

Il mondo dei database

Questo mese vi spieghiamo come potete aggiungere la potenza dei database ai vostri script e creare il codice per mostrare il numero di visitatori!



Nell'ultimo tutorial avete chiuso il vostro calendario di F1 2012 estendendo la funzionalità per permettere a un utente di selezionare una specifica gara da un menu a tendina HTML. Stavolta comincerete con qualcosa di nuovo: i database, che userete per mostrare un contatore di visite aggiornato in tempo reale. Per questo tutorial vi viene richiesto un passo in più, ovvero aggiungere MySQL ai server della vostra macchina. Potete scaricare e installare MySQL per Linux da <http://dev.mysql.com/downloads/mysql>: seguite i link per la versione di Linux che state usando. Se avete qualche dubbio, la documentazione online ha delle ottime linee guida e potete trovarla su <http://dev.mysql.com/doc>. Una volta configurata la piattaforma Linux per servire la pagine PHP con il browser Web (come scritto nel primo tutorial), assicuratevi che il vostro PHP sia configurato con l'opzione **--with-mysql[=DIR]** (controllate la documentazione ufficiale su <http://php.net/manual/en/mysql.installation.php>). Per gestire i vostri database è raccomandabile scaricare e installare **phpMyAdmin**, che offre un'interfaccia Web al server MySQL e vi permette di gestire completamente tutti gli aspetti dei vostri database. Vi consentirà anche di testare e debuggare con più facilità le vostre applicazioni. Lo trovate su www.phpmyadmin.net.

Quindi, cos'è questo MySQL?

MySQL è un sistema di gestione per database relazionali (RDBMS) che, attraverso un server, vi dà accesso ai database (archivi di dati) sfruttando lo Structured Query Language (SQL). MySQL è nato nel 1996, ed è attualmente una delle piattaforme per basi di dati più popolari al mondo. È la M nell'acronimo LAMP (Linux-Apache-MySQL-PHP). È Open Source e viene utilizzato da alcuni grossi nomi del Web, inclusi Facebook, Twitter e Wikipedia. I dati all'interno del database sono archiviati in tabelle. Una tabella è semplicemente una collezione di dati relazionati definiti e consistenti in colonne e righe.

Vi sarà più chiaro con il proseguire del tutorial (oppure leggete la puntata di Hardcore Linux in questo stesso numero). MySQL è disponibile su un grosso ventaglio di sistemi operativi diversi, ed è estremamente veloce.

Pianificare lo script

Il compito prefisso è contare il numero dei visitatori di un sito in tempo reale e mostrarlo nelle pagine. Ogni volta che avete un compito preciso, è buona norma fermarsi un attimo a riflettere quello che succederà. Una volta ottenuto ben chiaro il disegno generale in mente, scrivere il codice sarà 100 volte più facile: è una pratica altamente raccomandata, non c'è nulla di peggio che essere confusi. Non dev'essere nulla di complicato: qualche veloce schemino su un pezzo di carta, con un minimo di logica di flusso è solitamente abbastanza. Detto questo, ecco come sviscerare questo compito:

- un visitatore arriva su una pagina con lo script **whosonline.php** incluso;
- lo script recupera l'indirizzo IP e lo aggiunge (o aggiorna) al database;
- lo script conta quanti IP unici ci sono nel database in una finestra di 5 minuti;
- lo script stampa un intero che rappresenta il valore di cui sopra.

Quanto era facile? Sicuro, diventa più complicato mano a mano che lo script diventa più grande, ma ora che avete tutto scritto potete cominciare.

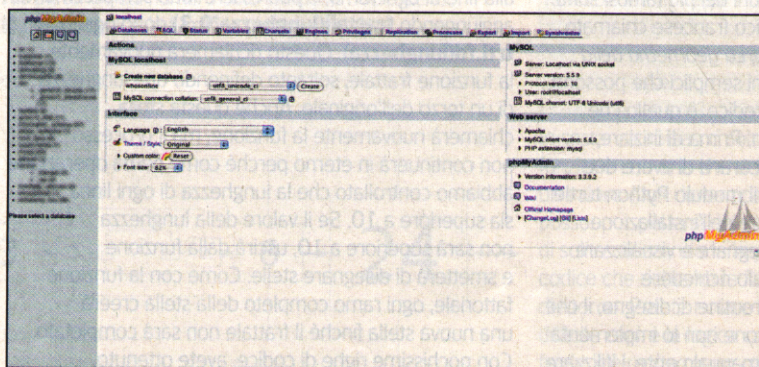
Cominciamo!

La via più facile per ottenere quello di cui avete bisogno è creare uno script PHP che faccia tutto il lavoro sporco e stampi un valore (come scritto sopra): in questo modo potete usare la funzione di PHP **include()** (come in altri tutorial) per inserire il contatore in qualsiasi punto della pagina vogliate. Quindi, aprite il vostro editor di testo preferito e create un nuovo file battezzandolo **whosonline.php**, aggiungendo i tag standard di PHP:

```
<?php
```

```
?>
```

Nei punti pianificati c'è scritto che conterete i visitatori solo all'interno di una finestra di 5 minuti. Potreste preferire un tempo più alto (o più basso), quindi renderete questo valore personalizzabile. Per farlo sfrutterete il tempo attuale in EPOCH per consentirvi un controllo piuttosto accurato. Cos'è EPOCH? EPOCH, detto anche UNIX



Creare un database in phpMyAdmin è un gioco da ragazzi. Tutto quello che vi serve è dargli un nome e premere "Crea"!

time, o anche POSIX time, è semplicemente il numero di secondi passati dalla mezzanotte (UTC) del 1 gennaio 1970. Ad esempio, il valore EPOCH del 1 gennaio 2012 è 1325419200: sono passati 1 miliardo, 325 milioni, 419 mila e 200 secondi dalla mezzanotte del 1 gennaio 1970. Chiaro?

Creare il database

Prima che lo script PHP possa fare alcunché, vi serve un database con cui lavorare. Usando phpMyAdmin create un database chiamato **whosonline**. Potete vedere dallo screenshot che è stato specificato un preciso formato (**utf8_unicode_ci**): non è comunque necessario, dal momento che memorizzerete solo indirizzi IP e timestamp. Una volta creato il database, vi sarà presentata una schermata per dirvi che non ci sono tabelle: create quella necessaria per memorizzare i dati, chiamandola **visitors**, composta da due campi. La terza schermata risulterà confusionaria, ma non vi preoccupate: vi verrà tutto spiegato.

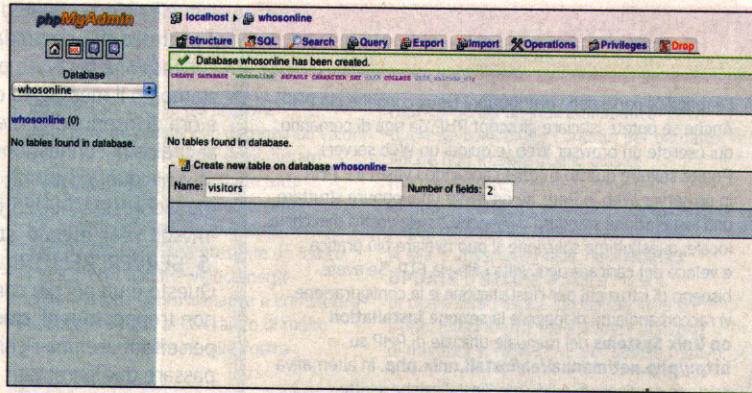
■ **Field** È il nome della colonna che volete. Nella prima riga, indicate **ip_address** e nella seconda **visited**.

■ **Type** Questo indica a MySQL il tipo di dato che userete in questa colonna. MySQL supporta diversi tipi di dati, integer, longtext, binary, float, varchar, date e time. Per questo esercizio, la prima riga per **ip_address** dovrà essere un **VARCHAR**, ovvero una stringa di lunghezza variabile. La seconda sarà **TIMESTAMP**, perché volete memorizzare una rappresentazione di time/date.

■ **Length/Values** Per alcuni tipi di dato, potete limitare la lunghezza del campo. Per il campo **VARCHAR (ip_address)** volete una lunghezza massima di 15 caratteri. Restringere la lunghezza del campo risparmia spazio di archiviazione e velocizza le ricerche. Dal momento che **TIMESTAMP** ha una lunghezza prefissata, potete lasciare vuoto il valore.

■ **Default** MySQL vi permette di specificare un valore di "fallback" usato quando il record inserito nel database non definisce nulla per quel campo. L'indirizzo IP è sempre richiesto ma potete specificare **CURRENT_TIMESTAMP** per **visited**, così da risparmiare il passaggio dell'informazione nell'inserire i visitatori in database.

■ **Index** L'indicizzazione nei database relazionali può essere tremendamente utile. Avete un guadagno sensibile nelle **SELECT** con gli indici al posto giusto. Per ulteriori informazioni, fate riferimento al box relativo. Per il vostro database, scegliete **PRIMARY** per la riga **ip_address**, e **INDEX** per l'altra. Questo è tutto quanto vi serve per strutturare la vostra tabella. Potete lasciar perdere gli altri parametri, non dovete averci a che fare in questo momento. L'unica altra modifica fatta è stata quella di specificare **MyISAM** come **Storage Type** e, ancora, **utf8_**



Il database "visitors" non è molto complicato, quindi vi basterà creare due colonne per memorizzare tutti i dati che vi servono

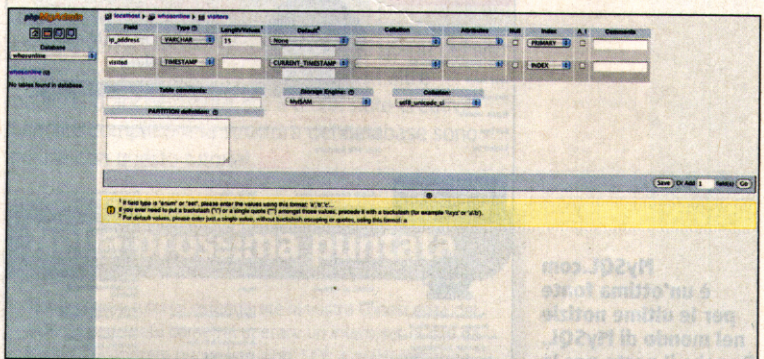
unicode_ci come **Collation** nella struttura generale. Assicuratevi di premere **Salva**, e andate avanti! Ora che il vostro database è sistemato con la sua tabella, siete pronti per lo script. Iniziate inserendo i valori di default: aprite l'editor e inserite il codice seguente appena sotto al tag PHP:

```
$time_limit = 300;

$dbHost = '127.0.0.1';
$dbUsername = 'myuser';
$dbPassword = 'mypassword';
$dbDatabase = 'whosonline';
```

Time limit

Questo codice dovrebbe essere relativamente autoesplicativo. Una cosa da notare è la variabile **\$time_limit**: contiene il valore in secondi entro il quale il visitatore è considerato online (da sopra, 300 secondi sono 5 minuti). Potete cambiarlo per adattarlo alle vostre esigenze. Il valore **\$dbHost** dovrebbe puntare al vostro server MySQL, nella maggior parte dei casi è **127.0.0.1** o **localhost**. **\$dbUsername** e **\$dbPassword** saranno diversi a seconda del vostro setup: dovete inserire qui le informazioni corrette nel vostro database (leggete il box **Utenti MySQL in breve**). **\$dbDatabase** dovrebbe restare lo stesso a meno che non abbiate chiamato il database in modo diverso. Utilizzerete il set di funzioni PHP MySQLi per la connessione. È simile al suo predecessore (MySQL), ma offre metodi procedurali e object-oriented. Per semplicità, comunque, utilizzerete solamente metodi procedurali in questo tutorial. Inizierete con la



Avete molteplici opzioni per le strutture dati in MySQL. Date un'occhiata a tutto quello che vi mette a disposizione per capire quello che si adatta più a voi

Tip



Commentate sempre il vostro codice man mano che lo scrivete. Anche se al momento non vi sembra aver molto senso, lo acquisterà quando tornerete su quello stesso codice qualche mese più tardi. I commenti daranno un senso a tutte quelle complicate funzioni che avete scritto.



Tip



Usate un editor di testo che disponga dell'evidenziazione della sintassi per PHP: vi aiuterà a identificare velocemente nel vostro codice blocchi specifici o funzioni. Esistono molti programmi liberi, quindi date un'occhiata in giro e scegliete ciò che più vi aggrada.

Installare PHP su Linux

La maggior parte delle distribuzioni Linux contiene già PHP. Anche se potete lanciare gli script PHP da riga di comando, qui userete un browser Web (e quindi un Web server). Potete seguire questo tutorial caricando i vostri file PHP in un server Web in Rete, se ne avete uno, oppure sfruttare una installazione standard di Apache2 sulla vostra macchina locale; quest'ultima soluzione si può rivelare più pratica e veloce del caricare ogni volta i file via FTP. Se avete bisogno di istruzioni per l'installazione e la configurazione, vi raccomandiamo di leggere la sezione **Installation on Unix Systems** nel manuale ufficiale di PHP su <http://php.net/manual/en/install.unix.php>. In alternativa esistono centinaia di guide per l'installazione scritte per praticamente qualsiasi distribuzione Linux. Google vi aiuterà nella vostra ricerca se il sito di PHP non basta.

funzione **mysqli_connect()**, e quindi **mysqli_query()** e un truccetto per stampare il numero di righe trovate, **mysqli_num_rows()**.

Connetti, chiedi, mostra!

Ora siete arrivati al nodo del tutorial: lavorare con il database. Con il file whosonline.php già aperto nell'editor, appena sotto la dichiarazione delle variabili aggiungete il codice seguente:

```
$mysqli = mysqli_connect($dbHost,$dbUsername,
$dbPassword,$dbDatabase);
if (mysqli_connect_
errno($mysqli)) echo "Failed
to connect to MySQL: ";
mysqli_connect_error();
```

Ancora una volta, dovrebbe essere sufficientemente semplice: Chiamate **mysqli_connect()** con le credenziali di accesso al database e l'handle della connessione viene assegnato alla variabile **\$mysqli**. Perché questo? Perché le funzioni aggiuntive di database necessitano di una risorsa disponibile per funzionare (vedrete fra poco come); in caso contrario, semplicemente non funzioneranno. È importante notare che questa regola si applica solamente ai metodi procedurali e non

"Per gestire i vostri archivi vi suggeriamo l'uso di phpMyAdmin"

Il ruolo della chiave primaria, appunto, è di identificare univocamente una singola riga, o parte di dati, in una tabella. La parte successiva della

a quelli object-oriented. La riga successiva, contenente il controllo **if()**, controlla velocemente se la connessione ha avuto problemi, nel qual caso viene stampato il messaggio d'errore. Se non ci sono errori, è ora di inserire l'utente nella tabella **visitors**. Sotto al codice di connessione, scrivete:

```
mysqli_query($mysqli, "REPLACE INTO visitors
(ip_address) VALUES ('"
mysqli_real_escape_string($mysqli,
$_SERVER['REMOTE_ADDR']) . "')");
```

Questo è un po' più complicato. La prima parte non troppo: **mysqli_query()** è la funzione usata per effettuare una richiesta al database. Dovete passare due parametri alla funzione, il primo è l'handle e il secondo la query SQL che volete effettuare. La query usata è l'SQL **REPLACE INTO**, estensione di MySQL a SQL standard, che indica a MySQL di inserire o aggiornare un valore nella tabella **visitors**. Potreste agire in modo diverso, e controllare se il valore esiste già, e quindi scrivere un **INSERT** o **UPDATE** di conseguenza, ma questo approccio è più succinto (sebbene, ripetiamo, fuori standard SQL). Una delle attenzioni nell'usare **REPLACE** è che si può fare solamente su valori distinti del database, ma identificando ogni visitatore con il proprio indirizzo IP, questo è perfetto per il vostro script. Per far sì che il database accetti solo valori distinti e permettere l'uso di **REPLACE INTO**, avete dovuto impostare il campo **ip_address** come **PRIMARY KEY** nella struttura.

query specifica su quale tabella volete aggiungere dati (**visitors**) e quali sono tali dati. L'approccio SQL è molto simile alla mappatura **\$chiave => \$valore** degli array. Qui state semplicemente dicendo (**colonna1, colonna2**) **VALUES (valore1, valore2)**, ma nell'inserire effettivamente il dato MySQL lo traduce in **colonna1 => valore1, colonna2 => valore2**. La query inserisce solamente una porzione

MySQL.com
è un'ottima fonte
per le ultime notizie
nel mondo di MySQL.
È anche il posto per la
documentazione
ufficiale



Utenti MySQL in breve

Gli utenti in MySQL funzionano similmente alla vostra piattaforma Linux. Quando installate MySQL per la prima volta, vi viene chiesta una password per l'utente root. Una volta completata l'installazione, potete quindi connettervi come root. Per lo sviluppo locale molti utenti lanciano le applicazioni come root, ma non serve dire che non dovrete mai usare questo utente per connettervi sul Web. È buona norma creare sempre un utente locale con lo stesso nome del database. In questo modo ogni database ha il suo utente ed è limitato a questo, riducendo i danni in caso di compromissione dell'account. Impostare correttamente un utente può proteggervi anche da alti livelli di SQL injection. I permessi che date all'utente dovrebbero essere quelli minimi per completare i propri compiti. Potete creare facilmente gli utenti in phpMyAdmin nella scheda **Privilegi**. Potete assegnare permessi globali o restringerli a determinati database. Pensate bene alle necessità e partite da questo.

di dati, l'indirizzo IP dell'utente, ma per farlo in maniera sicura è stato protetto con una funzione speciale. Nel tutorial precedente avete affrontato alcune basi riguardanti la sicurezza, come la validazione e la pulizia dei dati. Le stesse regole devono applicarsi quando si lavora con i database. Non dovrete solamente proteggere i dati, ma anche assicurarvi che

qualche utente non inserisca del codice malevolo per agire sul database stesso, come cancellare tutte le righe: una pratica conosciuta come *SQL injection*. Per evitare questo, utilizzerete la funzione MySQLi `mysqli_real_escape_string()`. Come per altre funzioni, il primo parametro è sempre l'handle. La funzione prende il secondo parametro e fa l'escape di ogni carattere speciale al suo interno, rendendola utilizzabile nelle query SQL. Questo può essere realmente un salvavita, si raccomanda di usarlo sempre, anche se pensate che l'input sia buono e corretto. Anche se potete essere sicuri al 99% che `$ _SERVER['REMOTE_ADDR']` contenga effettivamente l'indirizzo IP dell'utente, bonificare l'input in questo modo evita a chiunque di poter girare la query a proprio vantaggio. L'array globale `$ _SERVER` contiene informazioni come header, percorsi e posizioni degli script creati dal server Web stesso. Controllate la documentazione di PHP per ulteriori informazioni.

Contare i visitatori unici

Inserito il visitatore, tutto quello che vi resta da fare è contare quelli attivi e ritornare il valore. Sotto la query appena scritta, aggiungete le righe seguenti a `whosonline.php`:

```
$visitors = mysqli_query($mysqli, "SELECT ip_address
FROM visitors WHERE UNIX_TIMESTAMP(NOW()) -
UNIX_TIMESTAMP(visited) <= " .
mysqli_real_escape_string($mysqli,$time_limit) . "");
Ancora una volta usate la funzione mysqli_query()
con l'handle, ma stavolta con una query di tipo
SELECT sulla tabella visitors. La clausola WHERE
serve a specificare un filtro sui dati ritornati:
WHERE UNIX_TIMESTAMP(NOW()) -
UNIX_TIMESTAMP(visited) <= " .
mysqli_real_escape_string($mysqli,$time_limit) . ""
```

Ci sono due cose da notare qui: `UNIX_TIMESTAMP()` e `NOW()`. Siete perdonati per aver pensato che siano funzioni PHP, dal momento che la sintassi è identica, ma si tratta in realtà di funzioni SQL. `NOW()` è una funzione SQL che ritorna il date/time attuale (YYYY-MM-DD HH:MM:SS) e `UNIX_TIMESTAMP()` converte i date/time in formato EPOCH. Quello che state in effetti chiedendo è di ritornare gli indirizzi IP dei visitatori dove il momento della visita sia minore o uguale di 300 secondi fa. Potete ottenere questo prendendo il tempo memorizzato nel database (cioè il momento nel quale è stata aggiunta la visita) e sottraendolo dal tempo attuale. La differenza è il numero di secondi da quando hanno fatto l'ultima

Indicizzazione, questa sconosciuta

Gli indici sono usati per trovare velocemente record con determinati valori. Senza un indice, MySQL deve iniziare dalla prima riga e controllarle tutte finché non trova un valore corretto. Più grande è la tabella, più lento sarà il processo. Se è presente un indice, MySQL può 'saltare' velocemente alla posizione approssimativa e cominciare la ricerca da qui, accelerando di molto le prestazioni. Pensate alla vostra

rubrica: senza un indice alfabetico dovrete controllare contatto per contatto finché non trovate quello giusto, ma con l'indice sapete di andare direttamente alla Z se cercate Massimiliano Zagaglia, evitando tutti i contatti dalla A alla Y. Inserito l'indice, ogni **INSERT**, **UPDATE** o **DELETE** indica a MySQL di ricalcolare l'indice stesso. Se inserite troppi indici in una tabella, ogni richiesta di modifica si rallenterà, per questo motivo.

visita: la vostra finestra temporale. Concatenate la variabile `$time_limit`, come nei tutorial precedenti, e per sicurezza (anche se è controllata da voi) la racchiudete nella `mysqli_real_escape_string()`. Dove è importante la sicurezza, anche se pensate che i dati non cambieranno, è sempre vantaggioso inserire qualche controllo extra per essere sicuri. Meglio che perdere il vostro database! Tutto quello che rimane ora da fare è mostrare il numero di visitatori dalla query. È possibile farlo in due modi: quella seguente è la soluzione più pulita, vedrete l'altra nel prossimo tutorial. Dal momento che volete solo un mero conteggio di righe, e non i dati specifici, potete usare la funzione `mysqli_num_rows()`, che ritorna semplicemente un conteggio delle righe ritornate dalla query. Aggiungete quindi:

```
echo mysqli_num_rows($visitors);
```

Perché questo metodo possa funzionare, dovete passare come parametro una query valida ritornata da `mysqli_query()`. Nella query precedente, avete assegnato il valore della `SELECT` alla variabile `$visitors`, quindi tutto quello che vi resta da fare è passarlo alla funzione per contare le righe. Se voleste il numero di righe per una query diversa, potreste assegnarla a un'altra variabile e passarla come necessario.

Mettere in opera lo script

Fatto questo, lo script è completo. Se aprite il browser e lanciate lo script, dovrete semplicemente veder scritto "1". Se poi analizzate il database con phpMyAdmin, potete vedere l'indirizzo IP e il timestamp registrato nel record. Usare i database può essere veramente facile! Per inserire, come detto all'inizio, il valore all'interno di un sito Web o un template, potete usare la funzione **include** all'interno di un tag HTML:

```
<h4>Visitori online: <?php
include('whosonline.php'); ?></h4>
```

Il valore verrà quindi mostrato a tutti gli utenti del sito. Potete anche scegliere di visualizzarlo in altre maniere, o anche modificare il testo: siate creativi! Il codice completo e la struttura del database sono incluse con questo tutorial. **LXP**

Nella prossima puntata

Nel prossimo tutorial estenderete la vostra conoscenza del SQL e imparerete come recuperare un intero set di dati da MySQL. Scoprirete nuove istruzioni SQL e metterete tutte le nuove conoscenze in un'altra applicazione PHP/MySQL.