

PHP + MYSQL

Corso Facile di Programmazione Web



PHP

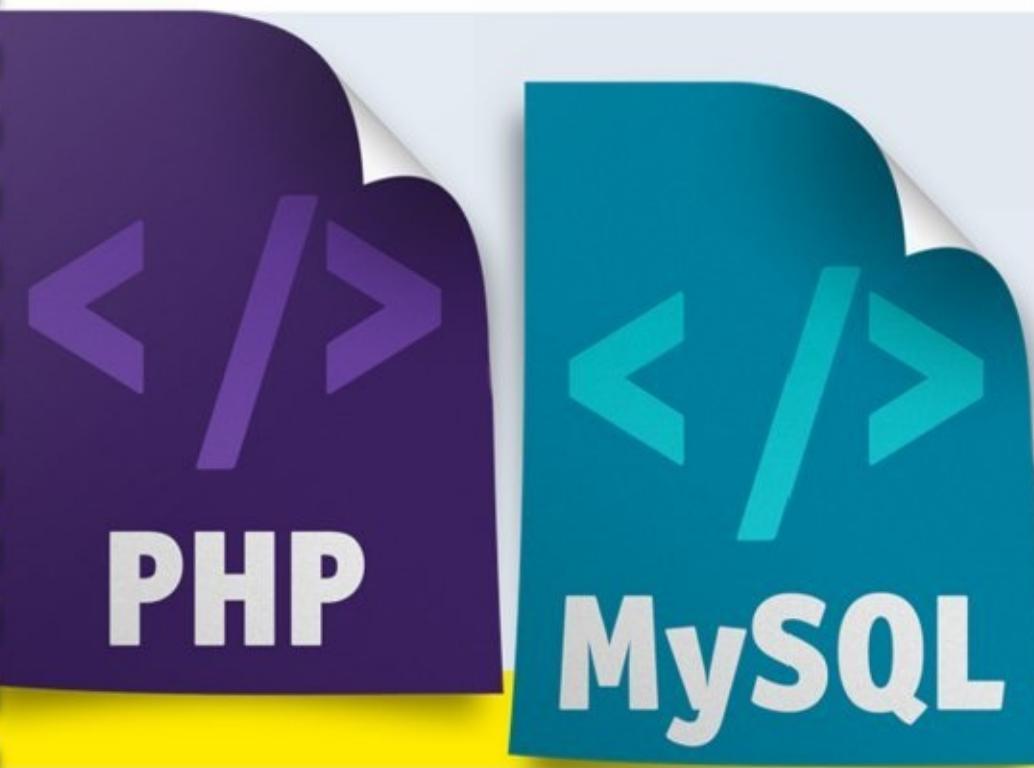


MySQL

EBOOK PRATICO AL 100%
di Daniele Venditti

PHP + MYSQL

Corso Facile
di Programmazione Web



EBOOK PRATICO AL 100%
di Daniele Venditti

PHP + MYSQL

CORSO FACILE PROGRAMMAZIONE WEB

“ Corso pratico per Webmaster...”
di Daniele Venditti

Manuale di informatica per il web

www.progettumultimediali.com

www.corsi-online.it

Questo Ebook è scritto interamente da Daniele Venditti in modo originale, è opera di ingegno tutelata per legge dal diritto di autore.

Iscriviti alla Newsletter per ricevere informazioni ed aggiornamenti.

<http://www.corsi-online.it>



A Gino...

INDICE

LEZIONE 1... Introduzione al corso. Le basi della programmazione PHP. funzionamento del linguaggio sul server. Configurazione di uno spazio hosting con server Linux e database MySql.

LEZIONE 2... Le Variabili, Il controllo di flusso if...else. Esercitazione pratica (Controllo Login)

LEZIONE 3... Il Database MySQL e lo strumento phpMyAdmin. Creazione di una tabella e inserimento dei dati

LEZIONE 4... Connessione al Database MySQL, ciclo WHILE, Funzione MYSQL_FETCH_ARRAY, Array, altre istruzioni.

LEZIONE 5... Inserire i dati nel database da interfaccia web con l'istruzione INSERT

LEZIONE 6... \$GET - \$POST – UPDATE – DELETE FROM

LEZIONE 7... Esercitazione: Lettura dati Tabella, modifica, cancellazione e inserimento dati

LEZIONE 8 Session – Realizzare una semplice area riservata con PHP e MySQL

PREFAZIONE

Il libro elettronico é scritto con un linguaggio chiaro e semplice anche adatto a chi per la prima volta si avvicina a questi nuovi concetti di apprendimento... pratico al 100% é formato da 8 lezioni utili per creare un sito web dinamico con l'ausilio del database MySql. Il corso é corredata da esempi pratici per l'inserimento, la modifica e l'eliminazione dei dati da interfaccia web phpmyadmin collegata con database su server Linux. Sarete operativi già dalla prima lezione perché il contenuto non é solo un Ebook ma un vero e proprio corso online di informatica per il web...accessibile a tutti, economico, professionale!

L'opera costituisce uno strumento indispensabile per chi vuole avvicinarsi, per professione o per conoscenza personale, al settore dei Webmaster imparando a realizzare pagine web dinamiche.

Al termine di ogni sezione il lettore sarà invitato ad eseguire delle esercitazioni. Per questo motivo abbiamo trovato più appropriato dividere il libro in Lezioni e non più comunemente in Capitoli.

Tutti i concetti elencati in questo **E-book** sono racchiusi anche in un **video corso** (indipendente) prelevabile dal portale di:

www.corsi-online.it

BUONA LETTURA...

(Daniele Venditti)

www.progettimmultimediali.com

www.corsi-online.it

LEZIONE 1

Introduzione al corso. Le basi della programmazione PHP. funzionamento del linguaggio sul server. Configurazione di uno spazio hosting con server Linux e database MySql.

Iniziamo il corso di PHP + MYSQL con l'indicazione degli strumenti necessari per affrontare il nostro percorso di webmaster.

Per effettuare le nostre prove utilizzeremo uno **spazio hosting** reale (gratuito) messo a disposizione dal provider Altevista. Il server di Altevista è costituito da un interprete PHP e un database MYSQL utile per eseguire le nostre esercitazioni.

Altro strumento indispensabile è un editor testi per mezzo del quale creare le pagine in PHP. Inizialmente utilizzeremo un semplice “Blocco note” ed in seguito preleveremo da internet un software gratuito adatto al nostro scopo.

Prima di iniziare il percorso è importante imparare il significato di applicazioni PHP **“Stand-alone”** ed **“Embedded”**

Applicazioni Stand-alone: indica un codice programmazione in grado di funzionare da solo senza alcun supporto di altri software nel quale “girare”... esempio tipico è il linguaggio C++

Applicazioni Embedded: il codice di programmazione per funzionare ha bisogno di un altro software come ad esempio un Browser web. Il linguaggio HTML, JAVACRIPT e PHP (con le dovute differenze) sono esempi di questo tipo.



Figura 1: Funzionamento di un sito web dinamico

Dalla figura precedente notiamo che gli elementi del sistema per il funzionamento di un sito web dinamico sono tre:

- Il sito web che visualizziamo per mezzo del nostro Browser.
- Il server web Apache su computer con OS Linux, all'interno del quale (Spazio Hosting) sono contenuti tutti i file per il funzionamento del nostro sito web.
- Il database MySql collegato con il server all'interno del quale sono contenute tutte le informazioni che verranno assemblate in tempo reale nel momento in cui l'utente ne fa richiesta con il suo browser.
-



HTML... In informatica l'**HyperText Markup Language (HTML)** (traduzione letterale: linguaggio a marcatori per ipertesti) è il linguaggio di markup solitamente usato per la formattazione di documenti ipertestuali disponibili nel World Wide Web sotto forma di pagine web. In generale una pagina web, per essere visibile e intelligibile sul Web, deve essere opportunamente formattata. Il linguaggio di

formattazione è l'HTML. La formattazione consiste nell'inserimento nel testo di marcatori o etichette, detti tag, che descrivono caratteristiche come la funzione, il colore, le dimensioni, la posizione relativa all'interno della pagina. Il contenuto delle pagine web solitamente consiste dunque di un documento HTML e dei file ad esso correlati che un **web browser** scarica da uno o più web server per elaborarli, interpretando il codice sorgente, al fine di generare la visualizzazione, sullo schermo del computer-client, della pagina desiderata, grazie al motore di rendering del browser stesso. **L'HTML non è un linguaggio di programmazione** (in quanto non prevede alcuna definizione di variabili, strutture dati, funzioni, strutture di controllo), **ma solamente un linguaggio di formattazione** che descrive cioè le modalità di impaginazione o visualizzazione grafica (layout) del contenuto, testuale e non, di una pagina web attraverso tag di formattazione. Tuttavia, l'HTML supporta l'inserimento di script e oggetti esterni quali immagini o filmati. Punto HTML (.html) o punto HTM (.htm) è anche l'estensione comune per riconoscere i documenti in questo formato.



LINGUAGGIO PHP... PHP (acronimo ricorsivo per PHP: Hypertext Preprocessor) è un linguaggio di scripting open source molto utilizzato, è specialmente indicato per lo sviluppo web e può essere integrato nell'HTML.

Invece di un sacco di comandi per produrre HTML, le pagine PHP contengono HTML con codice incorporato che fa "qualcosa. Il codice PHP è delimitato da speciali istruzioni di elaborazione di inizio e fine <?php e ?> che permettono di entrare e uscire dalla "modalità PHP".

Ciò che distingue PHP da altri linguaggi di scripting del tipo client-side JavaScript è che il codice viene eseguito nel server, generando HTML che sarà dopo inviato al client. Il client dovrebbe ricevere i risultati dell'esecuzione dello script, ma non potrà conoscere qual'è il codice eseguito. Potete persino configurare il vostro web server per processare tutte i vostri file HTML con PHP ed allora non ci sarebbe realmente alcun modo per gli utenti di sapere cosa avete sul vostro server. (Fonte www.php.net)

Nei cms serve per produrre pagine dinamiche cioè elaborate dal server e non dal computer dell'utente come avviene per HTML o Javascript interagendo con il database MySql che possiamo considerare come un grosso magazzino dove sono contenute tutte le informazioni del sito web.



DATABASE MySql... MySql è il database relazionale open source più diffuso al mondo. Originariamente sviluppato dalla società svedese TcX per uso interno, MySql costituisce oggi la soluzione ottimale (soprattutto in ambiente Linux) per chi è in cerca di un database veloce, flessibile, affidabile e, soprattutto, gratuito!

Nel sistema di funzionamento di un sito dinamico di tipo cms come è Drupal, il database è fondamentale in quanto tutte le informazioni del sito web sono in esso contenute e richiamate immediatamente nel momento in cui l'utente ne fa richiesta ad esempio cliccando su una specifica voce di menu della pagina web (Ricordiamo che il funzionamento di un sito dinamico è stato sinteticamente illustrato nella figura n. 1 della lezione precedente).



APACHE... è il nome della piattaforma [server Web](#) sviluppata dalla [Apache Software Foundation](#). È la piattaforma server Web modulare più diffusa, in grado di operare su una grande varietà di sistemi operativi, tra cui [UNIX/Linux](#), [Microsoft](#) e [OpenVMS](#). Apache è un software che realizza le funzioni di trasporto delle informazioni, di internetwork e di collegamento, ha il vantaggio di offrire anche funzioni di controllo per la [sicurezza](#). Apache dispone anche di moduli che comprendono gli interpreti PHP, PERL, ecc. Questi interpreti integrati in Apache vengono eseguiti con l'identità del server.

Consideriamo il nostro studio base del linguaggio PHP analizzando il comportamento della pagina web senza codice PHP (quindi una semplice pagina HTML) ed invece una struttura web che contiene anche codice PHP unitamente al linguaggio HTML.

Caso 1 (La pagina contiene solo codice HTML)

Esiste un sistema di comunicazione tra il Browser (Firefox, Safari, Opera, Chrome, Internet Explorer) ed il server dove sono contenuti i file del sito web. Non entreranno in gioco altri elementi del Server come l'interprete PHP contenuto nel software Apache o il database MySql. Un sito web creato in questo modo viene definito “**Statico**” poiché le pagine contenute nello spazio

Hosting non sono modificate dal Server prima di essere presentate all'utente che le visualizza per mezzo del Browser.

Caso 2 (La pagina contiene anche codice PHP)

La pagina viene prima Elaborata al livello del Server come ad esempio può avvenire in un Modulo contatti o di richiesta informazioni che frequentemente troviamo in molti siti web, e successivamente presentata nuovamente all'utente con azioni specifiche contenute nel codice PHP della pagina web. In questo caso possono entrare in funzione il solo Software Apache che interpreta le istruzioni o insieme ad un database (in genere MySql) dove sono archiviati i dati.

Quindi possiamo dire che una pagina web che contiene codice PHP viene prima elaborata a livello del server, può entrare in comunicazione con un database MySql e restituita all'utente che visualizzerà il tutto tramite il Browser web. Un sito web creato in questo modo viene definito "**dinamico**" poiché le informazioni vengono elaborate in tempo reale a livello del Server ed aggiunte alla struttura della pagina.

E' importante dire, come vedremo più avanti, che se la pagina web contiene codice PHP che verrà elaborato dal server, verrà presentata all'utente finale sempre in formato HTML.

Lo Spazio Hosting

Uno strumento indispensabile per eseguire le esercitazioni contenute nel corso di PHP MySql. Per il nostro lavoro utilizzeremo il Provider Altervista che gratuitamente ci mette a disposizione un server linux online corredata da database MySQL.



ALTERVISTA... è un **ISP (Internet Service Provider)** che mette a disposizione uno spazio hosting di dimensione limitata per creare il nostro sito web. Pur essendo uno spazio non professionale (almeno per la versione free) dispone di un pannello di controllo avanzato per mezzo del quale creare un sito web con uno dei cms più utilizzati come ad esempio: JOOMLA, WORDPRESS, DRUPAL etc...

E' di fondamentale importanza dunque creare un account su Altervista per poter utilizzare le nozioni delle nostre lezioni di Drupal. Per creare un nuovo profilo in Altervista, andare

sul seguente link ed attivare la procedura di registrazione facilmente individuabile sul portale. <http://it.altervista.org/>



HOSTING... In informatica si definisce hosting (dall'inglese to host, ospitare) un servizio di Rete che consiste nell'allocare su un server le pagine web di un sito, rendendolo così accessibile dalla rete Internet e ai suoi utenti.

Tale “server web”, definito “host”, è connesso ad Internet in modalità idonea a garantire l’accesso alle pagine del sito mediante il web browser dell’host client dell’utente, con identificazione dei contenuti tramite dominio ed indirizzo IP. Il servizio può essere gratuito o a pagamento, tipicamente a qualità maggiore nel secondo caso.

La fornitura di servizi di connessione ad Internet, hosting, housing, e servizi connessi, è oggi un settore economico molto specifico, compreso all’interno dell’ICT, in cui operano molte realtà nazionali, ma anche grandi aziende transnazionali. (Fonte Wikipedia)

Dunque occorre creare un account (cioè registrarsi) sul sito del fornitore raggiungibile dal seguente indirizzo url:

<http://it.altervista.org>



Figura 2: Creazione di un account su alter vista

È molto semplice...basta scegliere un nome appropriato e unico cliccare sul tasto continua, seguire le facili istruzioni successive e si riceveranno sulla casella di posta elettronica indicata tutti i dati per accedere al pannello di controllo di Altervista ed altre importanti informazioni che utilizzeremo in seguito.

Dopo aver effettuato la registrazione entriamo nel pannello di controllo con il nostro nome utente e password, clicchiamo sul pulsante accedi (a destra di Gestione File), invia nuovi files nella schermata successiva e carichiamo tutti gli oggetti esistenti nella cartella all'interno del nostro computer.

A questo punto possiamo puntare il nostro browser all'indirizzo URL scelto in fase di registrazione...il nostro piccolo sito web costituito da sole due pagine è adesso online e visibile da tutto il mondo dall'indirizzo:

www.nome_scelto.altervista.org



NOTA **index.html (index.php)...** è interessante notare che la pagina index.html o index.php che portiamo all'interno del server di Altervista sarà la prima pagina visualizzata. Questo perchè tutti I server sono configurati in modo che la pagina con quel nome specifico sarà la prima ad essere presentata all'utente una volta digitato l'indirizzo del sito web.

Una Valida alternativa ad Altervista (che noi descriveremo nel corso online) è **XAMPP** software gratuito e rilasciato per diversi sistemi operativi. Il software consiste in un web server e database MySql che verrà installato a livello locale. Tutte le prove effettuate dunque gireranno all'interno del proprio computer.

Altro strumento di lavoro utile per il nostro lavoro è avere a disposizione un

comodo software FTP che ci consentirà di trasferire comodamente i file all'interno dello spazio hosting. **Ricordiamo comunque che questa non è una operazione necessaria poiché i file possono essere trasferiti comodamente attraverso un facile “web manager” direttamente dal pannello di controllo di Altervista.**



FILEZILLA (CLIENT)... <https://filezilla-project.org/download.php>

FileZilla Client è un software libero che permette il trasferimento di file in Rete attraverso il protocollo FTP. Il programma è disponibile per GNU/Linux, Microsoft Windows, e Mac OS X. Tra i vari protocolli supportati, oltre all'FTP vi è l'SFTP, e l'FTP su SSL/TLS. Il 10 agosto del 2007 era uno dei dodici software più popolari di SourceForge di tutti i tempi.

FTP... In informatica e telecomunicazioni File Transfer Protocol (FTP) (protocollo di trasferimento file) è un protocollo per la trasmissione di dati tra host basato su TCP.

FTP è uno dei primi protocolli definiti della Rete Internet ed ha subito una lunga evoluzione negli anni. La prima specifica, sviluppata presso il MIT, risale al 1971 (RFC-114). L'attuale specifica fa riferimento all'RFC-959.

Gli obiettivi principali di FTP descritti nella sua RFC ufficiale sono:

- Promuovere la condivisione di file (programmi o dati)
- Incoraggiare l'uso indiretto o implicito di computer remoti.
- Risolvere in maniera trasparente incompatibilità tra differenti sistemi di stoccaggio file tra host.
- Trasferire dati in maniera affidabile ed efficiente.

Dopo aver descritto a livello teorico il funzionamento del linguaggio PHP e di un sistema web server, entriamo nel vivo del corso online con una semplice pagina html con codice PHP al suo interno.

A tale scopo ci serviamo inizialmente del semplice “**Blocco Note**” inserito all'interno dei sistemi operativi Windows. Per altri OS basta aprire un Editor Testi analogo.

E' importante ricordare che ogni pagina web (dalla più semplice alla più complessa) è strutturata secondo uno schema ben preciso (uguale per tutte le pagine) all'interno del quale vanno inseriti i diversi codici di programmazione

(Css, Javascript, PhP, Asp, etc...). Tale struttura è formata dalla seguente sintassi Tag (da scrivere nel Blocco Note):

```
<html>
<head>
<title>La mia prima pagina web con PhP</title>
</head>
<body>
<? Echo "La mia prima pagina web in HTML con PHP";?>
</body>
</html>
```

Salviamo il file all'interno del nostro computer con il nome:

prima.php

Trasferiamo il file all'interno del server di Altervista per mezzo del comodo “web manager” all'interno del pannello di controllo del nostro account. Dal Pannello di controllo di Altervista clicchiamo su “Accedi” in corrispondenza di “Gestione File”, quindi clicchiamo sul pulsante “Invia File” selezioniamo il file dal nostro computer “prima.php” e inviamolo all'interno dello spazio hosting.

A questo punto entriamo in una scheda del nostro Browser e digitiamo l'indirizzo URL che punterà alla pagina PHP appena creata. Nel nostro caso (voi sostituite l'indirizzo con il vostro nome utente):

<http://www.amokve.altervita.org/prima.php>



Figura 3: Risultato Finale

Se clicchiamo con il tasto destro all'interno della pagina per visualizzare il codice sorgente HTML noteremo che non c'è traccia della riga in PHP inserita nella pagina web. Dunque, il codice:

<? Echo “La mia prima pagina web in HTML con PHP”;?>

che ci consente di mostrare a video (istruzione Echo) la breve frase inserita verrà processato a livello del server e il tutto ci verrà nuovamente mostrato come una pagina HTML. L'utente finale non vedrà mai il codice in PHP inserito nella struttura della pagina web

(E' importante salvare i file in php in lettere minuscole per evitare il problema del case sensitive dei server Linux cioè della differenza di riconoscimento tra lettere maiuscole e minuscole).

LEZIONE 2

Le Variabili, Il controllo di flusso if...else. Esercitazione pratica (Controllo Login)

Oltre ad utilizzare lo strumento editor testi “Blocco Note” già contenuto all'interno dei sistemi operativi Windows può essere utile prelevare un software gratuito dalla rete più indicato per la programmazione PHP (e non solo).

Il programma che consigliamo è “**Notepad++**” prelevabile dall'indirizzo:

<http://notepad-plus-plus.org/>

FileZilla, come già illustrato brevemente nella lezione precedente, costituisce un altro importante strumento per il trasferimento dei file verso il server del nostro provider. Anche se è possibile utilizzare comode interfacce “web manager” rappresenta un software che non deve mai mancare per la sua pratica e veloce facilità di utilizzo.

Istallare la versione Client del software e non Server adatta per il sistema operativo contenuto del proprio computer. La procedura è semplice ed è possibile utilizzare il programma anche in lingua italiana.

Quando acquistiamo uno spazio hosting, il provider ci invierà tramite posta elettronica tutte le informazioni per utilizzare il Server (Spazio hosting già descritto precedentemente) dove andremo a trasferire con Filezilla tutti i file.

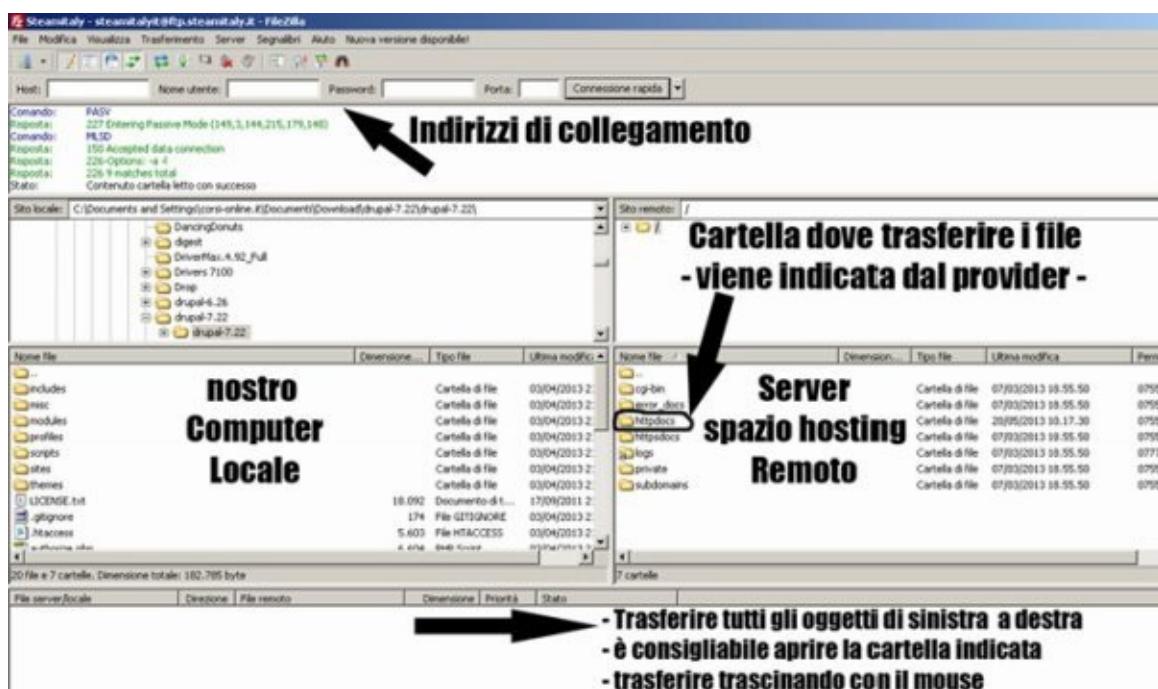


Figura 4: Funzionamento del client FTP Filezilla. Ogni provider indica con precisione la cartella dentro la quale trasferire i file.

I dati che occorrono per effettuare un connessione FTP tramite File Zilla sono i seguenti:

HOST: ftp.vostronome.altervista.org

NOME UTENTE: il nome utente di Altervista

PASSWORD: la password di Altervista

Tutti questi dati possono essere anche memorizzati in una comoda opzione di FileZilla.

Le Variabili in PHP

Un aspetto importante nel linguaggio PHP è quello di “**Variabile**”. Al fine di avere un codice più pulito ed intuitivo nel corso della programmazione, le variabili rappresentano un aspetto dal quale non è possibile prescindere (caratteristica questa comune a molti linguaggi di programmazione e non solo al PHP).

La sintassi di una variabile in PHP è ad esempio rappresentata dal seguente codice:

\$variabile=“ciao”;

Da notare il simbolo del “\$” che precede sempre il nome desiderato della variabile. La variabile è dunque un contenitore di dati, nel nostro caso la parola ciao.

Alcune caratteristiche della sintassi di una variabile è che questa:

- Non deve iniziare con uno spazio
- Non deve iniziare con un numero
- Gli spazi tra i nomi devono essere riempiti dal simbolo “_”

Eseguiamo in piccolo esempio per capire praticamente il concetto di variabile. Analizziamo il seguente codice:

```
<?php  
$nome=“Daniele”;  
$cognome=“Venditti”;  
$sitoweb=“www.corsi-online.it”;
```

```
echo "<center>Benvenuti nel mio sito web: $sitoweb </center>";  
echo "<br><center>Realizzato da $nome $cognome$ </center>";  
?>
```

Il codice in PHP si apre con la stringa “**<?php**” e si chiude con “**?>**”

Nel codice abbiamo inserito tre variabili \$nome, \$cognome, \$sitoweb che contengono i relativi dati (in termini tecnici si dice dichiarare una variabile). L’istruzione “echo” come già abbiamo visto nel precedente capitolo serve per mostrare a video i risultati del codice.

Se utilizziamo il software Notepad++ per scrivere il nostro codice (**dunque scegliere dal menu linguaggio il formato PHP**) possiamo salvare semplicemente il nostro lavoro e trasferirlo sul server del provider. Al contrario se utilizziamo un semplice editor testi come ad esempio Notepad di Windows bisogna ricordare durante il salvataggio del file di aggiungere l’estensione .php dopo il nome. Salviamo il file con il nome:

prova1.php

Effettuiamo il trasferimento del file nel server del provider (ad esempio con FileZilla) e visualizziamo il percorso con il nostro browser:

www.nomesito.altervista.org/prova1.php

Ecco risultato finale:

**Benvenuti nel mio sito web
Realizzato da Daniele Venditti**

Se mostriamo il codice sorgente della pagina non ci sarà traccia del codice PHP ma tutto sarà trasformato in HTML (dunque il codice PHP viene elaborato a livello del server e restituito all’utente finale sotto forma di semplice HTML).

È possibile anche scrivere diverse varianti per ottenere lo stesso risultato precedente come ad esempio il codice seguente:

```
<?php  
$nome="Daniele";  
$cognome="Venditti";  
$sitoweb="www.corsi-online.it";  
$nomecompleto=$nome. $cognome;  
echo "<center>Benvenuti nel mio sito web: $sitoweb </center>";  
echo "<br><center>Realizzato da $nomecompleto </center>";  
?>
```

\$nomecompleto=\$nome. \$cognome;

Sintassi per concatenare più variabili per mezzo del “.” da riutilizzare in diverse parti del codice di programmazione (Per le variabili concatenate non si usano le “”).

Controlli di Flusso (IF...ELSE)

I controlli di flusso sono molto utilizzati nei linguaggi di programmazione. Il flusso di flusso valuta una condizione, se questa condizione risulta soddisfatta il programma esegue un’azione (**IF**) altrimenti (**ELSE**) attiva un diverso procedimento.

Vediamo con un semplice esempio di spiegare il funzionamento del controllo di Flusso (If...else):

```
<?php  
$nome=“Daniele”;  
if($nome == “Mario”{  
    echo “Il nome è Mario”;  
    else “Il nome non è Mario”;}  
?>
```

In questo esempio il nome inizializzato nella variabile è “Daniele” ed in base al flusso di controllo che abbiamo inserito accade che:

Se (**if**) il nome contenuto nella variabile fosse uguale a “Mario” sul video apparirebbe “Il nome è Mario”, altrimenti (**else**) sul video comparirà “Il nome non è Mario”. Dall’analisi del listato si evince che quest’ultima condizione (**else**) sarà soddisfatta poiché il nome contenuto nella variabile non è uguale a Mario.



NOTA Operatore == In PHP l'operatore “uguale” è contrassegnato da un doppio ”==” Attenzione a non dare spazio tra i due operatori altrimenti il codice manifesterà un errore una volta trasferito all'interno del nostro spazio hosting. Inoltre se utilizzi il software consigliato Notepad++ nel nostro corso per scrivere il codice html, php etc ricorda prima di selezionare la voce “linguaggio” dal software.

Salvando il nostro lavoro con il nome **prova2.php** e trasferiamolo all'interno del server di Altervista. Il risultato finale sarà “il nome non è Mario”.

Eseguiamo adesso un esempio più complesso delle nozioni descritte precedentemente (un login con nome utente e password):

```
<html>
<head>
<title>Esempio semplice di Login</title>
</head>
<body>
<form action="login.php" method="post">
    Username: <input name="user" type="text"/>
    Password: <input name="pass" type="text"/>
    <input type="submit" value="Accedi">
</form>
</body>
</html>
```

Dopo aver scritto il codice in HTML salviamo la pagina con il nome:

login.html e portiamo la pagina all'interno del server di Altervista (per questa operazione

di trasferimento abbiamo utilizzato FileZilla, ma è possibile trasferire i file nello spazio hosting anche dal comodo pannello di controllo di Altervista).

Puntiamo con il nostro browser verso la pagina appena realizzata. Otterremo il risultato seguente:



Figura 5: il form login.html

Per creare una funzionalità nel Modulo, come si evince anche dal codice html, occorre creare il file login.php e dunque per mezzo del nostro editor preferito digitiamo il codice relativo al file in questione:

```
<?php  
$user=$_REQUEST["user"];  
$password=$_REQUEST["pass"];  
echo $user . "/" . $password;  
if($user == "mario" && $password == "rossi")  
{  
echo " – Sei stato accettato. Puoi Proseguire";}  
else{  
echo " – Login errato. Torna indietro e riprova";}  
?>
```

Nel codice abbiamo dichiarato due variabili “\$user” e “\$password” ed introdotto l’istruzione “\$_REQUEST”.

Quest’ultima istruzione in php serve per richiamare il testo inserito all’interno dei campi del form (quindi user e password). Il codice è scritto in maiuscolo.

L’istruzione “echo” serve per visualizzare (impropriamente a volte definito stampare) a video il nome utente e la password. Da notare il codice . “/” . che in php significa concatenato cioè insieme. (quindi visualizzare a video il valore della variabile \$user e \$password).

Con l’istruzione di flusso “if” “else” iniziamo a controllare i dati inseriti. (notare anche l’istruzione “&&” che in php significa “and” – e).

Dunque se nel nostro form verranno inseriti i dati in combinazione: user=mario e pass=rossi, allora il controllo di flusso (if) e a video apparirà “Sei stato accettato. Puoi proseguire” altrimenti (else) “Login errato. Torna indietro e riprova”.

Salviamo in file creato in login.php e trasferiamolo all'interno del server di Altervista. Proviamo l'esercitazione inserendo nel form prima una combinazione di dati login errata (esempio mario bianchi) e successivamente con il caso previsto (mario rossi).

LEZIONE 3

Il Database MySQL e lo strumento phpMyAdmin. Creazione di una tabella e inserimento dei dati

Nella lezione precedente abbiamo affrontato un semplice esempio che ci ha permesso di capire il funzionamento di un'area riservata gestita con solo coldice php e Form di inserimento dati in html.

In questa lezione invece ampliamo le nostre conoscenza adattando lo stesso esempio per mezzo di un database MySQL.

Un database è una caratteristica necessaria per tutte le applicazione moderne in informatica. È l'evoluzione dei vecchi archivi cartacei in modalità digitale ma che consentono maggiore flessibilità e velocità di utilizzo dei dati perché possono essere interconnessi tra di loro, possono essere interrogati tramite delle Query (ricerche nel database), e possono facilmente adattarsi a nuove esigenze operative.

Tutte le informazioni in un database sono contenute in una o più **Tabelle** collegate tra di loro (relazioni tra tabelle). Le tabelle all'interno del database possono essere interrogate per mezzo di **Query**, cioè domande formulate appositamente per ricevere le risposte che desideriamo. Le Query sono create con un linguaggio dedicato che si chiama **SQL** (Structured Query Language, linguaggio di interrogazione strutturato) per mezzo del quale è possibile gestire tutto il database. I siti web dinamici (Joomla, Drupal, Wordpress...)

utilizzano un collegamento al database per il loro funzionamento.

ID	Società	Nome di battesimo	Cognome
1	Società A	Anna	Bedecs
2	Società B	Antonio	Gratacos Solsona
3	Società C	Thomas	Axen

Figura 6: Tabella di un Database

La figura precedente mostra la struttura logica di un database. I termini che si utilizzano per indicare normalmente gli elementi della tabella cambiano leggermente. Ad esempio le Righe vengono definite **Record** (1 in figura) e le colonne invece **campi** (2 in figura).

Lo spazio Hosting di Altervista contiene anche un database MySQL che utilizzeremo per le esercitazioni pratiche in questo Ebook. Dunque portiamoci all'interno del pannello di controllo di Altervista, e nella bacheca troveremo un link che ci permetterà di attingere a tutti i dati utili per collegare i file php con il database MySQL “**Gestisci database**” e un link collegato all'applicazione grafica “**Accedi a php**” che ci consente di gestire facilmente un database MySQL.

Clicchiamo su “Accedi a phpMyAdmin” per gestire il database che Altervista ci mette a disposizione (se il sistema dovesse chiederci dati di login, inserire User e Pass per l'accesso al pannello di controllo del provider).

Figura 7: Pagina iniziale di phpMyAdmin con il database già creato da Altervista

Notiamo che il database (my_amokve) è stato già creato automaticamente da Altervista. Clicchiamo sul nome del database e procediamo con il nostro esercizio che parte dalla creazione di una nuova tabella:

Il nostro scopo è creare una tabella con il nome “utenti” con tre campi **IdUtente** (chiave primaria), **User**, **Password**.

Il campo “**Chiave Primaria**” è un elemento che riveste una grande importanza all'interno dei database. Serve per identificare in modo univoco un record di una tabella. Una specie di codice unico per ogni informazione inserita nel record. Ad esempio se per ipotesi

esistessero informazioni simili queste sarebbero differenziate dal codice chiave primaria. Oltre a questo, la chiave primaria è utilizzata anche per creare relazioni tra più tabelle del database.

La Chiave primaria può essere inserita manualmente oppure (come solitamente avviene) incrementata automaticamente tramite un'opzione del **tipo di dato** da selezionare in fase di creazione del database.

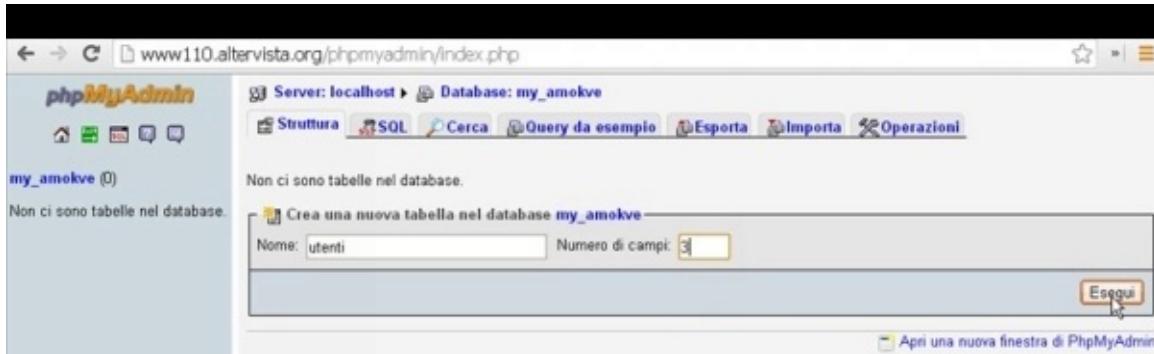


Figura 8: Creazione della tabella utenti.

Come mostrato nella figura 8 abbiamo creato la tabella utenti con tre campi (colonne).

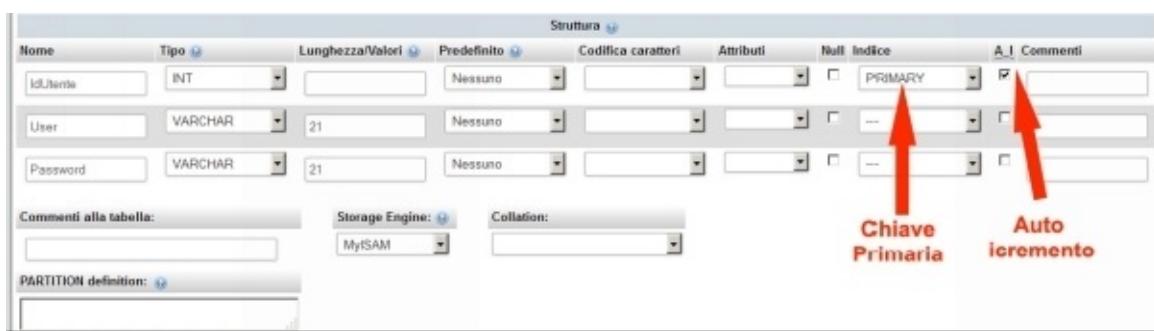


Figura 9: Creazione della tabella utenti.

In figura 9 la configurazione dei campi della nostra nuova tabella:

IdUtente: Tipo “INT” (è un valore numerico intero che si auto incrementa). Questo campo è impostato come chiave primaria. **Importante per consentire l’auto incremento della chiave primaria bisogna cliccare nella casella A_I “auto_increment”.**

User: Tipo “VARCHAR” (possibilità di inserire caratteri nel record). Nel nostro caso è stato impostato a 21 caratteri.

Password: Tipo “VARCHAR” (possibilità di inserire caratteri nel record). Nel nostro caso è stato impostato a 21 caratteri.

Inoltre è importante sapere che per creare la nostra tabella bisogna cliccare sul pulsante **salva** e non **esegui** altrimenti viene aggiunto un nuovo campo.

i Tabella `my_amokve`.`utenti` è stato creato.

query SQL:

```
CREATE TABLE `my_amokve`.`utenti` (
  `IdUtente` INT NOT NULL ,
  `User` VARCHAR(21) NOT NULL ,
  `Password` VARCHAR(21) NOT NULL ,
  PRIMARY KEY (`IdUtente`)
) ENGINE = MYISAM
```

codice SQL puro ← **interfaccia grafica phpmyadmin** ↓

Campo	Tipo	Collation	Attributi	Null	Predefinito	Extra
<input type="checkbox"/> IdUtente	int(11)			No		
<input type="checkbox"/> User	varchar(21)	latin1_swedish_ci		No		
<input type="checkbox"/> Password	varchar(21)	latin1_swedish_ci		No		

Selezione tutti / Deseleziona tutti Se selezionati:

Visualizza per stampa Proponi la struttura della tabella

Aggiungi 1 campo(i) Alla fine della tabella All'inizio della tabella Dopo IdUtente

Figura 10: Creazione della tabella utenti.

Nella schermata successiva, rappresentata dalla figura 10, è da notare il codice in puro SQL, che noi abbiamo omesso di scrivere, perché aiutati dall'interfaccia grafica di phpMyAdmin per mezzo del quale è possibile gestire tutto il database.

Cliccando sulla voce di menu “inserisci” (in alto) provvediamo a inserire i dati all'interno dei record della nostra tabella.

Server: localhost > Database: my_amokve > Tabella: utenti

Mostra **Struttura** **SQL** **Cerca** **Inserisci** **Esporta** **Importa**

Elimina

Campo	Tipo	Funzione	Null	Valore
IdUtente	int(11)			
User	varchar(21)			mario
Password	varchar(21)			rossi

Esegui

Ignora

Campo	Tipo	Funzione	Null	Valore
IdUtente	int(11)			
User	varchar(21)			carlo
Password	varchar(21)			bianchi

Esegui

Inserisci come nuova riga **e quindi** **Indietro** **Esegui** **Riavvia**

Riprendi inserimento con la riga **2**

Figura 11: Inserimento dati.

I dati possono essere inseriti singolarmente, oppure in blocco deselezionando l'opzione "Ignora" come in figura 11. Clicchiamo su Esegui. Per visualizzare i dati inseriti, andare sulla voce "Mostra" nel menu superiore:

Mostra: 30 righe a partire da 0

in modalità orizzontale e ripeti gli headers dopo 100 celle

Ordina per chiave: Nessuno

IdUtente	User	Password
1	mario	rossi
2	carlo	bianchi
3	mauro	vitale
4	paola	grossi

Selezione tutti / Deseleziona tutti Se selezionati: **Mostra: 30 righe a partire da 0**

in modalità orizzontale e ripeti gli headers dopo 100 celle

Figura 12: Visualizzazione dei dati

LEZIONE 4

**Connessione al Database MySQL, ciclo WHILE, Funzione
MySQL_fetch_array, Array, altre istruzioni.**

Variabili Accesso Database

```
// nome host o indirizzo server  
$host = "localhost";  
// nome del database al quale ci conserveremo  
$database = "nome nostro database";  
// username o login  
$user = "nome user";  
// password  
$password = "password di accesso database";
```

Figura 13: Parametri di connessione al database MySQL

Come collegare i file php ad database MySQL?

Questa è la naturale domanda che ci poniamo dopo aver appreso gli elementi delle lezioni precedenti. Abbiamo parlato di file in php, abbiamo parlato di database MySQL e dell'applicazione utile alla sua gestione che è phpMyAdmin adesso analizziamo come trasferire le informazioni da php al database.

L'immagine precedente rappresenta la sintesi di questo procedimento illustrando gli elementi che bisogna conoscere per unire php al database MySQL.

I parametri descritti nell'immagine precedente ci vengono assegnati automaticamente dal Provider (dunque nel nostro caso Altervista). Per recuperarli entriamo all'interno del pannello di controllo di Altervista, dalla Bacheca clicchiamo su “**Gestisci Database**” e visualizzeremo tutti i parametri necessari per collegare php con il database del nostro Provider (Credenziali di accesso).

Di seguito una serie di funzioni in php che utilizzeremo all'interno del codice per gestire la connessione con un database mysql:

Mysql_connect / Mysql_select_db

```
$connect=mysql_connect($host,$user,$password);  
mysql_select_db($database);
```

Figura 14: funzione di connessione (mysql_connect) e funzione di selezione del database (mysql_select_db).

Accesso database

```
$connect=mysql_connect("localhost","nome user","password") or  
die("Impossibile connettersi al server $host.");  
  
mysql_select_db($database) or die("Impossibile connettersi al  
database $database");
```

Figura 15: Gestione degli errori di collegamento (apertura connessione o selezione del database)

Chiusura connessione database

```
Mysql_close($database)
```

Figura 16: Funzione di chiusura connessione al database

Di seguito il codice in PHP che legge i dati all'interno del database mysql (dati inseriti nella lezione precedente).

```
<?php
```

```
//Dichiarazioni variabili per la connessione al database
```

```
$host = 'localhost';
```

```

$User = 'amokve';
$password = "";
$database = 'my_amokve';

// Connessione al database server
$connessione= mysql_connect($host, $User, $password) or die ("impossibile
connettersi al server");

// Selezione del database
mysql_select_db($database) or die ("impossibile connettersi al database
$database");

// Creazione interrogazione al database
$query = "SELECT * FROM utenti";

// Lancio della query (interrogazione)
$resultado = mysql_query($query);

//stampo a video i dati esistenti

while($dati = mysql_fetch_array($risultato))
{
    echo "Id utente: ".$dati['IdUtente']."<br/>";
    echo "user: ".$dati['User']."<br/>";
    echo "password: " . $dati['Password']."<br/><br/>";
}

?>

```

Analizziamo il codice appena scritto.



NOTA Per chi utilizza il software **NOTE PAD++** consigliato in questo libro: Prima di scrivere il codice in php con questo programma è importante selezionare dalla voce di menu “linguaggio” in corrispondenza della lettera “P” PHP. Naturalmente è possibile utilizzare anche un editor php diverso da Notepad++ (basta anche un semplice editor testi

in txt per Pc, Mac, o altri sistemi operativi. E' importante però salvare i file in “nome.php”).

Abbiamo già indicato che il codice in PHP va inserito fra i tag:

<?php

...

?>

Tutti gli elementi del listato che sono preceduti dal simbolo “//” introducono un commento cioè una spiegazione di ciò che si vuole realizzare con una parte di codice. Non vengono elaborati dall’interprete php ma servono solo per dare ordine e organizzazione a tutto il codice in php, magari in corrispondenza di un listato complesso per numero di righe (questa è una caratteristica comune a molti linguaggi di programmazione). Esempio:

//Dichiarazioni variabili per la connessione al database

Le istruzioni successive del nostro codice stabiliscono le variabili per la connessione con il database MySQL:

```
$host = 'localhost';  
$user = 'amokve';  
$password = " ";  
$database = 'my_amokve';
```

Come indicato in precedenza questi dati ci vengono forniti dal Provider (dunque Altervista). Da notare il campo “**password**” del database che può essere lasciato vuoto.

Abbiamo racchiuso queste informazioni all’interno di variabili che utilizzeremo nei passaggi successivi.

// Connessione al database server

```
$connessione= mysql_connect($host, $user, $password) or die ("impossibile  
connettersi al server");
```

Apriamo la connessione al server del database MySQL con la funzione “**mysql_connect**” con gestione dell’errore in caso di mancato collegamento. Tutta la funzione è racchiusa nella variabile “**\$connessione**” che utilizzeremo nei passaggi successivi.

```
// Selezione del database
```

```
mysql_select_db($database) or die ("impossibile connettersi al database  
$database");
```

Stesso significato della funzione precedente solo che qui viene identificato il database specifico che stiamo utilizzando attraverso la funzione “**mysql_select_db**”. Dunque abbiamo due tipi di connessioni: Quella al server MySQL e quella al Database specifico contenuto nel server.

```
// Creazione interrogazione al database
```

```
$query = "SELECT * FROM utenti";
```

Con questa istruzione racchiusa nella variabile “**\$query**” viene eseguita una ricerca o interrogazione (**query**) selezionando la tabella “utenti” del nostro database. Da notare che il codice “**SELECT * FROM utenti**” costituisce un tipico esempio di istruzione Sql (cioè il linguaggio che serve a gestire i database). In particolare questa istruzione si traduce nel seguente modo:

SELECT (*seleziona*)

***** (*Tutti i campi – questo è il significato dell'asterisco*)

FROM (*dalla tabella*)

utenti (*il nome della nostra tabella*)

```
// Lancio della query (interrogazione)
```

```
$risultato = mysql_query($query);
```

Con questa istruzione “**mysql_query**” lanciamo la query precedentemente definita precedentemente. Un pò come premere invio. L’istruzione è definita nella variabile “**\$risultato**” che utilizzeremo nel seguito del listato.

```
//stampo a video i dati esistenti
```

```

while($dati = mysql_fetch_array($risultato))
{
    echo "Id utente: " . $dati['IdUtente']. "<br/>";
    echo "user: " . $dati['User']. "<br/>";
    echo "password: " . $dati['Password']. "<br/><br/>";
}

```

Con questa parte di codice stampiamo (o meglio visualizziamo a video) i dati estrapolati dal database mysql. Notiamo:

Ciclo “**While**” e la funzione:

mysql_fetch_array

che meritano degli approfondimenti.



Figura 17: Funzionamento del ciclo WHILE e della funzione MySQL_FETCH_ARRAY

Nella figura 17 sono rappresentati tutti i dati che si trovano nella tabella mysql (e che anche voi potete provare a inserire utilizzando le istruzioni delle lezioni precedenti), e che vengono riproposti dalla funzione MySQL_FETCH_ARRAY al linguaggio PHP. In pratica questa funzione prende i dati contenuti nella tabella trasformandoli in istruzioni comprensibili per il PHP che in questo caso viene a conoscenza dell'esistenza di una tabella con dati da elaborare all'interno del database mysql (se volessimo fare un paragone, la funzione, è come se fosse una specie di specchio che riflette al php i dati contenuti nella tabella del database).

Tutti i dati contenuti nella tabella ricreata dalla funzione MySQL_FETCH_ARRAY sono letti dal php riga per riga (ovvero record per record) e questa azione è svolta dalla funzione del ciclo WHILE.

Dunque nell'ultima parte del nostro listato, il ciclo WHILE legge i dati contenuti nella tabella e contenuti nella variabile “\$dati” e mostrerà a video i risultati del primo record della tabella. L'istruzione “echo” serve appunto a mostre a video (si dice impropriamente stampare) i dati contenuti nel record.

Il ciclo WHILE non si fermerà fino a quando terminerà di leggere i record contenuti nella tabella generati dalla funzione MYSQL_FETCH_ARRAY.

Una nota...l'istruzione
 è tipica del linguaggio html e serve per mandare a capo le istruzioni. Per questo motivo i dati a video verranno mostrati in colonna.

Salviamo il file del listato precedente con il nome “leggitutto.php” e portiamolo all'interno del server di Altervista.

A questo punto apriamo l'indirizzo URL corrispondente al file appena caricato. Esempio:

www.vostronome.altervista.org/leggitutto.php

Id utente: 1
user: mario
password: rossi

Id utente: 2
user: carlo
password: bianchi

Id utente: 3
user: mauro
password: vitale

Id utente: 4
user: paola
password: grossi

Figura 18: Risultato finale mostrato a video – esiste uno spazio maggiore fra i record perché nel listato abbiamo inserito un doppio
 nell'ultima istruzione.

Array (Vettore)

```
<?php  
$nome = array('sara ','luca ','sandra ');  
$conta = count($nome);  
for($a=0;$a<$conta;$a++){  
echo $nome[$a]. '<br/>';  
}  
?>
```

Un Array è una variabile complessa dove le informazioni sono immagazzinate in una struttura di tipo chiave o indice che corrisponde ad un determinato valore.

Per esempio il codice precedente rappresenta un Vettore. In particolare:

```
$nome = array('sara ','luca ','sandra ');
```

Nella variabile “**\$nome**” è stato creato un vettore contenente tre nomi. Scrivere un codice in questo modo viene interpretato dal php nel seguente modo:

```
<?php  
$nome[0]='sara';  
$nome[1]='luca';  
$nome[2]='sandra';  
?>
```

Al primo nome (**sara**) viene assegnata automaticamente un indice numerico (o chiave) corrispondente al numero **[0]**, al secondo nome (**luca**) viene assegnato l’indice numerico **[1]**, al terzo nome (**sandra**) l’indice numerico **[2]**.

Un Array o Vettore può essere costruito anche in altri modi. Abbiamo preferito questa sintassi in quanto sono stati introdotti nel codice nuove istruzioni in php come la funzione “**count**” o il “**ciclo for**”

```
$conta = count($nome);
```

Nel codice, la variabile “**\$conta**” deve risultare uguale alla funzione “**count (\$nome)**”, funzione che conteggia tutti gli elementi contenuti nel vettore e racchiusi nella variabile “**\$nome**”

In seguito abbiamo inserito un **ciclo for** che come concetto è simile al ciclo While che abbiamo descritto precedentemente (anche se la sintassi del codice è leggermente diversa):

```
for($a=0;$a<$conta;$a++){  
echo $nome[$a].‘<br/>’;  
}
```

Il **ciclo for** va a leggere ciclicamente i dati contenuti nel vettore e con l’istruzione **echo** mostra a video i risultati di questa lettura.

In particolare nel ciclo for abbiamo dichiarato una nuova variabile “**\$a=0**” (perchè questo è il primo indice numerico che rappresenta la prima informazione del vettore - sara) e se questa variabile (zero nel primo ciclo for) è minore dei valori contenuti nel vettore e rappresentati dalla variabile “**\$conta**” (che rappresenta tre valori nel nostro caso), la variabile “**\$a**” sarà ulteriormente incrementata tramite l’istruzione “**\$a++**” fino a quando non raggiungerà lo stesso indice dei valori contenuti nell’array.

Salviamo il codice iniziale del vettore con il nome “**array.php**” e trasferiamolo all’interno dello spazio hosting di Altervista. Puntiamo il browser sul file ed otterremo il seguente risultato:

```
sara  
luca  
sandra
```

l’esempio appena descritto rappresenta un caso tipico di **Vettore numerico** dove la chiave o l’indice dei dati è rappresentato da numeri [0]; [1]; [2].

Ci sono invece casi di **Array associativi** dove l’indice dei dati non è rappresentato da numeri bensì da stringhe (cioè da testo). E’ il caso ad esempio del vettore generato dalla funzione **mysql_fetch_array** già incontrata, dove gli indici sono rappresentati dai nomi dei campi della tabella del database.

In basso il codice di un tipico esempio di Array associativo:

```
<?php  
$nome[‘nome1’] = ‘sara’;  
$nome[‘nome2’] = ‘luca’;  
$nome[‘nome3’] = ‘sandra’;  
foreach ( $nome as $chiave => $valore ) {  
    echo $chiave.“—”.$valore.“<br/>”;  
}  
?>
```

Nel listato precedente compare una nuova istruzione in php:

```
foreach ( $nome as $chiave => $valore)
```

Anche questa istruzione (**foreach**) viene utilizzata per leggere i dati contenuti all'interno di un vettore come per il ciclo for. Ma con questa differenza:

Ciclo for: legge solo indici numerici

Ciclo foreach: legge sia indici numerici che stringa (testuali)

Salviamo il codice con il nome “arrayass.php”, trasferiamo il file all'interno di Altervista. Il risultato a video sarà:

nome1–sara
nome2–luca
nome3–sandra

LEZIONE 5

Inserire i dati nel database da interfaccia web con l'istruzione INSERT

In questa lezione procederemo all'inserimento dei dati nel database MySQL da interfaccia web, adattando a tale scopo il modulo login.html incontrato all'inizio del nostro percorso.

Prendiamo il file **login.html** ed effettuiamo una copia. Rinominiamo il nuovo file in **insert.html** e adattiamo il nuovo modulo per inserire i dati nel database.

In basso il codice del modulo insert.html (modificato), simile al form login.html
Questo file va salvato e caricato nello spazio hosting.

```
<html>
<head>
<title>Esempio inserimento dati</title>
</head>
<body>
<form action="insert.php" method="post">
    Username: <input name="user" type="text"/>
    Password: <input name="pass" type="text"/>
    <input type="submit" value="Inserisci">
</form>
</body>
</html>
```

Il segreto è adesso creare il file **insert.php** che provvederà ad inserire i dati nella tabella utenti del database mysql.

```
<html>
<head>
<title>inserisci dati</title>
</head>
<body>
<?php
// CONNESSIONE AL DATABASE
// Dichiarazioni variabili per la connessione al database
$host = 'localhost';
$user = 'amokve';
$password = "";
$database = 'my_amokve';
```

```
// Connessione al database server
$connessione= mysql_connect($host, $user, $password)
or die ("impossibile connettersi al server");

// Selezione del database
mysql_select_db($database)
or die ("impossibile connettersi al database $database");

// INSERISCO I NUOVI DATI

// Creo le variabili e carico i dati inviati dallla pagina insert.html
$user= $_POST['user'];
$pass= $_POST['pass'];

// preparo la query di inserimento dei dati

//INSERT INTO "nome_della_tabella" ("nome colonna 1", "nome_colonna2", ...)
//VALUES ("valore 1", "valore 2", ...)

$query_insert = "INSERT INTO utenti (User, Password) VALUES ('$user',
'$pass')";

// lancio la query

$risultato_insert = mysql_query($query_insert);
// controllo l'esito

if (!$risultato_insert) {
die("Errore nella query $query_insert: ". mysql_error());
}

// chiudo la connessione a MySQL
mysql_close();
?>
<p>I tuoi dati sono stati inseriti correttamente</p>
```

```
</body>
```

```
</html>
```

La prima parte del codice “**CONNESIONE AL DATABASE**” riguarda la gestione al server mysql e al database. Abbiamo ampiamente parlato di queste impostazioni nelle lezioni precedenti.

La seconda parte del codice riguarda l’inserimento dei nuovi dati “**INSERISCO I NUOVI DATI**” :

```
$user= $_POST[‘user’];
```

```
$pass= $_POST[‘pass’];
```

Abbiamo creato due variabili. Importante è l’istruzione in php **\$_POST** serve per richiamare le istruzioni inserite nei campi: user e pass.

```
$query_insert = “INSERT INTO utenti (User, Password) VALUES (‘$user’, ‘$pass’);
```

Prepariamo la query per inserire i dati nella tabella utenti. Query di inserimento dati. Tutto racchiuso nella variabile **\$query_insert**.

Il codice va così tradotto: inserisci (INSERT) all’interno della tabella utenti (INTO utenti) ed in particolare nel campo (User, Password) i valori (VALUES) racchiusi nelle variabili (‘\$user’, ‘\$pass’) che in sintesi sono i dati inseriti nel modulo insert.html.

```
$risultato_insert = mysql_query($query_insert);
```

Lanciamo la query con l’istruzione **mysql_query**

```
if (!$risultato_insert) {  
    die(“Errore nella query $query_insert: “ . mysql_error());  
}
```

Viene effettuato un controllo della query. Da notare il punto esclamativo iniziale “!” che in php ha un significato di negazione. Tutto va tradotto in questo modo:

se (if) risultano della query (racchiuso nella variabile \$risultato_insert) non manifesta esito negativo (!) allora tutti i dati inseriti nel form saranno processati ed inseriti nei campi del database. In caso contrario non si procederà all'inserimento dei dati. Al termine del controllo la connessione al database verrà chiusa (**Mysql_close**) e apparirà a video un messaggio di corretto inserimento delle istruzioni.

Possiamo adesso effettuare le nostre prove:

Puntiamo con il nostro browser preferito sul file insert.html ed inseriamo un nome utente e una password corrispondente. Se tutto è corretto e il sistema non manifesta errori, apparirà un messaggio di Ok “i tuoi dati sono stati inseriti correttamente”. E a questo punto all'interno della tabella utenti del database mysql verrà inserito un nuovo record.

Per verificare entriamo nella tabella utenti in PhpMyAdmin del pannello di controllo di Altervista:

The screenshot shows a table titled 'utenti' with columns 'IdUtente', 'User', and 'Password'. The table contains six rows of data. The last row, which corresponds to the new record inserted via the form, has a green background and is highlighted. The data in the last row is: IdUtente 7, User Daniele, Password Venditti. The table also includes standard MySQL management icons (pencil, delete) for each row. Above the table, there are filtering and sorting options. Below the table, there are buttons for selecting all or deselecting all rows, and additional filtering options.

	<input type="checkbox"/>			IdUtente	User	Password
	<input type="checkbox"/>			1	mario	rossi
	<input type="checkbox"/>			2	carlo	banchi
	<input type="checkbox"/>			3	mauro	vitale
	<input type="checkbox"/>			4	paola	grossi
	<input type="checkbox"/>			7	Daniele	Venditti

Figura 19: il dato inserito attraverso il form insert.html corrispondente al record con IdUtente 7

LEZIONE 6

\$GET - \$POST – UPDATE – DELETE FROM

```
<form action="insert.php" method="post">
```

Inviamo i dati al file insert.php con il metodo post (file insert.html)

```
$user= $_POST['user'];
```

```
$pass= $_POST['pass'];
```

Abbiamo creato due variabili. Importante è l'istruzione in php **\$_POST** serve per richiamare le istruzioni inserite nei campi: user e pass (file insert.php)

Nei listati della lezione N.5 abbiamo utilizzato il metodo **POST** (in altro riprendiamo le parti del codice) per passare i dati dal form html al file **insert.php**

Un metodo diverso per passare i dati dal form html al file php è l'istruzione: **GET**

Il funzionamento è analogo al metodo POST.

A tale scopo eseguiamo un esercizio creando due nuovi file in HTML:

```
<html>
<head>
<title>Esempio inserimento dati</title>
</head>
<body>
<form action="get.php" method="get">
  Nome: <input name="nome" type="text"/>
  <input type="submit" value="Inserisci">
</form>
</body>
</html>
```

Salviamo il form con il nome **get.html** e trasferiamolo all'interno del server di Altervista.

```
<html>
<head>
<title>Esempio inserimento dati</title>
</head>
<body>
<form action="post.php" method="post">
    Nome: <input name="nome" type="text"/>
    <input type="submit" value="Inserisci">
</form>
</body>
</html>
```

Salviamo il form con il nome **post.html** e trasferiamolo all'interno del server di Altervista.

Adesso procediamo alla creazione dei due file in php. Uno collegato con il form get.html e l'altro con il form post.html

Istruzione GET (file get.php)

```
<?php
//Recupero il valore del campo nome
$nome_utente = $_GET ['nome'];
//mostrare a video i dati inseriti
echo "ciao". $nome_utente;
?>
```

```
$nome_utente = $_GET ['nome'];
```

Questa è l'istruzione in PHP che recupera il valore inserito nel campo nome del modulo get.html attraverso l'istruzione **`$_GET ['nome']`** trasferendo il risultato ad una variabile **`$nome_utente`**

Con l'istruzione:

```
echo "ciao" . $nome_utente;
```

i dati verranno mostrati a video.

Istruzione POST (file post.php)

```
<?php  
//Recupero il valore del campo nome  
$nome_utente = $_POST ['nome'];  
//mostrare a video i dati inseriti  
echo "ciao" . $nome_utente;  
?>
```

```
$nome_utente = $_POST ['nome'];
```

Questa è l'istruzione in PHP che recupera il valore inserito nel campo nome del modulo get.html attraverso l'istruzione **`$_POST ['nome']`** trasferendo il risultato ad una variabile **`$nome_utente`**

Con l'istruzione:

```
echo "ciao" . $nome_utente;
```

i dati verranno mostrati a video.

Naturalmente anche i file get.php e post.php andranno caricati nel server.

Proviamo il nostro lavoro puntando il nostro browser sul file get.html ed inseriamo un nome a piacere:

The screenshot shows a web browser window with the URL `www.amokve.altervista.org/get.html`. At the top, there is a navigation bar with back, forward, and refresh buttons. Below the URL is the Altervista logo and search fields. A form is displayed with a text input field containing "Nome: Mario" and a red-bordered "Inserisci" button. A cursor arrow is pointing at the "Inserisci" button.

Figura 20: inseriamo un nome nel modulo get.html



Figura 21: Risultato finale dopo l'inserimento con il metodo GET

Con il metodo **GET** è importante notare la parte finale dell'indirizzo url (parte selezionata nell'immagine 21) **"?nome=Mario"** dove viene mostrato il dato inserito nel campo all'interno dell'indirizzo della pagina web. Per questo motivo molti programmatori (e forse non a torto) ritengono il metodo GET meno sicuro rispetto ad altri metodi di richiamo delle informazioni, come il metodo POST.

Infatti con il metodo **POST** questi problemi (dato mostrato nell'indirizzo url della pagina web) non esistono e l'indirizzo della pagina appare più pulito. Puntiamo il browser sul file post.html ed inseriamo lo stesso nome. Otterremo un risultato uguale ma l'indirizzo url ci appare più sicuro.

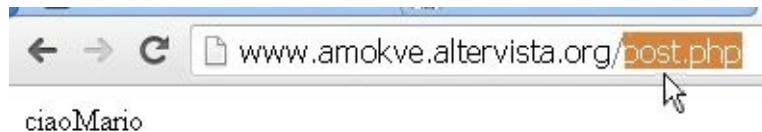


Figura 22: Risultato finale dopo l'inserimento con il metodo POST

Istruzione UPDATE e DELETE FROM

Sono due importanti istruzioni in php che completano la gestione delle informazioni all'interno di un database mysql. Nelle lezioni precedenti abbiamo visto come inserire i dati per mezzo di un form html all'interno della tabella di un database mysql. Adesso invece analizzeremo la modifica dei dati inseriti per mezzo dell'istruzione **UPDATE** o della loro cancellazione tramite **DELETE FROM**.

Iniziamo a descrivere il codice in php che richiama l'istruzione **UPDATE** e dunque ci consente di modificare i dati inseriti nel database mysql.

```
<?php
```

```
||||||||||||||||||||||||||||||||||||||||||||||
```

```
///          CONNESSIONE AL DATABASE
///
```

```
||||||||||||||||||||||||||||||||||||||||||
```

```
//Dichiarazioni variabili per la connessione al database
```

```
$host = 'localhost';
$user = 'amokve';
$password = "";
$database = 'my_amokve';
```

```
// Connessione al database server
```

```
$connessione= mysql_connect($host, $user, $password)
or die ("impossibile connettersi al server");
```

```
// Selezione del database
```

```
mysql_select_db($database)
or die ("impossibile connettersi al database $database");
```

```
||||||||||||||||||||||||||||||||||||||||||
```

```
///          MODIFICO I DATI
///
```

```
||||||||||||||||||||||||||||||||||||||||||
```

```
// script per la modifica dei dati nella tabella
```

```
//
```

```
$dati = mysql_query ("UPDATE utenti SET User='Davide' WHERE IdUtente='7'");
// fine script ?>
```

//Dichiarazioni variabili per la connessione al database

```
$host = 'localhost';  
$user = 'amokve';  
$password = "";  
$database = 'my_amokve'
```

Dichiarazione variabili per la connessione al database di cui abbiamo ampiamente parlato in precedenza. Basta solo ricordare di sostituire i propri dati relativi alla variabile \$user e \$database per la connessione al vostro database mysql.

// Connessione al database server

```
$connessione= mysql_connect($host, $user, $password)  
or die ("impossibile connettersi al server");
```

Apriamo la connessione al database server. Codice già incontrato in precedenza e racchiudiamo il valore di questa connessione nella variabile \$connessione.

// Selezione del database

```
mysql_select_db($database)  
or die ("impossibile connettersi al database $database");
```

Connessione al database specifico “nel nostro caso my_amokve”

// script per la modifica dei dati nella tabella

```
$dati = mysql_query ("UPDATE utenti SET User='Davide' WHERE  
IdUtente='5');
```

```
// fine script ?>
```

Questa è la parte di codice fondamentale che serve per modificare i dati all’interno della tabella del database. Tutta la query è racchiusa nella variabile \$dati dove l’istruzione UPDATE modifica i dati nella tabella “utenti” in corrispondenza del record IdUtente=’7’ la sostituzione del dato User esistente con quello proposto nella query (cioè Davide).

Per capire meglio il concetto torniamo nella tabella all’interno di PhpMyAdmin:

	IdUtente	User	Password
<input type="checkbox"/>	1	mario	rossi
<input type="checkbox"/>	2	carlo	bianchi
<input type="checkbox"/>	3	mauro	vitale
<input type="checkbox"/>	4	paola	grossi
<input type="checkbox"/>	7	Daniele	Venditti

← T → Selezione tutti / Deseleziona tutti Se selezionati

Figura 23: Dati all'interno della tabella “utenti”

Ricordiamo che il campo IdUtente è stato impostato come chiave primaria che si incrementa in automatico. Questo significa che la numerazione è attribuita dal sistema e non inserita manualmente da noi come per gli altri dati. L'aspetto importante della chiave primaria numerica è che non ammette duplicati, conteggia sempre in avanti, per cui se elimino alcuni record della tabella, non verranno più riproposti. Ed in effetti nel nostro caso i record 5 e 6 sono stati eliminati e l'inserimento del record successivo inizia con IdUtente=7.

Procediamo con il nostro esperimento:

Salviamo il file precedente con il nome **modifica.php** e trasferiamo l'oggetto nel server di Altervista.

Premettiamo che questa esercitazione sarà raffinata nelle lezioni successive. Qui invece ci interessa solo capire il funzionamento dell'istruzione UPDATE.

Puntiamo il browser sul file appena caricato all'interno dello spazio hosting e premiamo invio. Non visualizzeremo niente all'interno della pagina web, anche se è possibile inserire un'istruzione “echo” per mostrare un messaggio di avvenuta elaborazione, il file modifica.php eseguirà la query UPDATE, modificando il dato “User” in corrispondenza del valore IdUtente=7.

Tornando nella tabella utenti noteremo in corrispondenza di IdUtenti=7:

User=Davide e Password=Venditti

DELETE FROM

Passiamo adesso ad illustrare l'istruzione **DELETE FROM** che serve per eliminare i dati contenuti nella tabella del database.

Il codice è praticamente uguale a quello già scritto per l'istruzione UPDATE. Cambia solo la parte finale:

<?php

```
||||||| CONNESSIONE AL DATABASE |||||
||||||| //Dichiarazioni variabili per la connessione al database
||||||| $host = 'localhost';
||||||| $user = 'amokve';
||||||| $password = "";
||||||| $database = 'my_amokve';
||||||| // Connessione al database server
||||||| $connessione= mysql_connect($host, $user, $password)
||||||| or die ("impossibile connettersi al server");
||||||| // Selezione del database
||||||| mysql_select_db($database)
||||||| or die ("impossibile connettersi al database $database");
||||||| ELIMINO I DATI
||||||| // script eliminazione dei dati nella tabella
||||||| //
||||||| $dati = mysql_query ("DELETE FROM utenti WHERE IdUtente='7'");
```

```
// script eliminazione dei dati nella tabella  
$dati = mysql_query ("DELETE FROM utenti WHERE IdUtente='7'");  
// fine script ?>  
Eliminiamo dalla tabella utenti il record corrispondente a IdUtenti=7
```

Salviamo il file con il nome **delete.php**, puntiamo il browser sul file (trasferito nel server), premiamo invio e in questo modo elimineremo il dato relativo al record IdUtente=7.

Entriamo nella tabella di PhpMyAdmin per verificare la cancellazione del record.

LEZIONE 7

Esercitazione: Lettura dati Tabella, modifica, cancellazione e inserimento dati

In questa lezione metteremo in pratica gli elementi appresi in precedenza creando

Un’interfaccia web adatta a leggere, inserire, modificare, cancellare i dati contenuti in una tabella del database.

Prima è utile introdurre una nuova funzione che in php ci semplifica il nostro lavoro. L’istruzione:

```
include ('connect-db.php');
```

Con l’istruzione ‘**include**’ riusciamo a semplificare le istruzioni di connessione al database mysql. Infatti tutte le istruzioni di connessione al database utilizzate nei codici precedenti possono essere facilmente racchiuse in un file nome.php che poi verrà richiamato (se occorre) tramite ‘include’ all’inizio di ogni listato.

Creiamo il file **connect-db.php** (il nome è a nostra discrezione)

```
<?php
```

```
//Dichiarazioni variabili per la connessione al database
```

```
$host = 'localhost';  
$user = 'amokve';  
$password = '';  
$database = 'my_amokve';
```

```
// Connessione al database server
```

```
$connessione= mysql_connect($host, $user, $password) or die ("impossibile  
connettersi al server");
```

```
// Selezione del database
```

```
mysql_select_db($database) or die ("impossibile connettersi al database  
$database");
```

```
?>
```

Il listato successivo serve per leggere e dunque visualizzare i dati contenuti nella tabella del database. Abbiamo già incontrato questa caratteristica con il file leggitutto.php, adesso però i risultati verranno presentati in modo più ordinato in un nuovo file che chiameremo **visualizza.php**

```
<?php
```

```
/*
```

```
VIEW.PHP
```

```
Visualizza tutti i dati della tabella ‘utenti’
```

```

*/
```

```

// connessione al database
include('connect-db.php');

// ottiene i risultati dal database
$result = mysql_query("SELECT * FROM utenti")
or die(mysql_error());

echo "<table border='1' cellpadding='10'>";
echo "<tr> <th>IdUtente</th> <th>User</th> <th>Password</th>
<th>Modifica</th> <th>Cancella</th></tr>";

// loop tra i risultati della query del database, visualizzandoli in tabella
while($row = mysql_fetch_array( $result )) {

// emissione del contenuto di ogni riga in una tabella
echo "<tr>";
echo '<td>' . $row['IdUtente'] . '</td>';
echo '<td>' . $row['User'] . '</td>';
echo '<td>' . $row['Password'] . '</td>';
echo '<td><a href="edit.php?id=' . $row['IdUtente'] . '">Modifica</a></td>';
echo '<td><a href="delete.php?id=' . $row['IdUtente'] . '">Cancella</a></td>';
echo "</tr>";
}

// chiude la tabella>
echo "</table>";

?>
```

Commentiamo il codice appena inserito:

```
// connessione al database
```

```
include('connect-db.php');
```

L'istruzione "include" di cui abbiamo parlato all'inizio della lezione che racchiude la gestione della connessione al database mysql

```
// ottiene i risultati dal database
```

```
$result = mysql_query("SELECT * FROM utenti")
```

```
or die(mysql_error());
```

Creiamo una query per selezionare tutte le informazioni contenute nella tabella utenti e racchiuse nella variabile \$result. Ricordiamo "*" che serve per selezionare tutti i campi della tabella. La parte finale serve per gestire il mancato funzionamento della query.

```
echo "<table border='1' cellpadding='10'>";
```

```
echo "<tr> <th>IdUtente</th> <th>User</th> <th>Password</th>  
<th>Modifica</th> <th>Cancella</th></tr>";
```

```
// loop tra i risultati della query del database, visualizzandoli in tabella
```

```
while($row = mysql_fetch_array( $result )) {
```

```
// emissione del contenuto di ogni riga in una tabella
```

```
echo "<tr>";
```

```
echo '<td>' . $row['IdUtente'] . '</td>';
```

```
echo '<td>' . $row['User'] . '</td>';
```

```
echo '<td>' . $row['Password'] . '</td>';
```

```
echo '<td><a href="edit.php?id=' . $row['IdUtente'] . '">Modifica</a></td>';
```

```
echo '<td><a href="delete.php?id=' . $row['IdUtente'] . '">Cancella</a></td>';
```

```
echo "</tr>";
```

```
}
```

```
// chiude la tabella>
```

```
echo "</table>";
```

Creazione della tabella che verrà mostrata nella pagina web con i dati della query precedente.

Il ciclo **While** serve per leggere i dati della tabella del database attraverso l'istruzione **mysql_fetch_array** racchiudendo il tutto in una variabile **\$row**.

Le istruzioni successive stampano a video il risultato dei record della tabella del database riprodotte in una tabella html.

A questo punto la nostra interfaccia web per la gestione del database è pronta, ci occorre solo creare i file **edit.php**, **delete.php** (indicati come link nel listato della tabella) e in seguito il file **insert.php** per l'inserimento delle informazioni. Si tratterà comunque di apportare modifiche più dettagliate ai file che abbiamo illustrato nelle lezioni precedenti.

Se puntiamo il browser sul file **visualizza.php** tutti i dati della tabella verranno mostrati come nella figura successiva in una pagina web.



The screenshot shows a web browser window with the URL www.amokve.altervista.org/visualizza.php. The page displays a table with four rows of data. The columns are labeled: IdUtente, User, Password, Modifica, and Cancella. The data is as follows:

IdUtente	User	Password	Modifica	Cancella
1	mario	rossi	Modifica	Cancella
2	carlo	bianchi	Modifica	Cancella
3	mauro	vitale	Modifica	Cancella
4	paola	grossi	Modifica	Cancella

Figura 24: Dati all'interno della tabella “utenti” mostrati nella pagina web

Creazione del file edit.php

Per modificare i dati nel database occorre creare due file:

edit.php

edit2.php

di seguito il codice per il file **edit.php**

```
<?php  
// CONNESSIONE AL DATABASE  
include('connect-db.php');  
$id=$_GET['id']; // RECUPERA VARIABILE ID DA PAGINA PRECEDENTE
```

```

$rs = mysql_query("SELECT * FROM utenti WHERE IdUtente='".$id""); // QUERY
SQL

$row = mysql_fetch_array($rs); // DEFINISCE VARIABILE $row

?>
<?php

// CREA CODICE HTML

?>

<p><strong>ID:</strong> <?php echo $id; ?></p>
<form method="post" action="edit2.php">
<input type="hidden" name="id" value="<?php echo $row['IdUtente']?>">
User:<br>
<input type="text" name="user" size=20 value="<?php echo $row['User']?>"/>
<br><br>
Password:<br>
<input type="text" name="password" size=20 value="<?php echo
$row['Password']?>"/>
<br><br>
<input type="submit">
</form>

```

Commentiamo il codice

// CONNESSIONE AL DATABASE

include('connect-db.php');

Gestione della connessione al database

\$id=\$_GET['id']; // RECUPERA VARIABILE ID DA PAGINA PRECEDENTE

Dichiariamo la variabile \$id, mentre con l'istruzione \$_GET recuperiamo il contenuto della variabile **id** del file visualizza.php ed in particolare della riga di codice:

echo '<td>Modifica</td>;

che è uguale al numero **IdUtente** nella riga della tabella.

\$rs = mysql_query("SELECT * FROM utenti WHERE IdUtente='".\$id""); //

QUERY SQL

Costruzione della query racchiusa nella variabile \$rs, selezioniamo dalla tabella utenti tutti i campi dove IdUtente è uguale alla variabile \$id (Recuperata dall'istruzione \$_GET)

```
$row = mysql_fetch_array($rs); // DEFINISCE VARIABILE $row
```

Andiamo a leggere i record all'interno della tabella utenti passando i dati al file in PHP.

A questo punto creiamo un codice in html dove verranno richiamati tutti i dati corrispondenti all'ID da modificare:

```
<p><strong>ID:</strong> <?php echo $id; ?></p>
```

Viene visualizzato il numero ID che andiamo a modificare.

```
<form method="post" action="edit2.php">
<input type="hidden" name="id" value="<?php echo $row['IdUtente']?>">
User:<br>
<input type="text" name="user" size=20 value="<?php echo $row['User']?>">
<br><br>
Password:<br>
<input type="text" name="password" size=20 value="<?php echo
$row['Password']?>">
<br><br>
<input type="submit">
</form>
```

Modulo che ci consente di modificare i dati corrispondenti all'IdUtente desiderato.

di seguito il codice per il file **edit2.php**

```
<?php
```

```
// CONNESSIONE AL DATABASE
include('connect-db.php');

// RECUPERA DATI DAL FORM E AGGIORNA
$id=$_POST['id'];
$user=$_POST['user'];
$password = $_POST['password'];

mysql_query ("UPDATE utenti SET User='".$user' , Password='".$password' WHERE IdUtente='".$id."')
or die(mysql_error());
mysql_close();

// una volta salvato, si viene reindirizzati alla pagina di visualizzazione

header("Location: visualizza.php");
?>
```

// CONNESSIONE AL DATABASE

include('connect-db.php');

Stabiliamo la connessione con il database

// RECUPERA DATI DAL FORM E AGGIORNA

\$id=\$_POST['id'];

\$user=\$_POST['user'];

\$password = \$_POST['password'];

Recuperiamo tutti i dati del form precedente (compreso in edit.php)

mysql_query ("UPDATE utenti SET User='".\$user' , Password='".\$password' WHERE IdUtente='".\$id."')

or die(mysql_error());

mysql_close();

Costruzione della query dove modifichiamo (UPDATE) tutti i campi della tabella corrispondenti a User e Password dove IdUtente è uguale a quello contenuto nella

variabile \$id

mysql_close();

chiudiamo la connessione al database

header("Location: visualizza.php");

Istruzione che ci indirizza in automatico al file visualizza.php

Dopo aver trasferito tutti i file all'interno del server di Altervista possiamo eseguire le nostre prove.

Apriamo nuovamente con il nostro Browser il file visualizza.php e proviamo a cliccare sul link modifica corrispondente al primo record (Mario Rossi):

IdUtente	User	Password	Modifica	Cancella
1	mario	rossi	Modifica	Cancella
2	carlo	bianchi	Modifica	Cancella
3	mauro	vitale	Modifica	Cancella
4	paola	grossi	Modifica	Cancella

Figura 25: Clicchiamo su modifica dell' IdUtente 1

ID: 1

User: mario

Password: rossi

Invia

Figura 26: Richiamiamo tutti i dati corrispondenti a IdUtente 1

Nella figura 26 entra in gioco il file edit.php. Proviamo a modificare le informazioni e a cliccare su “invia”, il tutto verrà passato al file edit2.php che eseguirà la query update di aggiornamento delle informazioni.

Proviamo a tornare sul file visualizza.php e noteremo i dati corrispondenti al primo record modificati.

Cancellazione dei dati (**delete.php**)

```
<?php  
// CONNESSIONE AL DATABASE  
include('connect-db.php');  
  
$id=$_GET['id'];  
  
mysql_query ("DELETE FROM utenti WHERE IdUtente='$id'")  
or die(mysql_error());  
mysql_close();  
// una volta salvato, si viene reindirizzati alla pagina di visualizzazione  
  
header("Location: visualizza.php");  
?>
```

Commentiamo il codice:

```
include('connect-db.php');
```

Gestione della connessione al database mysql

```
$id=$_GET['id'];
```

Richiamiamo in questo modo le informazioni corrispondenti al record che ci interessa cancellare

```
mysql_query (“DELETE FROM utenti WHERE IdUtente=’$id’”)  
or die(mysql_error());  
mysql_close();
```

Query di cancellazione (con gestione degli errori e chiusura della connessione al database) del record corrispondente all’Id richiamato. Si legge:

Cancella dalla tabella utenti il dato IdUtente (ricordiamo che è un numero impostato come chiave primaria) dove il numero IdUtente è uguale al record scelto.

Puntiamo ancora il nostro browser al file **visualizza.php** e cliccando su cancella in corrispondenza di un determinato record, si attiverà il file **delete.php** che provvederà ad eliminare il record corrispondente all’informazione scelta.

Inserimento dei dati (**insert.html** e **insert.php** già descritti in precedenza)

Per effettuare questa operazione che ci consentirà di immettere nuovi dati all’interno della tabella del database tramite la nostra comoda interfaccia web, dobbiamo apportare prima una modifica nella parte finale del codice del file **visualizza.php** come mostrato di seguito:

```
<?php  
/*  
VIEW.PHP  
Visualizza tutti i dati della tabella ‘utenti’  
*/  
  
// connessione al database  
include(‘connect-db.php’);  
  
// ottiene i risultati dal database  
$result = mysql_query(“SELECT * FROM utenti”)  
or die(mysql_error());  
  
echo “<table border=‘1’ cellpadding=‘10’>”;  
echo “<tr> <th>IdUtente</th> <th>User</th> <th>Password</th>  
<th>Modifica</th> <th>Cancella</th></tr>”;
```

```

// loop tra i risultati della query del database, visualizzandoli in tabella
while($row = mysql_fetch_array( $result )) {

// emissione del contenuto di ogni riga in una tabella
echo "<tr>";
echo '<td>' . $row['IdUtente'] . '</td>';
echo '<td>' . $row['User'] . '</td>';
echo '<td>' . $row['Password'] . '</td>';
echo '<td><a href='edit.php?id=' . $row['IdUtente'] . '">Modifica</a></td>';
echo '<td><a href='delete.php?id=' . $row['IdUtente'] . '">Cancella</a></td>';
echo "</tr>";
}

// chiude la tabella>
echo "</table>";

?>

```

<p>Aggiungi un nuovo record</p>

<p>Aggiungi un nuovo record</p>

Un link di richiamo al form insert.html che ci occorrerà per inserire i dati nel database attraverso il file insert.php

Ricordiamo i due file già descritti:

insert.html

```

<html>
<head>
<title>Esempio inserimento dati</title>
</head>
<body>
<form action="insert.php" method="post">
Username: <input name="user" type="text"/>
Password: <input name="pass" type="text"/>

```

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

Insert.php

```
<html>  
<head>  
<title>inserisci dati</title>  
</head>  
<body>  
<?php  
  
// CONNESSIONE AL DATABASE  
// Dichiarazioni variabili per la connessione al database  
$host = 'localhost';  
$user = 'amokve';  
$password = '';  
$database = 'my_amokve';  
// Connessione al database server  
$connessione= mysql_connect($host, $user, $password)  
or die ("impossibile connettersi al server");  
// Selezione del database  
mysql_select_db($database)  
or die ("impossibile connettersi al database $database");  
// INSERISCO I NUOVI DATI  
  
// Creo le variabili e carico i dati inviati dallla pagina insert.html  
$user= $_POST['user'];  
$pass= $_POST['pass'];  
  
// preparo la query di inserimento dei dati
```

```

//INSERT INTO "nome_della_tabella" ("nome colonna 1", "nome_colonna2", ...)
//VALUES ("valore 1", "valore 2", ...)

$query_insert = "INSERT INTO utenti (User, Password) VALUES ('$user',
'$pass')";

// lancio la query

$risultato_insert = mysql_query($query_insert);

// controllo l'esito

if (!$risultato_insert) {
    die("Errore nella query $query_insert: " . mysql_error());
}

// chiudo la connessione a MySQL
mysql_close();

?>

<p>I tuoi dati sono stati inseriti correttamente</p>
<p><a href="visualizza.php">Torna alla tabella</a></p>

</body>
</html>

```

Salviamo e proviamo di nuovo l'esercitazione.



The screenshot shows a web browser window with the URL www.amokve.altervista.org/visualizza.php. The page displays a table with the following data:

IdUtente	User	Password	Modifica	Cancella
1	mario2	rossi2	Modifica	Cancella
2	carlo2	bianchi2	Modifica	Cancella
3	mauro	vitale	Modifica	Cancella

[Aggiungi un nuovo record](#)

Figura 27: Risultato finale

LEZIONE 8

Session – Realizzare una semplice area riservata con PHP e MySQL

In questa lezione torneremo alle origini della prima parte del libro, quando abbiamo realizzato come esempio iniziale, una semplicissima area riservata gestita con un file php.

Adesso invece sviluppiamo l'esercitazione con un collegamento al database MySQL. Riutilizzeremo a tale scopo il codice già incontrato nelle precedenti lezioni con le opportune modifiche del caso:

connect-db.php (per la gestione della connessione al database)

insert.html (Form di inserimento dei dati)

insert.php (file per inviare i dati all'interno della tabella utenti)

Il nostro scopo è permettere agli utenti esistenti nella tabella utenti di entrare all'interno di un'area riservata, essere riconosciuti e quindi visualizzare delle pagine protette nel nostro server.

Questo nuovo approccio ci introduce a un nuovo concetto che non è stato fino a qui trattato: **Le Sessioni in php**

Riprendiamo il file connect-db.php precedente ed aggiungiamo una nuova riga di codice:

```
<?php

// avvio la sessione
session_start();

//Dichiarazioni variabili per la connessione al database
$host = 'localhost';
$user = 'amokve';
$password = "";
$database = 'my_amokve';

// Connessione al database server
$connessione= mysql_connect($host, $user, $password) or die ("impossibile
connettersi al server");

// Selezione del database
mysql_select_db($database) or die ("impossibile connettersi al database
$database");

?>
```

session_start();

Come si evince dal listato abbiamo aggiunto una nuova istruzione in PHP. Questo codice indica l'apertura di una sessione.

Le sessioni sono importanti per la gestione degli utenti che entrano all'interno di un'area riservata. **Stabilisce un momento di dialogo iniziale tra l'utente che è iscritto all'interno di una tabella del database e il server. L'utente sarà riconosciuto e gli verrà attribuito un riconoscimento valido per tutta la durata della sua permanenza nell'area riservata.** E' dunque un elemento di sicurezza ed esclude che pagine protette possano essere accessibili ad utenti non registrati nella tabella del database.

La sessione è stata strategicamente inserita nel file di connessione al database poiché tutti gli altri file in php che useremo richiamano **connect-db.php** tramite l'istruzione **include** o **require** (istruzione analoga che non abbiamo menzionato).

In basso presentiamo il codice riguardante il file **login.php**, molto importante per la nostra esercitazione.

```
<?php

// Includo la connessione al database
```

```
require('connect-db.php');

// Se il modulo viene inviato...
if(isset($_POST['login']))
{
    // Dati Inviati dal modulo
    $user = $_POST['user'];
    $pass = $_POST['pass'];

    // Controllo l'utente esiste
    $query = mysql_query("SELECT IdUtente FROM utenti WHERE User = '$user' AND Password = '$pass' LIMIT 1");

    // Se ha trovato un record
    if(mysql_num_rows($query) == 1)
    {
        // prelevo l'id dal database
        $login = mysql_fetch_array($query);

        // Creo una variabile di sessione
        $_SESSION['login'] = $login['IdUtente'];

        // reindirizzo l'utente
        header('Location: privata.php');
        exit;
    }

    // se non esiste da l'errore
    else
        die('Nome Utente o Password errati');
}
```

```

}

?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Login</title>
</head>

<body>
<form action="" method="post">
<input name="user" type="text" id="user" value="Nome Utente" /><br />
<input name="pass" type="password" id="pass" value="Password" /><br />
<input name="login" type="submit" value="Login" /><br />
</form>
</body>
</html>

```

Commentiamo il codice precedente:

```

// Includo la connessione al database
require('connect-db.php');

```

Gestione della connessione al database. Qui abbiamo utilizzato l'**istruzione require** invece di **include**. Svolge la stessa funzione.

```

// Se il modulo viene inviato...
if(isset($_POST['login']))
{
    // Dati Inviati dal modulo
    $user = $_POST['user'];
    $pass = $_POST['pass'];
}

```

Con **if(isset(\$_POST['login']))** ci accertiamo che tutti i dati del **modulo login** vengono inviati e che non sono nulli (questo è il significato dell'istruzione isset)

Attenzione perché in questo caso il modulo login è stato inglobato direttamente nel file login.php e il codice si trova nella parte inferiore. Non esiste dunque in questa esercitazione un file autonomo per eseguire il login.

// Dati Inviati dal modulo

```
$user = $_POST['user'];
$pass = $_POST['pass'];
```

Se questi dati non sono nulli vengono recuperati tramite l'istruzione **\$_POST** e racchiusi in delle variabili (**\$user** e **\$pass**).

// Controllo l'utente esiste

```
$query = mysql_query("SELECT IdUtente FROM utenti WHERE User =
'$user' AND Password = '$pass' LIMIT 1");
```

Il sistema a questo punto esegue una query per accertarsi che l'utente sia presente nella tabella de database. La query seleziona (**SELECT**) il campo **IdUtente** dalla tabella **utenti** dove il campo **user** è uguale alla variabile **\$user** (prelevata dal form). Idem per la Password. Con l'istruzione **LIMIT 1** limitiamo la query ad un record della tabella.

// Se ha trovato un record

```
if(mysql_num_rows($query) == 1)
```

Se **un** (==1) record esiste, viene estratto attraverso l'istruzione **mysql_num_rows**

// prelevo l'id dal database

```
$login = mysql_fetch_array($query);
```

Preleviamo **Id** dal database e attraverso la funzione **mysql_fetch_array** viene ricreata per il php la stessa struttura del database mysql. Il risultato sarà racchiuso nella variabile **\$login**.

// Creo una variabile di sessione

```
$_SESSION['login'] = $login['IdUtente'];
```

Viene creata una variabile di sessione legata all'IdUtente. L'utente è a questo punto riconosciuto e potrà essere tracciato per tutta la sua permanenza nell'area riservata.

```
// reindirizzo l'utente
```

```
header('Location: privata.php');  
exit;  
}
```

Se tutte le condizioni sono verificate l'utente (riconosciuto perché presente nella tabella del database) sarà reindirizzato ad una pagina **privata.php**

```
// se non esiste da l'errore
```

```
else  
die('Nome Utente o Password errati');
```

In caso contrario apparirà un messaggio di errore.

Descriviamo adesso il listato della pagina “**privata.php**” a cui l'utente viene indirizzato se riconosciuto dal sistema in fase di login.

```
<?php
```

```
// Includo la connessione al database
```

```
require('connect-db.php');
```

```
// Se non è stata definita la variabile manda l'utente alla homepage
```

```
if(!isset($_SESSION['login']))
```

```
{
```

```
header('Location: index.php');
```

```
exit;
```

```
}
```

```
?>
```

```

<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN”
“http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>
<html xmlns=“http://www.w3.org/1999/xhtml”>
<head>
<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8” />
<title>Area privata</title>
</head>

<body>
Pagina privata!<br />
<br />
<br />
<?php
// recupera i dati della sessione

echo “IdUtente = “ . $_SESSION[“login”];
?>
<p><a href=“logout.php”>Logout</a><br /></p>
</body>
</html>

```

Commentiamo il codice:

// Includo la connessione al database

require(‘connect-db.php’);

Connessione al database che apre anche la sessione utente.

```
if(!isset($_SESSION[‘login’]))
```

```
{
```

```
    header(‘Location: index.php’);
```

```
    exit;
```

```
}
```

Se la sessione di login non è nulla (!isset) allora si viene reindirizzati alla pagina **index.php**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
<title>Area privata</title>  
</head>  
  
<body>  
Pagina privata!<br />  
<br />  
<br />  
<?php  
// recupera i dati della sessione  
  
echo "IdUtente = ". $_SESSION["login"];  
?>  
<p><a href="logout.php">Logout</a><br /></p>  
</body>  
</html>
```

Questo è un codice html (ricordiamo che è possibile inserire codice html in un file php) dove abbiamo inserito un'istruzione che mostra IdUtente a video.

Di seguito il codice del file **index.php**

```

<?php
// Includo la connessione al database
require('connect-db.php');
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Area riservata</title>
</head>

<body>

```

Menu:

```

<br />
<br />
<?php
if(isset($_SESSION['login']))
    echo '<a href="privata.php">Area privata</a><br /><a
href="logout.php">Logout</a><br />';
else
    echo '<a href="insert.html">Aggiungi Utente</a><br /><a
href="login.php">Login</a><br />';
?>
</body>
</html>

```

Questa pagina contiene un menu che può aiutare l'utente a gestire la connessione. È dunque una pagina di supporto.

In questo codice viene richiamato un altro file **logout.php** di seguito il codice:

```

<?php
// Includo la connessione al database
require('connect-db.php');

// Esegue il logout cancellando la sessione
session_destroy();
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Logout</title>
</head>

<body>
Logout eseguito<br />
<a href="index.php">Vai all'index</a><br />
</body>
</html>

```

session_destroy();

è importante questa istruzione perché chiude la sessione dell'utente quando esce dall'area riservata

Proviamo il risultato della nostra esercitazione “Area Riservata”

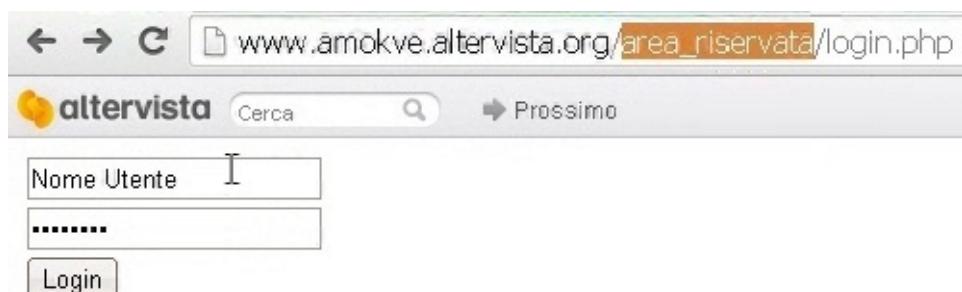


Figura 28: Puntiamo sul file **login.php** – come si nota dall’indirizzo url della pagina tutti i

file dell'esercitazione descritti in questa lezione sono stati inseriti in una cartella “**area_riservata**” che è stata trasferita nel server

Se proviamo ad inserire un Nome Utente e una Password non comprese nel database il sistema genererà un risultato di errore.



Figura 29: Abbiamo inserito un Nome Utente e Password compresi nella tabella del database. Il sistema riconosce l'utente (es. mario rossi) e lo indirizza nella pagina riservata “**privata.php**”

Se clicchiamo su Logout verremo reindirizzati nella pagina index.php dove l'utente troverà un menu per gestire l'area riservata.



Figura 30: il menu della pagina **index.php** – possiamo considerarla anche come pagina di partenza nella gestione della nostra area riservata.

Gentile Utente, Ti ringraziamo per aver effettuato il Download del libro elettronico dedicato alla programmazione PHP+MySql

E' possibile utilizzare il Coupon seguente per ottenere uno sconto del
– 25% su tutti i prodotti che troverai nel portale di:

www.corsi-online.it

Scegli il tuo corso o la tua guida preferita (video o ebook) e segui le informazioni riportate successivamente:

- 1) Scegli il tuo prodotto preferito e procedi all'acquisto
- 2) Aggiungi il prodotto al carrello e vai al carrello
- 3) Procedi ad inserire il coupon sottostante e clicca "**Applica codice promozionale**".
- 4) Effettua il pagamento con PayPal.

apriweb