

# MYSQL SELECT QUERY

<http://www.tutorialspoint.com/mysql/mysql-select-query.htm>

Copyright © tutorialspoint.com

The SQL **SELECT** command is used to fetch data from MySQL database. You can use this command at mysql> prompt as well as in any script like PHP.

## Syntax:

Here is generic SQL syntax of SELECT command to fetch data from MySQL table:

```
SELECT field1, field2, ...fieldN table_name1, table_name2...  
[WHERE Clause]  
[OFFSET M ][LIMIT N]
```

- You can use one or more tables separated by comma to include various conditions using a WHERE clause, but WHERE clause is an optional part of SELECT command.
- You can fetch one or more fields in a single SELECT command.
- You can specify star \* in place of fields. In this case, SELECT will return all the fields.
- You can specify any condition using WHERE clause.
- You can specify an offset using **OFFSET** from where SELECT will start returning records. By default offset is zero.
- You can limit the number of returns using **LIMIT** attribute.

## Fetching Data from Command Prompt:

This will use SQL SELECT command to fetch data from MySQL table tutorials\_tbl

## Example:

Following example will return all the records from **tutorials\_tbl** table:

```
root@host# mysql -u root -p password;  
Enter password:*****  
mysql> use TUTORIALS;  
Database changed  
mysql> SELECT * from tutorials_tbl  
+-----+-----+-----+-----+  
| tutorial_id | tutorial_title | tutorial_author | submission_date |  
+-----+-----+-----+-----+  
| 1 | Learn PHP | John Poul | 2007-05-21 |  
| 2 | Learn MySQL | Abdul S | 2007-05-21 |  
| 3 | JAVA Tutorial | Sanjay | 2007-05-21 |  
+-----+-----+-----+-----+  
3 rows in set (0.01 sec)  
  
mysql>
```

## Fetching Data Using PHP Script:

You can use same SQL SELECT command into PHP function **mysql\_query**. This function is used to execute SQL command and later another PHP function **mysql\_fetch\_array** can be used to fetch all the selected data. This function returns row as an associative array, a numeric array, or both. This function returns FALSE if there are no more rows.

Below is a simple example to fetch records from **tutorials\_tbl** table.

## Example:

Try out the following example to display all the records from tutorials\_tbl table.

```
<?php
$dbhost = 'localhost:3036';
$dbuser = 'root';
$dbpass = 'rootpassword';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if(! $conn )
{
    die('Could not connect: ' . mysql_error());
}
$sql = 'SELECT tutorial_id, tutorial_title,
        tutorial_author, submission_date
        FROM tutorials_tbl';

mysql_select_db('TUTORIALS');
$retval = mysql_query( $sql, $conn );
if(! $retval )
{
    die('Could not get data: ' . mysql_error());
}
while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
{
    echo "Tutorial ID :{$row['tutorial_id']} <br> ".
        "Title: {$row['tutorial_title']} <br> ".
        "Author: {$row['tutorial_author']} <br> ".
        "Submission Date : {$row['submission_date']} <br> ".
        "-----<br>";
}
echo "Fetched data successfully\n";
mysql_close($conn);
?>
```

The content of the rows are assigned to the variable \$row and the values in row are then printed.

**NOTE:** Always remember to put curly brackets when you want to insert an array value directly into a string.

In above example, the constant **MYSQL\_ASSOC** is used as the second argument to PHP function **mysql\_fetch\_array**, so that it returns the row as an associative array. With an associative array you can access the field by using their name instead of using the index.

PHP provides another function called **mysql\_fetch\_assoc**, which also returns the row as an associative array.

## Example:

Try out the following example to display all the records from tutorial\_tbl table using mysql\_fetch\_assoc function.

```
<?php
$dbhost = 'localhost:3036';
$dbuser = 'root';
$dbpass = 'rootpassword';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if(! $conn )
{
    die('Could not connect: ' . mysql_error());
}
$sql = 'SELECT tutorial_id, tutorial_title,
        tutorial_author, submission_date
        FROM tutorials_tbl';

mysql_select_db('TUTORIALS');
$retval = mysql_query( $sql, $conn );
if(! $retval )
{
    die('Could not get data: ' . mysql_error());
}
```

```

}
while($row = mysql_fetch_assoc($retval))
{
    echo "Tutorial ID :{$row['tutorial_id']} <br> ".
        "Title: {$row['tutorial_title']} <br> ".
        "Author: {$row['tutorial_author']} <br> ".
        "Submission Date : {$row['submission_date']} <br> ".
        "-----<br>";
}
echo "Fetched data successfully\n";
mysql_close($conn);
?>

```

You can also use the constant **MYSQL\_NUM** as the second argument to PHP function `mysql_fetch_array`. This will cause the function to return an array with numeric index.

## Example:

Try out the following example to display all the records from `tutorials_tbl` table using `MYSQL_NUM` argument.

```

<?php
$dbhost = 'localhost:3036';
$dbuser = 'root';
$dbpass = 'rootpassword';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if(! $conn )
{
    die('Could not connect: ' . mysql_error());
}
$sql = 'SELECT tutorial_id, tutorial_title,
        tutorial_author, submission_date
        FROM tutorials_tbl';

mysql_select_db('TUTORIALS');
$retval = mysql_query( $sql, $conn );
if(! $retval )
{
    die('Could not get data: ' . mysql_error());
}
while($row = mysql_fetch_array($retval, MYSQL_NUM))
{
    echo "Tutorial ID :{$row[0]} <br> ".
        "Title: {$row[1]} <br> ".
        "Author: {$row[2]} <br> ".
        "Submission Date : {$row[3]} <br> ".
        "-----<br>";
}
echo "Fetched data successfully\n";
mysql_close($conn);
?>

```

All the above three examples will produce the same result.

## Releasing Memory:

It's a good practice to release cursor memory at the end of each `SELECT` statement. This can be done by using PHP function **`mysql_free_result`**. Below is the example to show how it has to be used.

## Example:

Try out the following example

```

<?php
$dbhost = 'localhost:3036';
$dbuser = 'root';

```

```

$dbpass = 'rootpassword';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if(! $conn )
{
    die('Could not connect: ' . mysql_error());
}
$sql = 'SELECT tutorial_id, tutorial_title,
        tutorial_author, submission_date
        FROM tutorials_tbl';

mysql_select_db('TUTORIALS');
$retval = mysql_query( $sql, $conn );
if(! $retval )
{
    die('Could not get data: ' . mysql_error());
}
while($row = mysql_fetch_array($retval, MYSQL_NUM))
{
    echo "Tutorial ID :{$row[0]} <br> ".
        "Title: {$row[1]} <br> ".
        "Author: {$row[2]} <br> ".
        "Submission Date : {$row[3]} <br> ".
        "-----<br>";
}
mysql_free_result($retval);
echo "Fetched data successfully\n";
mysql_close($conn);
?>

```

While fetching data, you can write as complex SQL as you like. Procedure will remain same as mentioned above

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js