MySQL Database Management System

http://www.mysql.com/

DATABASE RELAZIONALI

Un database è una collezione strutturata di informazioni. I database sono delle strutture nelle quali è possibile memorizzare grandi quantità di informazioni. Quello attualmente più diffuso è sicuramente il *modello relazionale* che ci permette di memorizzare i dati all'interno di un'unica grande **tabella** piuttosto che in **tabelle** separate.

campo1	campo2	campo3	campo4	……	…						
1	……..	……..	……..	……..	… …..						
2	…….	……..	……..	……..	… …..						

Queste tabelle possono però avere informazioni incrociate e con un certo legame logico. Questi li possiamo definire attraverso **relazioni** (da cui il nome *database relazionale*).

La struttura della query

I comandi che inviamo al database sono detti **query**. Tutte le query possono essere suddivise in due categorie. La prima contiene tutte quelle che richiedono al database di restituirci determinati dati. Per esempio possiamo richiedere il contenuto di determinate righe di una tabella o il numero di record presenti in un'altra tabella. La seconda categoria comprende le query che richiedono a MySQL di apportare modifiche al database: inserimento di un record, eliminazione di una tabella, modifica di un insieme di record ecc.

CREATE

Innanzi tutto esaminiamo le query che vengono utilizzate per creare database e tabelle. Nel primo caso dobbiano solamente indicare il nome:

```
CREATE DATABASE IF NOT EXISTS db_name
```

- ***db_name*** indica il nome che vogliamo assegnare al database.
- l'opzione facoltativa ***IF NOT EXISTS*** evita che venga visualizzato un errore nel caso esista già.

Per creare una tabella utilizziamo invece il generico comando:

CREATE TABLE IF NOT EXISTS tbl_name (definizioni)

- ***db_name*** indica il nome che vogliamo assegnare allla tabella.
- l'opzione facoltativa ***IF NOT EXISTS*** evita che venga visualizzato un errore nel caso la tabella esista già.
- Le definizioni indicano i tipi di colonne di cui deve essere composta la tabella e la chiave primaria che verrà utilizzata.

CREATE TABLE news (campo1 tipo_campo1 (ampiezza campo1) UNSIGNED not null AUTO_INCREMENT, campo2 tipo_campo2 (ampiezza campo2) not null , campo3 tipo_campo3 not null , campo4 tipo_campo4 (ampiezza campo4), …….,PRIMARY KEY (campo1))

Interazione tra Php e MySql

I linguaggi di programmazione orientati al web publishing come Php, asp o perl sono ottimi per la creazione di pagine dinamiche, ma non offrono nessuna possibilità di memorizzazione dei dati.
La soluzione attuale è quindi quella di utilizzare parallelamente un linguaggio di programmazione e un database.
Da qui il movente di far interagire **Php** con il database relazionale **MySQL**. Questa accoppiata è una delle più diffuse in rete in quanto abbiamo a disposizione gratuitamente un linguaggio solido, capace di sopportare grandi carichi di lavoro, e un database dalle notevoli qualità tecniche.

Connessione

Prima di fare qualunque operazione con MySQL bisogna creare una **connessione** attraverso la funzione:

```
$mydb = mysql_connect ($server, $username, $password);
```

Se si lavora in locale e non si è indicata nessuna password al proprio MySQL, tutti e tre i parametri possono essere omessi. In questo caso i valori di default saranno *localhost:3306* per il server (:3306 indica il numero della porta), username sarà quello del proprietario del database e la password sarà vuota. Visto che generalmente la password viene modificata per prima cosa dal gestore del server , una volta portato on line su uno spazio preso in hosting , risulterà impossibile effettuare una connessione senza indicare almeno gli ultimi due parametri.

Per questo motivo creiamo una pagina che chiameremo *config.inc.php* che contenga:

```
// parametri della connessione al database
   $db_host = “127.0.0.1";
   $db_user = “Anonymous";
   $db_password = "";
   $db_name = "";
```

ed un’altra pagina che chiameremo *open_db_conn.php* che contenga:

```
include("config.inc.php");

$db = mysql_connect($db_host, $db_user,
   $db_password);

if ($db == FALSE)
   die ("Errore …….");
```

Nella maggior parte dei casi di hosting, ci viene assegnato un database dal gestore e non possiamo crearne altri. Dopo la connessione **selezioniamo un database** attraverso la funzione:

```
mysql_select_db ($database_name, $db);
```

Il primo parametro conterrà una stringa con il nome del database che ci è stato assegnato dal gestore. Il secondo parametro indica la connessione attiva e può essere omesso, in quanto Php considera come identificatore di default l'ultimo creato.

```
include(" config.inc.php ");
$db = mysql_connect($db_host, $db_user,
   $db_password);
if ($db == FALSE)
   die ("Errore …….");
mysql_select_db($db_name, $db)
   or die ("Errore ……");
/*In questo caso non abbiamo bisogno di
   memorizzare alcun valore, visto che la funzione
   restituisce solo TRUE o FALSE. */
```

A questo punto introduciamo un'istruzione che crei una tabella con la struttura che ci interessa:

```
$query = "CREATE TABLE my_table
  (campo1 tipo_campo1 (ampiezza campo1)
  UNSIGNED not null
  AUTO_INCREMENT, campo2
  tipo_campo2 (ampiezza campo2) not null ,
  campo3 tipo_campo3 not null , campo4
  tipo_campo4 (ampiezza campo4),
  ……..,PRIMARY KEY (campo1))";
```

Vediamo un esempio:

ID	TITOLO	TESTO	DATA	AUTORE	MAIL
1	Primo articolo	Ecco il primo articolo	4-5-04	freephp	mail@html.it
2	Secondo artocolo	Ecco il secondo articolo	11-5-04	freephp	mail@html.it

```
$query = "CREATE TABLE news (id INT (5) UNSIGNED
  not null AUTO_INCREMENT, titolo VARCHAR (255)
  not null , testo TEXT not null , data INT (11) , autore
  VARCHAR (50) , mail VARCHAR (50) , PRIMARY
  KEY (id))";
```

Una volta definita la query, possiamo comunicarla al database attraverso la funzione **mysql_query**:

```
mysql_query($query, $db);
```

Anche questa funzione restituisce FALSE in caso di errore, generalmente quando la query contiene uno o più errori di sintassi.

```
include("config.inc.php");

$db = mysql_connect($db_host, $db_user, $db_password);

if ($db == FALSE)
   die ("Errore …….");

mysql_select_db($db_name, $db)
   or die ("Errore ……");

$query = "CREATE TABLE news (…….)";

if (mysql_query($query, $db))
   echo "Tabella creata…";
   else
   echo "Errore….";

mysql_close($db);
   ?>
```

DROP

Così come abbiamo creato database e tabelle, li possiamo eliminare:

DROP DATABASE IF EXISTS db_name
DROP TABLE IF EXISTS tbl_name

- massima prudenza nell'usare questi comandi, visto che elminando un database o una tabella si elimina anche tutto il loro contenuto!
- l'opzione facoltativa ***IF NOT EXISTS*** evita che venga visualizzato un errore nel caso esista già.

ALTER

Una volta creata una tabella, possiamo modificarla inserendo, rimuovendo o modificarndo colonne:

ALTER TABLE tbl_name ADD COLUMN definizione

- Con una query di questo tipo aggiungiamo la colonna specificata in "definizione" alla tabella "tbl_name".
- La sintassi di "definizione" è uguale a quella usata per la creazione della tabella.
- In fondo al comando possiamo aggiungere *FIRST* se vogliamo che la colonna inserita sia la prima o *AFTER column_name* per assegnare una posizione diversa.
- Se non specifichiamo niente la colonna verrà inserita in ultima posizione.

Allo stesso modo possiamo eliminare una colonna:

ALTER TABLE tbl_name DROP COLUMN definizione

Possiamo infine modificare il tipo di una colonna:

ALTER TABLE tbl_name MODIFY colonna new_type

Per fare un esempio

- ALTER TABLE my_table ADD COLUMN cognome VARCHAR(20) AFTER id
- ALTER TABLE my_table DROP COLUMN cognome
- ALTER TABLE my_table MODIFY nome VARCHAR(30)

INSERT

Una volta creato un database dovremo popolarlo inserendo i record:

INSERT INTO tbl_name (cols) VALUES (values)

- **INSERT INTO tbl_name** indica che vogliamo inserire un nuovo elemento nella tabella *tbl_name* La parentesi che segue indica in quali colonne vogliamo specificare il valore da inserire.
- Tramite **VALUES** indichiamo che ci apprestiamo a elencare i valori che vanno inseriti nelle colonne specificate in precedenza. Questi valori sono contenuti nella seconda parentesi e disposti nello stesso ordine con cui abbiamo specificato le colonne. Tutti i valori devono essere indicati fra due apici che possono essere omessi solo nel caso di valori numerici.

Per fare un esempio

INSERT INTO news (titolo, testo, data, autore, mail)
VALUES ('primo articolo', 'Ecco il primo articolo',
'1002664800', 'freephp.it', 'mail@html.it');

Vediamo quindi le principali query per gestire il contenuto di un database:

UPDATE

Tramite questa funzione possiamo modificare alcuni valori di determinate righe:

UPDATE tbl_name SET col_name=expr WHERE where_definition

- ***tbl_name*** indica il nome della tabella.
- ***col_name=expr*** indica la modifica da apportare a una determinata colonna .
- ***where_definition*** indica le condizioni che si devono verificare in una riga perché questa possa essere modificata.

Per fare un esempio

- UPDATE my_table SET nome='Mario' WHERE cognome='Rossi'
- UPDATE my_table SET id=id*1000, nome='Mario' WHERE (cognome='Rossi' OR cognome='Bianchi') AND id>5

DELETE

Per eliminare una o più righe ci basta indicare le condizioni che devono verificarsi:

DELETE FROM tbl_name WHERE condizioni

- DELETE FROM my_table WHERE id<1000 AND (nome='Mario' OR nome='Luigi')

Per fare un esempio

- UPDATE my_table SET nome='Mario' WHERE cognome='Rossi'
- UPDATE my_table SET id=id*1000, nome='Mario' WHERE (cognome='Rossi' OR cognome='Bianchi') AND id>5

ESERCIZIO

- creare un modulo per l'inserzione degli articoli in un nuovo file di nome **insertion.php**
- le informazioni che richiediamo sono più o meno quelle che dovranno essere inserite nel database
- il campo *id* non sarà presente, infatti verrà aggiornato automaticamente da MySQL
- richiedere una *password* per evitare che chiunque possa inserire un articolo. Il valore di questo campo verrà confrontato con una password scelta dall'utente da aggiungere al file *config.inc.php*

Dopo di che…….

- verificare che la password sia stata inserita correttamente
- controllare che i dati necessari siano stati inseriti
- verificare che i campi di testo non siano vuoti o non contengano solo spazi
- verificare che le stringhe non contengano caratteri particolari (come l'apice o le virgolette)
- inserire i dati nella tabella

Per creare una tabella utilizziamo invece il generico comando:

CREATE TABLE IF NOT EXISTS tbl_name (definizioni)

- ***db_name*** indica il nome che vogliamo assegnare allla tabella.
- l'opzione facoltativa ***IF NOT EXISTS*** evita che venga visualizzato un errore nel caso la tabella esista già.
- Le definizioni indicano i tipi di colonne di cui deve essere composta la tabella e la chiave primaria che verrà utilizzata.