

Web Development 2



Departamento de Lenguajes y
Sistemas Informáticos



Universitat d'Alacant
Universidad de Alicante

Connecting to MySQL database server

Sergio Luján Mora
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante

Index

- Connecting to a database server
- Connecting to MySQL

Connecting to a database server (1)

- Specific functions for each DBMS:
- Function syntax:
`DBMSname_functionName()`
- Support for more than 20 DBMS:
 - Informix, InterBase
 - Microsoft SQL Server, mSQL, MySQL,
 - Oracle
 - PostgreSQL
 - Sybase
 - ...

Connecting to a database server (2)

- Problem:
 - The use of specific functions makes code not portable
- Solution:
 - Use a middle layer (ODBC)
 - Trade off: less performance
- MySQL integrates easily with PHP

Connecting to MySQL (1)

- Open a new connection:
`mysql_connect(DBMSserver, user, password)`
- Returns a MySQL link identifier on success, or `FALSE` on failure
 - Identifier is used in the following functions
- Close a connection:
`mysql_close(identifier)`

Connecting to MySQL (2)

- Checks whether or not the connection to the server is working:

```
mysql_ping(identifier)
```

- Sets the current active database on the server:

```
mysql_select_db(DBname, identifier)
```

- Every subsequent call to `mysql_query()` will be made on the active database

- Both functions returns `TRUE` on success or `FALSE` on failure

Connecting to MySQL (3)

- Send an SQL query:

```
mysql_query(SQLquery, identifier)
```

- For `SELECT`, `SHOW`, `DESCRIBE`, `EXPLAIN` and other statements returning resultset, `mysql_query()` returns a resource on success, or `FALSE` on error.

- For other type of SQL statements, `UPDATE`, `DELETE`, `DROP`, etc, `mysql_query()` returns `TRUE` on success or `FALSE` on error

Connecting to MySQL (4)

- Returns number of affected rows by the last query:

- INSERT, UPDATE, REPLACE or DELETE query:

```
mysql_affected_rows(identifier)
```

- SELECT or SHOW → Returns the number of rows in the result:

```
mysql_num_rows(result)
```

Connecting to MySQL (5)

- Free result memory:

```
mysql_free_result(result)
```

- All result memory will automatically be freed when the script ends

- `mysql_free_result()` only needs to be called if you are concerned about how much memory is being used for queries that return large result sets

Connecting to MySQL (6)

- Fetch a result row as an associative array, a numeric array, or both:

```
mysql_fetch_array(result [, type])
```

- Returns an array of strings that corresponds to the fetched row, or FALSE if there are no more rows
- Moves the internal data pointer ahead
- type:
 - MYSQL_ASSOC: associative (column name)
 - MYSQL_NUM: numeric (column position)
 - MYSQL_BOTH (default): both

Connecting to MySQL (7)

- You can also use:

- `mysql_fetch_assoc()`, like `mysql_fetch_array()` with `MYSQL_ASSOC`: associative (column name)
- `mysql_fetch_row()`, like `mysql_fetch_array()` with `MYSQL_NUM`: numeric (column position)

- Important:

- An important thing to note is that using `mysql_fetch_array()` is not significantly slower than using `mysql_fetch_row()`, while it provides a significant added value
- Field names returned by this function are case-sensitive.

Connecting to MySQL (8)

- Example:

```
<?
if(!($siden = mysql_connect("127.0.0.1", "user", "password")))
    die("Error: Couldn't connect");

if(!mysql_select_db("bd", $siden))
    die("Error: Database name incorrect");

$sentencia = "SELECT * FROM Students";
$alumnos = mysql_query($sentencia, $siden);
if(!$alumnos)
    die("Error: couldn't execute the query");

while($fila = mysql_fetch_array($alumnos))
{
    echo $fila['Surname'] . ', ' . $fila['Name'] . '<br>';
}
mysql_close($siden);
?>
```

Connecting to MySQL (9)

- Example (a different way of making the same):

```
<?
// Connecting, selecting database
$link = mysql_connect('mysql_host', 'mysql_user', 'mysql_password')
    or die('Could not connect: ' . mysql_error());
echo 'Connected successfully';
mysql_select_db('my_database') or die('Could not select database');

// Performing SQL query
$query = 'SELECT * FROM my_table';
$result = mysql_query($query) or die('Query failed: ' . mysql_error());
```

Connecting to MySQL (10)

- Example:

```
// Printing results in HTML
echo "<table>\n";
while ($line = mysql_fetch_array($result, MYSQL_ASSOC
)) {
    echo "\t<tr>\n";
    foreach ($line as $col_value) {
        echo "\t\t<td>$col_value</td>\n";
    }
    echo "\t</tr>\n";
}
echo "</table>\n";

// Free resultset
mysql_free_result($result);
// Closing connection
mysql_close($link);
?>
```

Connecting to MySQL (11)

- You can query everything you want, not only SELECT, INSERT, UPDATE and DELETE:
 - CREATE TABLE
 - DROP TABLE
 - DROP DATABASE
 - ...

Web Development 2

Connecting to MySQL (12)

```
<?php
// Opens connection
$con = mysql_connect("localhost","user","password");
if (!$con)
    die('Error, couldn't connect: ' . mysql_error());

// Creates a new database
if (mysql_query("CREATE DATABASE my_db",$con))
    echo "Database created";
else
    // New database may already exist
    echo "Error, couldn't create database: " . mysql_error();
```

Web Development 2

Connecting to MySQL (and 13)

```
// Create a new table
if(!mysql_select_db("my_db", $con))
    die('Error, couldn't select database: ' . mysql_error());

$sql = "CREATE TABLE person
(
Name varchar(15),
Surname varchar(15),
Age int
)";
if(!mysql_query($sql, $con))
    die('Error, couldn't create table: ' . mysql_error());

mysql_close($con);
?>
```