2. Costruire un database con Apache, PHP e MySql

Lo scopo di questa esercitazione è realizzare un server web e costruire un database accessibile attraverso Internet utilizzando il server **Apache**, il linguaggio **PHP** e il DBMS (*Database Management System*, sistema di gestione di un database) **MySQL**. Ci avvarremo inoltre di programmi gratuiti (*freeware*) e scaricabili dalla rete che faciliteranno l'installazione del software richiesto, la sua configurazione e la sua gestione. L'esercitazione prevede l'utilizzo di macchine con sistema operativo Windows.

Come iniziare: installazione del software necessario

- **Apache**: nome dato alla piattaforma server web più diffusa, che realizza le funzioni di trasporto delle informazioni e di collegamento tra computer connessi a una rete.
- **PHP**: è un linguaggio concepito per la realizzazione di **pagine web dinamiche**. Attualmente è utilizzato principalmente per sviluppare applicazioni web lato server. Il suo nome è un acronimo ricorsivo di *PHP: Hypertext Preprocessor*.
- **MySQL**: è un sistema software progettato per consentire la creazione e manipolazione efficiente di database relazionali solitamente da parte di più utenti.
- **phpMyAdmin**: interfaccia grafica che permette di gestire facilmente il database MySQL.

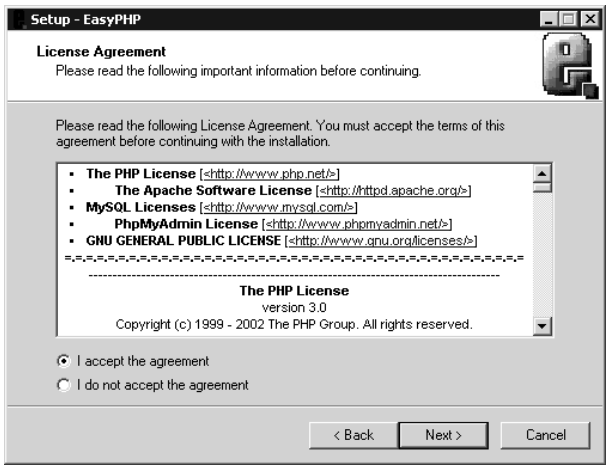
→ Tutti i programmi descritti saranno installati e configurati automaticamente attraverso il software **EasyPHP**

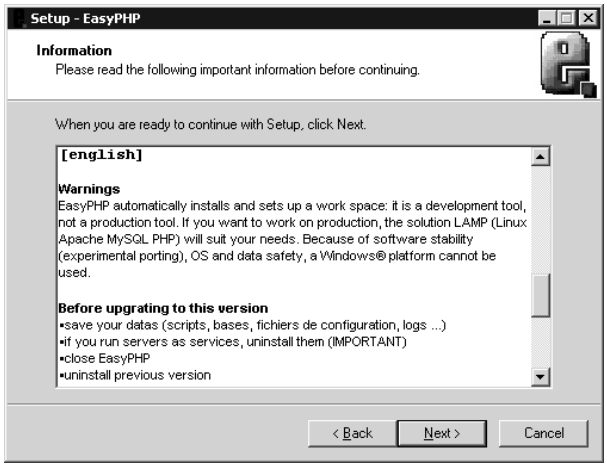
- **Koala-edit** (opzionale): software utilizzato per la creazione di pagine HTML

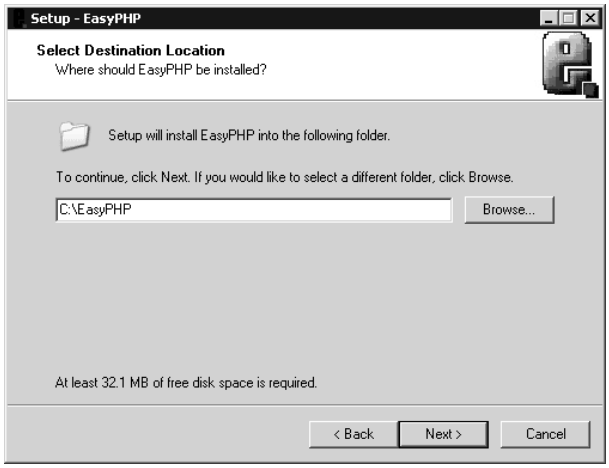
Le versioni dei programmi sono la 1.8 per EasyPHP (consigliata) e 6.2 per Koala-edit (opzionale). EasyPHP è scaricabile dal sito www.easyphp.org, mentre Koala-edit è scaricabile dal sito www.hixus.com.

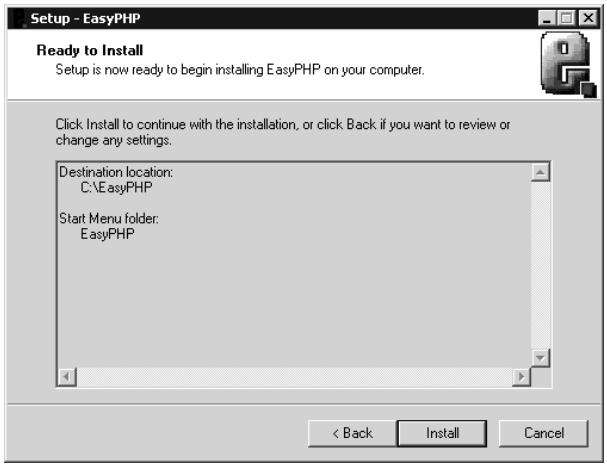
I programmi utilizzati sono completamente gratuiti!

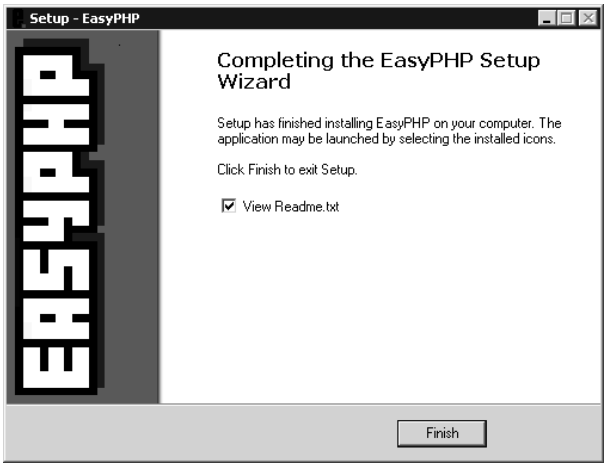
Cliccando sul file di installazione di EasyPHP vi appariranno, nell'ordine, le seguenti schermate. Cliccate semplicemente sul tasto *Next* fino alla fine dell'installazione:







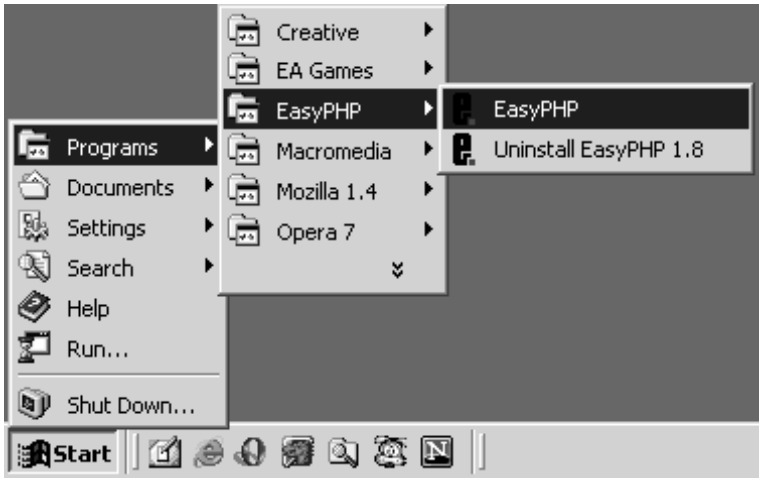




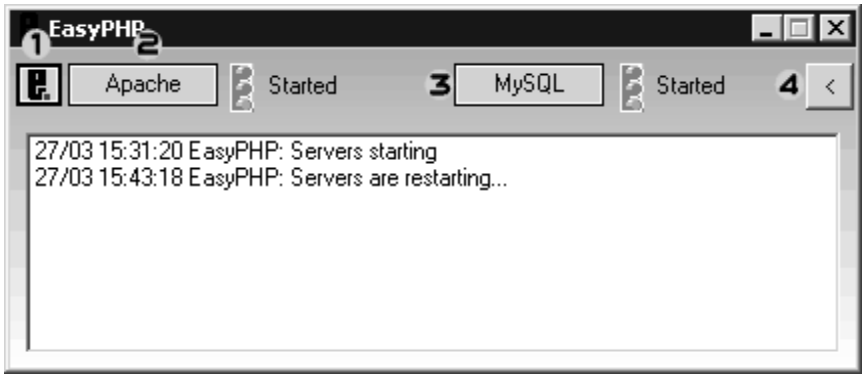
La procedura per l'installazione di Koala Edit è simile.

Avvio di EasyPHP

Per avviare EasyPHP è sufficiente accedere al menù Start->Programmi->EasyPHP:



Comparirà una finestra che vi informerà che due dei programmi gestiti da EasyPHP sono avviati:



I menù di EasyPHP, Apache e MySQL, sono accessibili da questa finestra (1, 2 e 3). Il menù di EasyPHP è accessibile anche attraverso l'icona sulla barra delle applicazioni:

La finestra riporta anche un resoconto (*log*) delle operazioni compiute dal programma. È possibile nascondere il *log* attraverso la freccia sulla destra (4).
Nel momento in cui EasyPHP è avviato il server è pronto. Per motivi di sicurezza, il web server sarà inizialmente raggiungibile esclusivamente in locale (cioè solo dal calcolatore su cui state oeprando). L'indirizzo IP locale di qualsiasi macchina di solito è 127.0.0.1. Potete verificarlo aprendo una finestra del browser e digitando nella barra degli indirizzi *http://127.0.0.1* oppure *http://localhost* (localhost = macchina locale).

Da questo momento potete provare il vostro sito e le vostre pagine web in locale. Per rendere il vostro sito accessibile al mondo, dovete disporre di un IP statico (da richiedere al vostro *Internet provider*) e modificare la seguente linea del file di configurazione di Apache **httpd.conf** (questo file si troverà in C:\Programmi\EasyPHP1-8\conf_files\httpd.conf se avete installato EasyPHP nella cartella C:\Programmi):

```
Listen 127.0.0.1:80
```

Questo comando dice al web server Apache di accettare le connessioni *solo* dal localhost e di mettersi in ascolto dietro alla porta 80. Per ignorarlo, lo si commenta aggiungendo all'inizio il simbolo aserisco:

```
#Listen 127.0.0.1:80
```

A questo punto chiunque, conoscendo il nome del vostro sito (fornito sempre dal provider di servizi Internet) o il vostro IP potrà visualizzare le vostre pagine web. In questa esercitazione confideremo il sito non accessibile dall'esterno.

Primi passi con PHP e MySQL

Siamo ora pronti a visualizzare la nostra prima, semplicissima, pagina web!
Apriamo ora Koala Edit (il blocco note di Windows va ugualmente bene) e digitiamo in un file vuoto (se si fa uso di Koala Edit: in modalità *code editing* se è presente del codice HTML di default lo potete cancellare):

```
<html>
<body>
<?php

    echo "Hello World!!!";

?>
</body>
</html>
```

Salviamo il file appena creato nominandolo *saluto.php* in C:\Programmi\EasyPHP1-8\www\ (questa è la cartella in cui saranno contenute le nostre pagine web).
Apriamo poi una finestra del browser e digitiamo nella barra degli indirizzi:

http://localhost/saluto.php

(si noti che http://localhost/ viene tradotto da Apache in C:\Programmi\EasyPHP1-8\www\).
Se tutto è stato eseguito correttamente comparirà il messaggio «Hello World!!!».
Diamo ora qualche spiegazione del funzionamento del sistema. Il nostro sito, http://localhost, cerca i file da visualizzare di default nella cartella C:\Programmi\EasyPHP1-8\www\, per cui quando digitiamo http://localhost/saluto.php nella barra degli indirizzi del browser:

1. Il browser «chiama» localhost, dove Apache è in ascolto per accettare richieste.
2. La pagina richiesta è *saluto.php*; Apache sa che deve cercarla in C:\Programmi\EasyPHP1-8\www.
3. Apache nota che la pagina ha estensione .php, quindi fa eseguire il codice PHP presente nella pagina e restituisce il risultato al nostro browser.

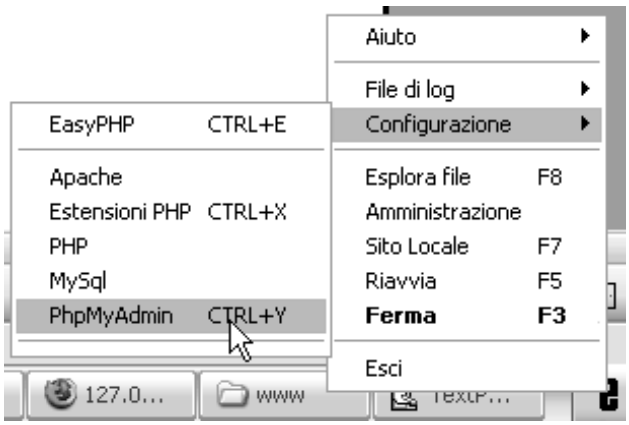
(è possibile cambiare la cartella di default in cui Apache cerca i file web modificando il file di configurazione di Apache, ma ciò esula dagli scopi di questa esercitazione).

Come si può notare, il documento saluto.php è composto da una lista di *tag* HTML, gerarchicamente organizzati come scatole cinesi: il primo *tag* dichiarato, <html>, che indica al browser di visualizzare il contenuto del documento in formato HTML, contiene tutti i restanti *tag* ed è anche l'ultimo ad essere chiuso (</html>; il simbolo "/", detto *slash*, distingue il *tag* di chiusura da quello di apertura). Il documento è poi composto da due sezioni: una sezione introduttiva, detta intestazione (*header*), racchiusa tra i *tag* <HEAD> e </HEAD> (non presente nell'esempio); una sezione contenente il corpo (*body*) effettivo del documento tra i *tag* <BODY> e </BODY>. I tag <?php e ?> delimitano le istruzioni proprie del linguaggio PHP che sono eseguite dall'interprete del linguaggio. La nostra semplice riga in PHP, `echo "Hello World!!!";`, è costituita dal comando `echo` che serve a stampare la sequenza di caratteri racchiusa tra le virgolette; il punto e virgola (;) compare alla fine di qualsiasi istruzione PHP. Provate ad arricchire la vostra prima pagina aggiungendo le seguenti linee:

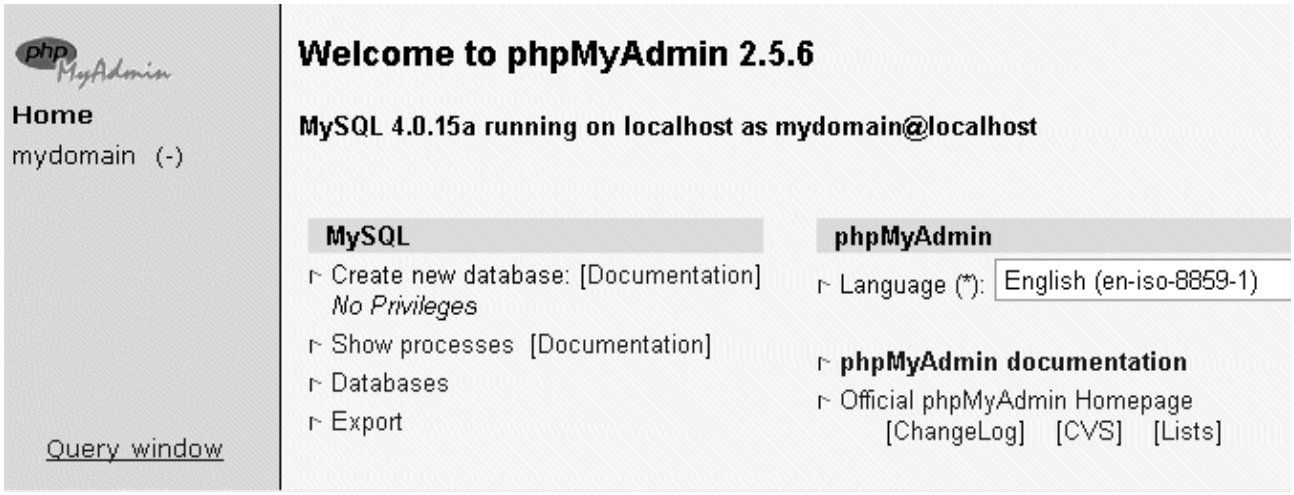
```
<html>
<head>
<title>La mia prima pagina in PHP</title>
</head>
<body>
<?php

    echo "Hello World!!!";
    echo "<br>";
    echo "Oggi è: ";
    echo Date("d F Y");

?>
</body>
</html>
```

Cosa succede quando vi collegate a questa pagina? A cosa serve il comando `echo "<br>";`? E il comando `date`? Perché non si trova tra le virgolette?

Siamo ora pronti a creare e interrogare il nostro primo database con SQL. Utilizzeremo prima di tutto phpMyAdmin per creare il database. Per accedere a phpMyAdmin, cliccate con il tasto destro del mouse sul simbolo di EasyPHP, quindi selezionate Configurazione e di seguito phpMyAdmin:

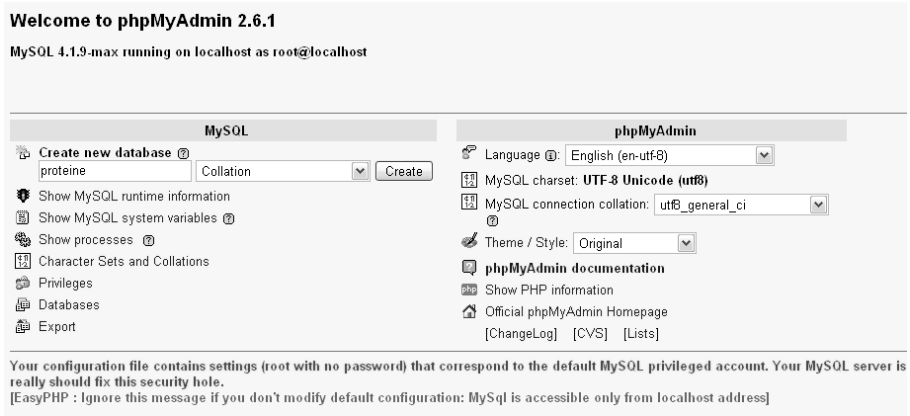


Comparirà una pagina simile alla seguente:

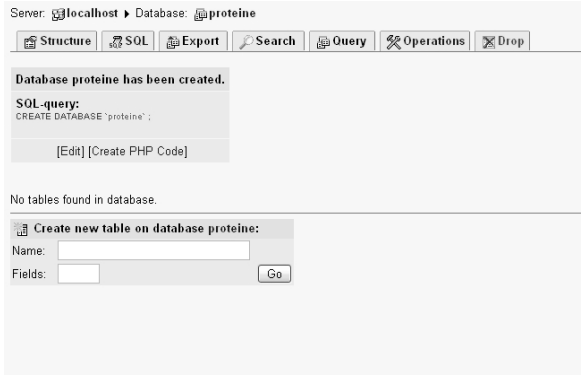


Il database che creeremo può essere di qualsiasi tipo e contenere i nomi, gli indirizzi e i numeri di telefono dei nostri amici, le spese fatte nel corso del mese o l'inventario degli oggetti della nostra stanza. Naturalmente, se si tratta di un database biologico potrà contenere dati di sequenze nucleotidiche, proteiche, strutture e molto altro. L'importante è capire che, indipendentemente dal suo contenuto, la struttura e l'interrogazione di un database SQL è sostanzialmente la stessa.
Come esempio creeremo il database *proteine*, che conterrà i nomi di alcune proteine e le relative sequenze:

Nel campo *Create new database* inserite il nome del database (per esempio «proteine») e cliccate sul tasto *Create*:



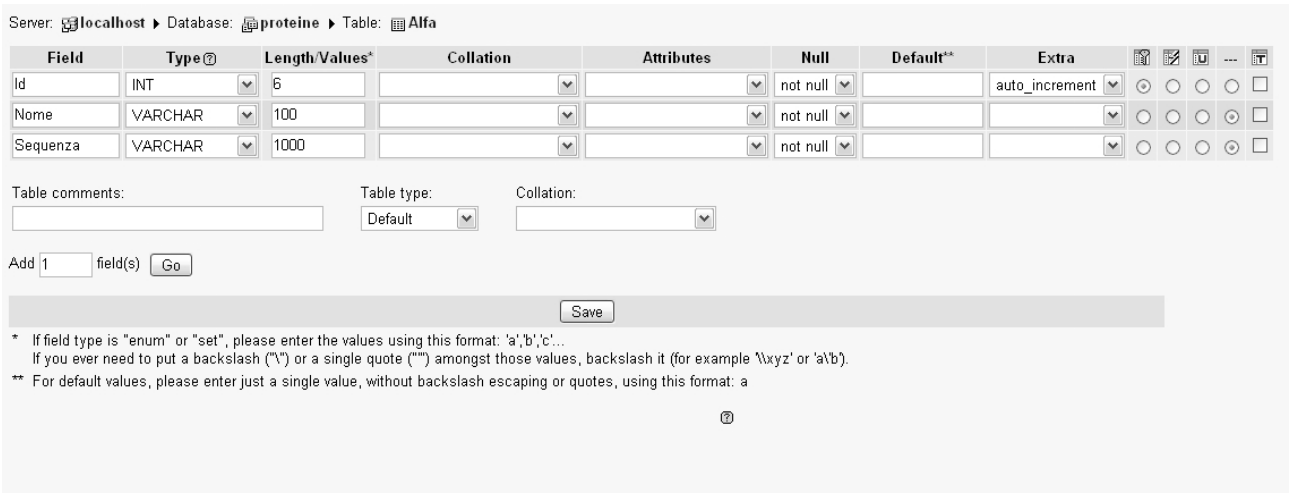
Comparirà una pagina simile alla seguente:



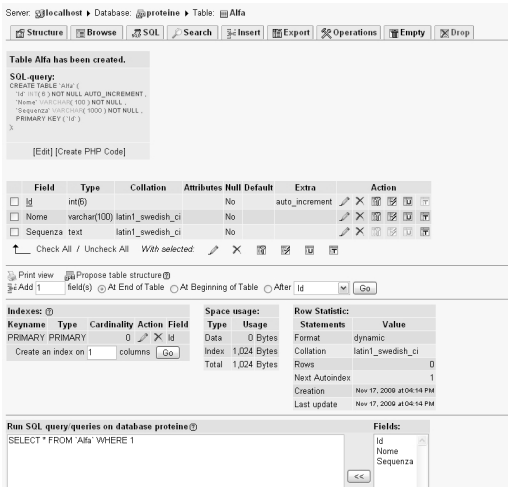
Ogni database può contenere più tabelle. Per esempio, il database proteine può contenere le tabelle Alfa, Alfabeta e Beta, relative a diverse classi proteiche. Ogni tabella è composta inoltre da campi, uno per ogni tipo di informazione contenuta. Nel nostro caso vogliamo avere una tabella di questo tipo (le sequenze che seguono sono, naturalmente, inventate):

Id	**Nome**	**Sequenza**
1	SHMD	ACDEFG
2	AAD	HILMNP
3	GSD	QRSTYZ

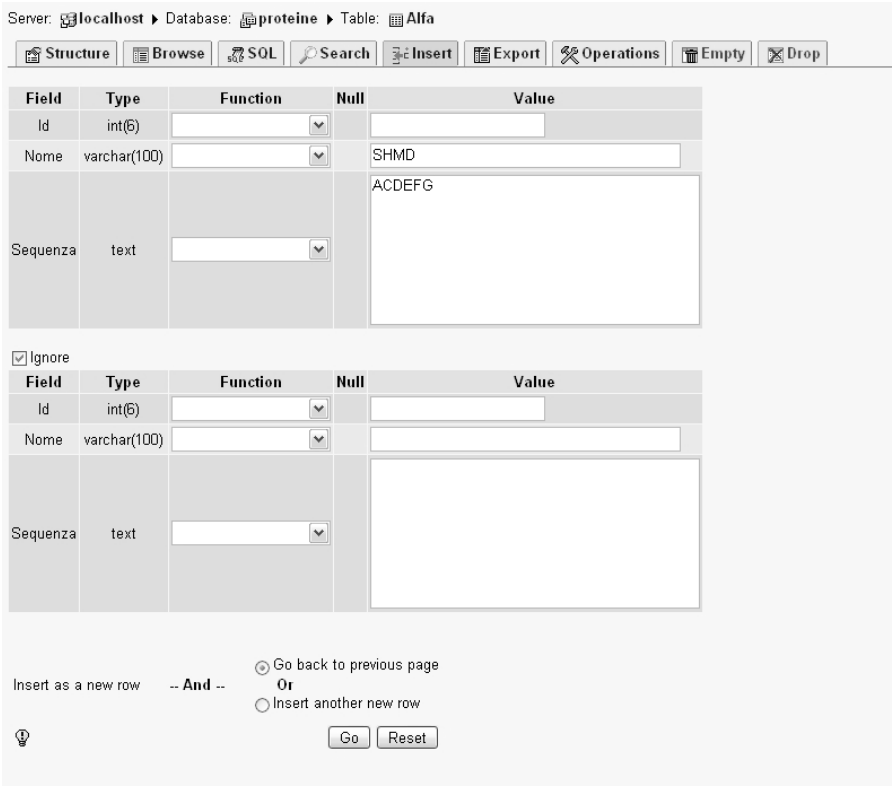
Quindi dovremo costruire una tabella con tre campi: Id, Nome e Sequenza. phpMyAdmin ci avverte che il nostro database è stato creato, ma che non contiene tabelle (*table*). Creiamo quindi una tabella Alfa (*fate attenzione alle lettere maiuscole*: SQL è ***case sensitive***, in altre parole c'è differenza tra le medesime parole con lettere maiuscole e minuscole) con 3 campi (*fiels*), Id, Nome e Sequenza, e premete il tasto *Go*.
Riempite poi i campi nel modo seguente:



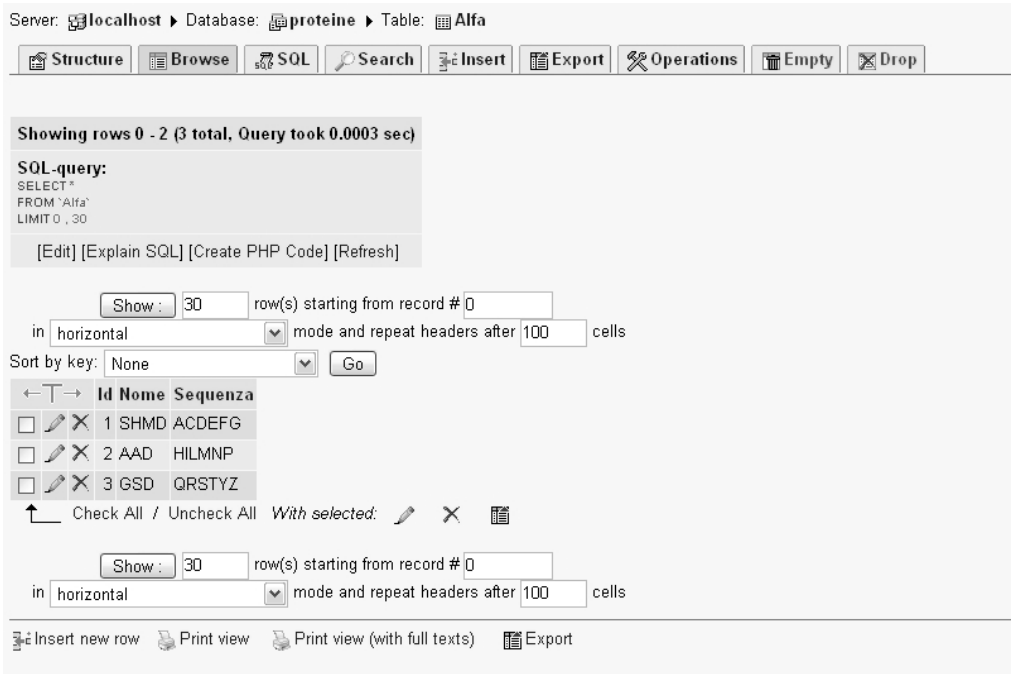
Id, l'identificativo univoco della proteina, è un dato di tipo *INT* (numero intero), mentre gli altri campi contengono stringhe di caratteri (*VARCHAR*). Il campo successivo indica la lunghezza massima di una entry. Selezionando per l'Id la voce *auto_increment*, il valore relativo sarà incrementato ogni volta che aggiungeremo un record (una proteina) alla tabella. Il pulsante *Primary key*, selezionato per il campo Id, assicura l'univocità delle singole entry (non sono ammessi doppioni nel database). A questo punto premete *Save*:



Sulla pagina successiva sarà visualizzato un riassunto della tabella appena creata con la sintassi SQL utilizzata, la possibilità di modificare i vari campi ecc. Per una descrizione più approfondita dei campi consultate il paragrafo «Per saperne di più» (riportato in seguito).
Adesso possiamo inserire i nomi e le sequenze delle nostre proteine cliccando su *Insert*. Inserite i nomi e le sequenze come mostrato nella figura seguente (ripetendo l'inserimento per AAD e GSD) cliccando su *Insert another new row*.



Premete *Go* dopo ogni inserimento. Alla fine cliccate su *Go back* e premete sul tasto *Go*. Cliccando su *Browse* (mostra) potete visualizzare la tabella appena creata:



Interrogazione di un database con PHP

Ora che abbiamo creato il nostro database, lo possiamo interrogare attraverso il nostro sito web grazie a PHP. Riprendiamo la pagina saluto.php creata in precedenza e aggiungiamo le seguenti righe di codice:

```
<html>
<head>
<title>La mia prima pagina in PHP</title>
</head>
<body>
<?php

    echo "Benvenuto nel mio Database!!!";
    echo "<br>";
    echo "Oggi è: ";
    echo Date("F d Y");

?>
<form action="interroga.php" method="POST">
    Nome della proteina:
        <input type="text" name="proteina_alfa">
        <input type="submit" name="submit" value="invia">
</form>
</body>
</html>
```

Provate ora a visualizzare la vostra pagina; cosa succede? Il tag `<form></form>`, come suggerisce il termine stesso, genera un campo all'interno del quale è possibile inserire una stringa di caratteri (per esempio il nome di una proteina). Il tipo di campo generato è definito dall'atributo `type=″text″` del tag `input`. Il nome della nostra proteina sarà salvato come valore di una variabile chiamata `proteina_alfa`. Un secondo tag `input`, questa volta con attributo `submit`, genera un bottone di nome `submit` e sul quale compare la scritta *invia* (`value="invia"`). Premendo il tasto *invia* la variabile `proteina_alfa` e il relativo valore vengono inviati a uno script PHP (nel nostro caso, allo script interroga.php (`action="interroga.php"`) che creeremo tra poco. POST si riferisce a un particolare metodo di invio di dati in rete; l'alternativa è GET (si veda il paragrafo «Per saperne di più» per approfondire l'argomento). Che cosa farà il nostro script PHP con la variabile `proteina_alfa` e il suo valore? Per il momento, nulla! (potete constatarlo voi stessi premendo il tasto invia sulla vostra pagina).
Creiamo quindi un nuovo file di testo (che nominiamo interroga.php) nella cartella C:\Programmi\EasyPHP1-8\www. In questo file dobbiamo aggiungere del codice PHP in modo che la pagina, quando chiamata:

- si connetta al web server (nel nostro caso host e web-server sono la stessa macchina);
- selezioni il database proteine;
- selezioni la tabella Alfa e cerchi nella colonna Nome il nome salvato nella variabile `proteina_alfa`;
- restituisca infine la sequenza della proteina.

Per connetterci al web server, aggiungiamo al nostro script le seguenti righe di codice:

```
<?php
$connessione = mysql_connect("localhost", "root", "");
if ($connessione == 0)
        die ("Connessione non riuscita");
```

```
echo "Connesso al web server! ";
?>
```

Ricordiamo che il codice PHP deve essere compreso tra `<?php` e `?>`. Definiamo poi una variabile `connessione` (in PHP le variabili si definiscono anteponendo al loro nome il simbolo `$`). A questa variabile sarà assegnato come valore il risultato del tentativo di connessione al web server: 1 se connesso, 0 se non connesso. Se il tentativo di connessione non riesce (`if ($connessione == 0)`) il programma stampa la stringa «Connessione non riuscita» e si chiude (`die ("Connessione non riuscita");`). Come potete vedere, il tentativo di connessione viene effettuato con il comando `mysql_connect` che accetta tre parametri: il nome del server web (nel nostro caso localhost); il nome di chi gestisce il database (in easyPHP il valore di default è «root»); una password per accedere al database, nel nostro caso non inserita (si veda il paragrafo «Per saperne di più»). Provate a invocare il nuovo script dal browser.

Per selezionare il database proteine, aggiungiamo al nostro script le seguenti righe:

```
<?php
$connessione = mysql_connect("localhost", "root", "");
if ($connessione == 0)
        die ("Connessione non riuscita");
echo "Connesso al web server!";
mysql_select_db("proteine");
echo "accesso al database di proteine";
?>
```

Come potete vedere, il tentativo di selezione del database viene effettuato con la funzione `mysql_select_db` che accetta come parametro il nome del database.

Per selezionare la sequenza della nostra proteina di tipo Alfa, aggiungiamo al nostro script il seguente codice:

```
<?php
$connessione = mysql_connect("localhost", "root", "");
if ($connessione == 0)
        die ("Connessione non riuscita");
echo "Connesso al web server!";
mysql_select_db("proteine");
echo "accesso al database di proteine";
$proteina_alfa = $_POST['proteina_alfa'];
$query = "SELECT * FROM Alfa WHERE Nome = '$proteina_alfa'";
$sequenza = mysql_query($query) or die ("Query fallita...");
$valore_sequenza=mysql_fetch_array($sequenza);
?>
```

La stringa `$proteina_alfa = $_POST['proteina_alfa'];` definisce la variabile `$proteina_alfa`, alla quale assegna il valore della variabile `proteina_alfa` passata in POST dal nostro primo script (saluto.php). È nella riga precedente che avviene il passaggio del nome della proteina inserita nel campo di ricerca.
La seconda riga definisce la variabile `$query` che permette di interrogare il database; la query si leggerà così:

Seleziona	`SELECT`
tutto	`*`
dalla tabella Alfa	`FROM Alfa`

dove il nome della proteina è uguale al valore della variabile `$proteina_alfa` → `WHERE Nome = '$proteina_alfa'`

La terza riga definisce la variabile `$sequenza` che immagazzina i risultati della richiesta al database, effettuata con l'istruzione `mysql_query($query)`.
Infine l'istruzione `mysql_fetch_array` prende i risultati della richiesta contenuti nella variabile `$sequenza` e li pone in un **array associativo** (`$valore_sequenza`), che possiamo immaginare così composto:

Indici ->	`['Nome']`	`['Sequenza']`
Valori ->	SHMD	ACDEFG

Quindi nell'esempio riportato `$valore_sequenza['Nome']` ha valore SHMD e `$valore_sequenza['Sequenza']` ha valore ACDEFG. Questo array associativo ci servirà per stampare correttamente i risultati.

Per ottenere i risultati tanto attesi aggiungiamo infine:

```
<?php
$connessione = mysql_connect("localhost", "root", "");
if ($connessione == 0)
        die ("Connessione non riuscita");
echo "Connesso al web server!";
mysql_select_db("proteine");
echo "accesso al database di proteine";
$proteina_alfa = $_POST['proteina_alfa'];
$query = "SELECT * FROM Alfa WHERE Nome = '$proteina_alfa'";
$sequenza = mysql_query($query) or die ("Query fallita...");
$valore_sequenza=mysql_fetch_array($sequenza);
echo "<br><h1>La sequenza di ";
echo $valore_sequenza['Nome'];
echo " è :";
echo $valore_sequenza['Sequenza'];
echo "</h1>";
?>
```

Provate ora lo script completo (http://localhost/saluto.php). A cosa serve, secondo voi, il tag <h1> ?

Il sito è ora completo di un sistema per interrogare il database. Si può fare molto per migliorarlo sia da un punto di vista grafico sia funzionale; per esempio, si può stampare a video la scritta «sequenza non trovata» se la lunghezza (*strlen*) della variabile `$valore_sequenza['Sequenza']` è 0 (le parentesi graffe nell'esempio indicano un blocco di codice da eseguire nel costrutto *if-else)*:

```
<?php
$connessione = mysql_connect("localhost", "root", "");
if ($connessione == 0)
        die ("Connessione non riuscita");
echo "Connesso al web server! ";
mysql_select_db("proteine");
echo "accesso al database di proteine";
$proteina_alfa = $_POST['proteina_alfa'];
$query = "SELECT * FROM Alfa WHERE Nome = '$proteina_alfa'";
$sequenza = mysql_query($query) or die ("Query fallita...");
```

```
$valore_sequenza=mysql_fetch_array($sequenza);
if (strlen($valore_sequenza['Sequenza']) == 0)
        {
        echo "<br><h1>";
        echo " Sequenza non trovata! ";
        echo "</h1>";
        die;
        }
else
        {
        echo "<br><h1>La sequenza di ";
        echo $valore_sequenza['Nome'];
        echo " è: ";
        echo $valore_sequenza['Sequenza'];
        echo "</h1>";
        }
?>
```

Per saperne di più

Avete appreso le basi per realizzare un sito web con contenuti dinamici attraverso PHP e MySQL. Per chi volesse approfondire questi argomenti, ecco alcuni utili link:

www.canowhoopass.com/guides/easyphp/running.php
Utile guida per installare e configurare Apache, SQL e PHP con EasyPHP.

www.php-editors.com/articles/sql_phpmyadmin.php
Guida sulla sintassi delle query SQL con PhpMyAdmin.

www.reg.ca/faq/PhpMyAdminTutorial.html
Utile tutorial di PhpMyAdmin.

http://php.html.it/
Ampia documentazione per imparare tutto sul PHP e trovare utili script.

http://database.html.it/guide/leggi/40/guida-linguaggio-sql/
Guida approfondita per diventare esperti di SQL e database relazionali.