

PHP – Funzioni per la gestione di database MySQL

| Funzione | Sintassi | Note |
|--|---|--|
| mysqli_connect — Apre una connessione ad un server MySQL | <code>\$link = mysqli_connect("127.0.0.1", "my_user", "my_password", "my_db");</code> <code>\$mysqli = new mysqli('localhost', 'my_user', 'my_password', 'my_db');</code> | Restituisce un identificativo di connessione MySQL in caso di successo oppure FALSE in caso di fallimento. |
| mysqli_query — Invia una query MySQL | <code>mixed mysqli_query (mysqli \$link , string \$query [, int \$resultmode = MYSQLI_STORE_RESULT])</code> <code>\$result = \$mysqli->query(\$sql)</code> | Invia una query al database. Solo per le istruzioni SELECT, SHOW, EXPLAIN o DESCRIBE restituisce un identificativo di risorsa o FALSE se la query non è stata eseguita correttamente. Per altri tipi di istruzioni SQL, restituisce TRUE in caso di successo e FALSE in caso di errore. |
| mysqli_fetch_array — Carica una riga del risultato come un array associativo, un array numerico o entrambi. | <code>mixed mysqli_fetch_array (mysqli_result \$result [, int \$resulttype = MYSQLI_BOTH])</code> <code>\$row = \$result->fetch_array(MYSQLI_NUM);</code> <code>MYSQLI_ASSOC o MYSQLI_BOTH</code> | Restituisce un array che corrisponde alla riga caricata o NULL se non ci sono più righe. |
| mysqli_num_rows — Ottiene il numero di righe in un risultato | <code>int mysqli_num_rows (mysqli_result \$result)</code> <code>\$row_cnt = \$result->num_rows;</code> | Restituisce il numero di righe in un risultato. Questo comando è valido solo per le istruzioni SELECT . |
| mysqli_affected_rows — Ottiene il numero di righe coinvolte nelle precedenti operazioni MySQL | <code>int mysqli_affected_rows (mysqli \$link)</code> <code>\$mysqli->affected_rows</code> | Restituisce il numero di righe coinvolte nell'ultima query INSERT, UPDATE o DELETE . Se l'ultima query fallisce, questa funzione restituisce -1. |
| mysqli_insert_id — Ottiene l'identificativo generato dalla precedente operazione INSERT | <code>mixed mysqli_insert_id (mysqli \$link)</code> <code>\$mysqli->insert_id</code> | restituisce l'identificativo generato per una colonna AUTO_INCREMENT dalla precedente query INSERT . Restituisce 0 se la precedente query non ha generato un valore AUTO_INCREMENT . |
| mysqli_error — Restituisce il testo del messaggio di errore della precedente operazione MySQL | <code>string mysqli_error (mysqli \$link)</code> <code>\$mysqli->error</code> | Restituisce il testo dell'errore dall'ultima funzione MySQL, oppure "" (la stringa vuota) se nessun errore intercorre |
| mysqli_connect_error — Restituisce un errore di connessione a MySQL | <code>string mysqli_connect_error (void)</code> <code>\$mysqli->connect_error</code> | Restituisce l'ultima stringa di errore dopo l'ultima chiamata a <code>mysqli_connect()</code> o NULL se non c'è errore |
| mysqli_real_escape_string — Crea una stringa SQL "legale" | <code>string mysqli_real_escape_string (mysqli \$link , string \$escapestr)</code> | I caratteri codificati sono <i>NUL (ASCII 0), \n, \r, \, ' , " , e Control-Z.</i> |
| mysqli_close — Chiude una connessione MySQL | <code>bool mysqli_close ([resource \$identificativo_connessione])</code> | Restituisce TRUE in caso di successo, FALSE in caso di fallimento. |

Indice dei contenuti

- [`mysqli::\$affected_rows`](#) — Gets the number of affected rows in a previous MySQL operation
- [`mysqli::autocommit`](#) — Turns on or off auto-committing database modifications
- [`mysqli::begin_transaction`](#) — Starts a transaction
- [`mysqli::change_user`](#) — Changes the user of the specified database connection
- [`mysqli::character_set_name`](#) — Returns the default character set for the database connection
- [`mysqli::\$client_info`](#) — Get MySQL client info
- [`mysqli::\$client_version`](#) — Returns the MySQL client version as a string
- [`mysqli::close`](#) — Closes a previously opened database connection
- [`mysqli::commit`](#) — Commits the current transaction
- [`mysqli::\$connect_errno`](#) — Returns the error code from last connect call
- [`mysqli::\$connect_error`](#) — Returns a string description of the last connect error
- [`mysqli::construct`](#) — Open a new connection to the MySQL server
- [`mysqli::debug`](#) — Performs debugging operations
- [`mysqli::dump_debug_info`](#) — Dump debugging information into the log
- [`mysqli::\$errno`](#) — Returns the error code for the most recent function call
- [`mysqli::\$error_list`](#) — Returns a list of errors from the last command executed
- [`mysqli::\$error`](#) — Returns a string description of the last error
- [`mysqli::\$field_count`](#) — Returns the number of columns for the most recent query
- [`mysqli::get_charset`](#) — Returns a character set object
- [`mysqli::get_client_info`](#) — Get MySQL client info
- [`mysqli_get_client_stats`](#) — Returns client per-process statistics
- [`mysqli_get_client_version`](#) — Returns the MySQL client version as an integer
- [`mysqli::get_connection_stats`](#) — Returns statistics about the client connection
- [`mysqli::\$host_info`](#) — Returns a string representing the type of connection used
- [`mysqli::\$protocol_version`](#) — Returns the version of the MySQL protocol used
- [`mysqli::\$server_info`](#) — Returns the version of the MySQL server
- [`mysqli::\$server_version`](#) — Returns the version of the MySQL server as an integer
- [`mysqli::get_warnings`](#) — Get result of SHOW WARNINGS
- [`mysqli::\$info`](#) — Retrieves information about the most recently executed query
- [`mysqli::init`](#) — Initializes MySQLi and returns a resource for use with `mysqli_real_connect()`
- [`mysqli::\$insert_id`](#) — Returns the auto generated id used in the last query
- [`mysqli::kill`](#) — Asks the server to kill a MySQL thread
- [`mysqli::more_results`](#) — Check if there are any more query results from a multi query
- [`mysqli::multi_query`](#) — Performs a query on the database
- [`mysqli::next_result`](#) — Prepare next result from multi_query
- [`mysqli::options`](#) — Set options
- [`mysqli::ping`](#) — Pings a server connection, or tries to reconnect if the connection has gone down
- [`mysqli::poll`](#) — Poll connections
- [`mysqli::prepare`](#) — Prepare an SQL statement for execution
- [`mysqli::query`](#) — Performs a query on the database
- [`mysqli::real_connect`](#) — Opens a connection to a mysql server
- [`mysqli::real_escape_string`](#) — Escapes special characters in a string for use in an SQL statement, taking into account the current charset of the connection
- [`mysqli::real_query`](#) — Execute an SQL query
- [`mysqli::reap_async_query`](#) — Get result from async query
- [`mysqli::refresh`](#) — Refreshes
- [`mysqli::release_savepoint`](#) — Rolls back a transaction to the named savepoint
- [`mysqli::rollback`](#) — Rolls back current transaction
- [`mysqli::rpl_query_type`](#) — Returns RPL query type
- [`mysqli::savepoint`](#) — Set a named transaction savepoint

PHP – Funzioni per la gestione di database MySQL

- [mysqli::select_db](#) — Selects the default database for database queries
- [mysqli::send_query](#) — Send the query and return
- [mysqli::set_charset](#) — Sets the default client character set
- [mysqli::set_local_infile_default](#) — Unsets user defined handler for load local infile command
- [mysqli::set_local_infile_handler](#) — Set callback function for LOAD DATA LOCAL INFILE command
- [mysqli::\\$sqlstate](#) — Returns the SQLSTATE error from previous MySQL operation
- [mysqli::ssl_set](#) — Used for establishing secure connections using SSL
- [mysqli::stat](#) — Gets the current system status
- [mysqli::stmt_init](#) — Initializes a statement and returns an object for use with mysqli_stmt_prepare
- [mysqli::store_result](#) — Transfers a result set from the last query
- [mysqli::\\$thread_id](#) — Returns the thread ID for the current connection
- [mysqli::thread_safe](#) — Returns whether thread safety is given or not
- [mysqli::use_result](#) — Initiate a result set retrieval
- [mysqli::\\$warning_count](#) — Returns the number of warnings from the last query for the given link

Indice dei contenuti

- [mysqli_result::\\$current_field](#) — Get current field offset of a result pointer
- [mysqli_result::data_seek](#) — Adjusts the result pointer to an arbitrary row in the result
- [mysqli_result::fetch_all](#) — Fetches all result rows as an associative array, a numeric array, or both
- [mysqli_result::fetch_array](#) — Fetch a result row as an associative, a numeric array, or both
- [mysqli_result::fetch_assoc](#) — Fetch a result row as an associative array
- [mysqli_result::fetch_field_direct](#) — Fetch meta-data for a single field
- [mysqli_result::fetch_field](#) — Returns the next field in the result set
- [mysqli_result::fetch_fields](#) — Returns an array of objects representing the fields in a result set
- [mysqli_result::fetch_object](#) — Returns the current row of a result set as an object
- [mysqli_result::fetch_row](#) — Get a result row as an enumerated array
- [mysqli_result::\\$field_count](#) — Get the number of fields in a result
- [mysqli_result::field_seek](#) — Set result pointer to a specified field offset
- [mysqli_result::free](#) — Frees the memory associated with a result
- [mysqli_result::\\$lengths](#) — Returns the lengths of the columns of the current row in the result set
- [mysqli_result::\\$num_rows](#) — Gets the number of rows in a result

PHP – Funzioni per la gestione di database MySQL

link: <http://php.net/manual/en/mysqli.examples-basic.php>

```
<?php
// Let's pass in a $_GET variable to our example, in this case
// it's aid for actor_id in our Sakila database. Let's make it
// default to 1, and cast it to an integer as to avoid SQL injection
// and/or related security problems. Handling all of this goes beyond
// the scope of this simple example. Example:
// http://example.org/script.php?aid=42
if (isset($_GET['aid']) && is_numeric($_GET['aid'])) {
    $aid = (int) $_GET['aid'];
} else {
    $aid = 1;
}

// Connecting to and selecting a MySQL database named sakila
// Hostname: 127.0.0.1, username: your_user, password: your_pass, db: sakila
$mysqli = new mysqli('127.0.0.1', 'your_user', 'your_pass', 'sakila');

// Oh no! A connect_errno exists so the connection attempt failed!
if ($mysqli->connect_errno) {
    // The connection failed. What do you want to do?
    // You could contact yourself (email?), log the error, show a nice page, etc.
    // You do not want to reveal sensitive information

    // Let's try this:
    echo "Sorry, this website is experiencing problems.";

    // Something you should not do on a public site, but this example will show
    you
    // anyways, is print out MySQL error related information --
    you might log this
    echo "Error: Failed to make a MySQL connection, here is why: \n";
    echo "Errno: " . $mysqli->connect_errno . "\n";
    echo "Error: " . $mysqli->connect_error . "\n";

    // You might want to show them something nice, but we will simply exit
    exit;
}

// Perform an SQL query
$sql = "SELECT actor_id, first_name, last_name FROM actor WHERE actor_id = $aid";
if (!$result = $mysqli->query($sql)) {
    // Oh no! The query failed.
    echo "Sorry, the website is experiencing problems.";

    // Again, do not do this on a public site, but we'll show you how
    // to get the error information
    echo "Error: Our query failed to execute and here is why: \n";
    echo "Query: " . $sql . "\n";
    echo "Errno: " . $mysqli->errno . "\n";
    echo "Error: " . $mysqli->error . "\n";
    exit;
}

// Phew, we made it. We know our MySQL connection and query
// succeeded, but do we have a result?
if ($result->num_rows === 0) {
    // Oh, no rows! Sometimes that's expected and okay, sometimes
    // it is not. You decide. In this case, maybe actor_id was too
    // large?
    echo "We could not find a match for ID $aid, sorry about that. Please try ag
ain.";
    exit;
}
```

PHP – Funzioni per la gestione di database MySQL

```
}

// Now, we know only one result will exist in this example so let's
// fetch it into an associated array where the array's keys are the
// table's column names
$actor = $result->fetch_assoc();
echo "Sometimes I see " . $actor['first_name'] . " " . $actor['last_name'] . " o
n TV.";

// Now, let's fetch five random actors and output their names to a list.
// We'll add less error handling here as you can do that on your own now
$sql = "SELECT actor_id, first_name, last_name FROM actor ORDER BY rand() LIMIT
5";
if (!$result = $mysqli->query($sql)) {
    echo "Sorry, the website is experiencing problems.";
    exit;
}

// Print our 5 random actors in a list, and link to each actor
echo "<ul>\n";
while ($actor = $result->fetch_assoc()) {
    echo "<li><a href='" . $_SERVER['SCRIPT_FILENAME'] . "?aid=" . $actor['actor
_id'] . "'>\n";
    echo $actor['first_name'] . ' ' . $actor['last_name'];
    echo "</a></li>\n";
}
echo "</ul>\n";

// The script will automatically free the result and close the MySQL
// connection when it exits, but let's just do it anyways
$result->free();
$mysqli->close();
?>
```

Altri Link:

<http://php.net/manual/en/mysqli.quickstart.statements.php>