del campo nel
record.
\$records è un array numerico

4. fetch_assoc

```
$sql="SELECT * FROM agenda";  
$res=mysql_query($sql,$conn);  
  
while($records=mysql_fetch_assoc($res)) {  
    echo "ID:          $records['id']          ";  
    echo "Nome:       $records['nome']        ";  
    echo "Cognome: $records['cognome'] ";  
}
```

L'indice dell'array corrisponde
al nome del campo nel record.
\$records è un array associativo

4. fetch_array

```
$sql="SELECT * FROM agenda";
$res=mysql_query($sql,$conn);

while($records=mysql_fetch_array($res)) {
    echo "ID:          $records['id']          ";
    echo "Nome:       $records[1]              ";
    echo "Cognome: $records['cognome'] ";
}
```

Per accedere ai campi del record
posso usare sia l'indice che il
nome del campo

4. Controlli sui risultati di una SELECT

```
$sql="SELECT * FROM agenda";
$res=mysql_query($sql,$conn);
$rows=mysql_num_rows($res);
echo "sono stati trovati $rows record";
if ($rows==0) { // controllo se la risposta è vuota
echo "non ci sono record";
}
else { // altrimenti li visualizzo
while ($records=mysql_fetch_assoc($res)) {
echo "ID: $records[id] <BR>";
echo "Nome: $records[Nome] <BR>";
echo "Cognome:$records[Cognome] <BR>";
} }
}
```

5. Chiusura della connessione

```
mysql_close($conn);
```

Riferimenti

- MySQL:
 - <http://dev.mysql.com/doc/refman/5.0/en/index.html>
 - <http://dev.mysql.com/doc/refman/5.0/en/sql-syntax.html>
- Php:
 - <http://www.php.net/manual/en/ref.mysql.php>

Prima esercitazione !

- Creare una tabella geografia all'interno del nostro DB lpwDB
 - geografia (id, nazione, capitale)
 - usare phpAdmin o uno script PHP
- Creare uno script PHP che consenta:
 - inserire un nuovo record
 - visualizzi i record già presenti
 - consenta di cancellare i record presenti
 - *consenta di editare i record presenti*

Input dati geografia

Inserimento

Nazione

Capitale

ADD

Nazione

Capitale

Italia

Roma

delete

Francia

Parigi

delete

Spagna

Madrid

delete

Inghilterra

Londra

delete

Portogallo

Lisbona

delete

TABELLA

```
CREATE TABLE geografia (  
    idgeografia INT AUTO_INCREMENT,  
    nazione VARCHAR(16) NOT NULL ,  
    capitale VARCHAR(16) NOT NULL ,  
    PRIMARY KEY ('idgeografia') ,  
    UNIQUE INDEX 'nazione_UNIQUE` ('nazione' ASC) ,  
    UNIQUE INDEX 'capitale_UNIQUE' ('capitale` ASC) )  
  
DEFAULT CHARACTER SET = latin1  
  
COLLATE = latin1_bin;
```

Esercizio

- Riscrivere il test di geografia utilizzando la tabella appena creata