MYSQL Introduction

* Mysql is the most popular database system.
* Mysql is freely available and widely used with PHP scripts to create power full and dynamic server side application.
* It is easy to install on a wide range Os include one windows, mac and unix.
* It simple to use and includes some handling administration tools.
* It is fast ,powerful ,client-server system.
* It is SQL compatible.
* A data in Mysql is stored in database object called tables.
* Table is a collection of related data entries and consist of rows and columns.

# Data Types of MYSQL (Field Type)

* + The following is a list of datatypes available in MySQL, which includes

## String

* + 1. **Numeric**

## Date/time

## String Data Types

* + The following are the **String Data Types** in MySQL:

|  |  |  |
| --- | --- | --- |
| **Data Type Syntax** | **Maximum Size** | **Explanation** |
| CHAR(*size*) | Maximum size of 255 characters. | Where ***size*** is the number of characters to store. Fixed-length strings. Space padded on  right to equal ***size*** characters. |
| VARCHAR(*size*) | Maximum size of 255 characters. | Where ***size*** is the number of characters to store. Variable-length string. |
| TINYTEXT(*size*) | Maximum size of 255 characters. | Where ***size*** is the number of characters to store. |
| TEXT(*size*) | Maximum size of 65,535 characters. | Where ***size*** is the number of characters to store. |
| MEDIUMTEXT(*size*) | Maximum size of 16,777,215 characters. | Where ***size*** is the number of characters to store. |
| LONGTEXT(*size*) | Maximum size of 4GB or 4,294,967,295  characters. | Where ***size*** is the number of characters to store. |
| BINARY(*size*) | Maximum size of 255 characters. | Where ***size*** is the number of binary characters  to store. Fixed-length strings. Space padded on right to equal ***size*** characters |
| VARBINARY(*size*) | Maximum size of 255 characters. | Where ***size*** is the number of characters to store. Variable-length string. |
| BLOB | Maximum size of 65,535 bytes. | Normal sized BLOB. |
| MEDIUMBLOB | Maximum size of  16,777,215 bytes. | Medium sized BLOB. |
| LONGBLOB | Maximum size of 4,294,967,295 bytes. | Long sized BLOB. |

## Numeric Data Types

* + The following are the **Numeric Data types** in MySQL:

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Maximum Size** | **Explanation** |
| BIT | Signed values range from -128 to 127. Unsigned values range from 0 to 255. | Very small integer value that is equivalent to TINYINT(1). |
| TINYINT(*m*) | Signed values range from -128 to 127. Unsigned values range from 0 to 255. | Very small integer value. |
| SMALLINT(*m*) | Signed values range from -32768 to 32767. Unsigned values range from 0 to 65535. | Small integer value. |
| MEDIUMINT(*m*) | Signed values range from -8388608 to 8388607. Unsigned values range from 0 to  16777215. | Medium integer value. |
| INT(*m*) | Signed values range from -2147483648 to 2147483647. Unsigned values range from 0  to 4294967295. | Standard integer value. |
| BIGINT(*m*) | Signed values range from - 9223372036854775808 to  9223372036854775807. Unsigned values  range from 0 to 18446744073709551615. | Big integer value. |
| DECIMAL(*m*,*d*) | Unpacked fixed point number.  ***m*** defaults to 10, if not specified.  ***d*** defaults to 0, if not specified. | Where ***m*** is the total digits  and ***d*** is the number of digits after the decimal. |
| FLOAT(*m*,*d*) | Single precision floating point number. | Where ***m*** is the total digits and ***d*** is the number of digits after the decimal. |
| DOUBLE(*m*,*d*) | Double precision floating point number. | Where ***m*** is the total digits  and ***d*** is the number of digits after the decimal. |
| FLOAT(p) | Floating point number. | Where ***p*** is the precision. |
| BOOL | Synonym for TINYINT(1) | Treated as a boolean data type where a value of 0 is considered to be FALSE and any other value is considered to be  TRUE. |
| BOOLEAN | Synonym for TINYINT(1) | Treated as a boolean data type where a value of 0 is considered to be FALSE and any other value is considered to be  TRUE. |

## Date/Time Data types

The following are the **Date/Time Data types** in MySQL:

|  |  |  |
| --- | --- | --- |
| **Data Type Syntax** | **Maximum Size** | **Explanation** |
| DATE | Values range from '1000-01-01' to '9999-12-31'. | Displayed as 'YYYY-MM- DD'. |
| DATETIME | Values range from '1000-01-01 00:00:00' to  '9999-12-31 23:59:59'. | Displayed as 'YYYY-MM- DD HH:MM:SS'. |
| TIMESTAMP(*m*) | Values range from '1970-01-01 00:00:01' UTC  to '2038-01-19 03:14:07' UTC. | Displayed as 'YYYY-MM- DD HH:MM:SS'. |
| TIME | Values range from '-838:59:59' to '838:59:59'. | Displayed as  'HH:MM:SS'. |
| YEAR[(2|4)] | Year value as 2 digits or 4 digits. | Default is 4 digits. |

# Datatype Attributes

### AUTO\_INCREMENT

* The auto increment attribute is used to assign unique integer identifier to newly inserted rows.
* It increments the field value by value when a new row is inserted.
* The auto increment mostly used with primary key.

### DEFAULT

* The DEFAULT attribute ensures that some constant value will be assigned when no other value is available.
* This value must be constant because MYSQL does not allow functional or expressional values to be inserted.
* If the NULL attribute has been assigned to this value the default value will be NULL and if no default is specified.

### NULL

* The null attribute indicates that no value can exist for the given field .
* The NULL attribute is assigned to a field by Default.

### NOT NULL

* If a column is defined as not null than one cannot insert a null value for this column.

### PRIMARY KEY

* Keyword **PRIMARY KEY** is used to define a column as primary key.
* It is used guarantee uniqueness for a given row. one cannot insert null values for the primary key column.
* There are two other values to ensure a record uniqueness.
  + **Single-field primary keys:** are used when there is a non-modifiable unique identifier for each row entered into the database. They are never changed once set.
  + **Multi-field primary keys:** are useful when it is not possible to guarantee uniqueness from any single field with a record. Thus, multiple fields are joined to ensure uniqueness.

### UNIQUE

* A column assigned the UNIQUE attribute will ensure that all values possess different values, except than NULL values are repeatable.

### ZEROFILL

* It is available to any of the numeric type and will result in the replacement of all remaining field space with zero.
* When used in conjunction with the optional (nonstandard) attribute ZEROFILL, the default padding of spaces is replaced with zeros.
* For example, for a column declared as INT(4) ZEROFILL, a value of 5 is retrieved as 0005.The ZEROFILL attribute is ignored when a column is involved in expressions.

### INDEX

* Indexing a column creates a sorted array of key for that column .each of which points to its corresponding table row.
* By creating index the searching will become faster.

### BINARY

* The binary attribute is only used with char and varchar values when columns are assigned this attributes they will be stored in case sensitive manner,(According to their ASCII value)

1. **UNSIGNED**: if specified, disallows negative values.

# Database functions

|  |  |
| --- | --- |
| **Function** | **Description** |
| mysql\_connect() | Opens a new connection to the MySQL server |
| mysql\_close() | Closes a previously opened database connection |
| mysql\_select\_db() | Selects the default database for database queries |
| mysql\_query() | Performs a query on the database |
| mysql\_affected\_rows() | Returns the number of affected rows in the previous MySQL operation |
| mysql\_error() | Returns the last error message for the most recent MySQL function call |
| mysql\_fetch\_array() | Fetches a result row as an associative, a numeric array, or both |
| mysql\_free\_result() | Frees the memory associated with a result |
| mysql\_num\_rows() | Returns the number of rows in a result set |
| mysql\_fetch\_row() | Fetches one row of data from the result set and returns it as an enumerated array |
| mysql\_fetch\_assoc() | Fetches a result row as an associative array |
| mysql\_result() | Retrieves the content of one cell(field) |
| mysql\_fetch\_object() | Returns row from a result set as an object |

## mysql\_connect()

* The mysql\_connect() function opens a new connection to the MySQL server.
* It creates a connection to a MySQL server. This function takes three parameters and returns a MySQL link identifier on success or FALSE on failure.

### Syntax:

**resource mysql\_connect(string hostname, string username, string password);**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Hostname/server | Optional - The host name running database server. If not specified, then default value is localhost:3036. |
| Username | Optional - The username accessing the database. If not specified, then default is the name of the user that owns the server process. |
| Password | Optional - The password of the user accessing the database. If not specified, then default is an empty password. |

### Example :

<?php

$conn = mysql\_connect("localhost", "root", "") or die("ERROR: Could not connect. ");

// ....some PHP code...

?>

## mysql\_close()

* The mysql\_close() function is used to close an open MySQL connection.
* The link to the MySQL server is closed when the script is terminated.
* It takes one argument which is the resource of the database. If the connection is not specified as a parameter within the mysql\_close(), the last opened link is used.
* If a resource parameter is not specified then last opened database is closed. This function returns true if it closes connection successfully otherwise it returns false.

### Syntax:

**bool mysql\_close ( [resource $link] );**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Link | Required. Specifies the MySQL connection link to close . |

### Example

### <?php

$conn=mysql\_connect("localhost","root","") die("ERROR: Could not connect. ");

// ....some PHP code...

Mysql\_close($conn)

?>

## mysql\_select\_db()

* It is used to select a database.
* It returns true on success and false on failure ,the name of the database as an argument and the resource name is optional.

### Syntax:

**bool mysql\_select\_db( String $databasename,[,resource $link]);**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Databasename/db\_name | Required - MySQL Database name to be select. |
| Link/connection | Optional - name of the mysql connection. If not specified, then last opened connection by mysql\_connect() will be used. |

### Example:

<?php

$conn=mysql\_connect("localhost","root"," ") or die("ERROR: Could not connect. "); mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

// ....some PHP code...

mysql\_close($conn);

?>

## mysql\_query()

* The mysql\_query() function executes a query on a MySQL database connection.
* It sends the query to the currently active database on the server. It takes two arguments. one is the query and the other is the resource which is optional.
* This function returns the query handle for SELECT queries, TRUE/FALSE for other queries, or FALSE on failure.

### Syntax:

**resource mysql\_query ( String $query [, resource $link] )**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Query | Required. Specifies the SQL query to send (should not end with a semicolon) |
| Link/connection | Optional. Specifies the MySQL connection. If not specified, the last connection opened by mysql\_connect() or mysql\_connect() is used. |

### Example:Insert Record in the table

<?php

$link=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

//open connection

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database.");

**//selects a database Employee**

mysql\_query("insert into emp\_per values(101,'Raman')",$conn) or die("ERROR: Could not inserted. ");

**//insert in to emp\_per table**

mysql\_close($conn);**//close a connection**

### Example: Update Record in the table

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

mysql\_query("Update emp\_per set name='Raj' where eno=101",$conn) or die("ERROR: Could not updated. ");

**//Update in to emp\_per table**

mysql\_close($conn);

**//close a connection**

?>

### Example:Delete Record in the table

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. "); mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. "); mysql\_query("delete from emp\_per where rollno=101",$conn) or die("ERROR: Could not deleted. ");

**//delete from student table**

mysql\_close($conn);

?>

## mysql\_num\_rows

* This function Retrieves the number of rows from a result set. This command is only valid for statements like SELECT or SHOW that return an actual result set.
* Resultset will be the array that is returned by mysql\_query() when used with select. This function returns False on failure.

### Syntax:

**int mysql\_num\_rows ( resource result ) Example:**

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect.); mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$query="select \* from emp\_per";

$resultset=mysql\_query($query,$conn);

$r=mysql\_num\_rows($resultset);

echo "<br>Rows: ".$r;

**//$r contains total no. of rows in a resultset**

mysql\_close($conn);

?>

This command is only valid for statements like SELECT that return an actual statment.To retrieve the number of rows affected by a INSERT, UPDATE, REPLACE or DELETE query, use [mysql\_affected\_rows()](http://www.nusphere.com/kb/phpmanual/function.mysql-affected-rows.htm)

## [mysql\_affected\_rows()](http://www.nusphere.com/kb/phpmanual/function.mysql-affected-rows.htm)

* The mysql\_affected\_rows() function is used to get the number of affected rows by the last MySQL query.
* If you run a mysql query to insert, update, replace or delete records, and want to know how many records are being affected by that query, you have to use mysql\_affected\_rows().
* This function returns the number of affected rows on success, or -1 if the last operation failed.

### Syntax:

**int mysql\_affected\_rows ( [resource $link] )**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Link/connection | Optional. Specifies the MySQL connection. If not specified, the last connection opened by mysql\_connect() is used. |

### Example:

<?php

$link=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die(mysql\_error());

mysql\_query("delete from emp\_per where rollno=101",$conn) or die(mysql\_error());

$r=mysql\_affected\_rows();

echo "<br>Deleted Records Are: ".$r;

//Prints total no. of rows deleted

mysql\_close($link);

//close a connection

?>

## mysql\_fetch\_array()

* This function fetches rows from the mysql\_query() function and returns an array on success .or false on failure or when there are no more rows.
* Array may be an associative array, a numeric array, or both.
* It moves the internal data pointer ahead.

### Syntax:

**array mysql\_fetch\_array ( resource $resultset [, int $result\_type=MYSQL\_BOTH] )**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Description** | | | |
| resultset | Refers to the resource return by a valid mysql query (calling by mysql\_query() function). | | | |
| resultset\_type | The type of the result is an array. Possible values : | | | |
|  | MYSQL\_ASSOC - Associative array MYSQL\_NUM - Numeric array MYSQL\_BOTH - Both associative  Default : MYSQL\_BOTH | and | numeric | array |

### Example: Display all Records

<?php

$link=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. "); mysql\_select\_db("college",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$conn) or die(mysql\_error()); while($row=mysql\_fetch\_array($resultset))

{

echo $row[0]."-".$row[1]."<br>";

}

mysql\_free\_result($resultset); mysql\_close($conn);//close a connection

?>

## mysql\_free\_result()

* The mysql\_free\_result() function frees memory used by a resultset handle. 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.
* All associated result memory is automatically freed at the end of the script's execution. This function returns TRUE on success, or FALSE on failure.

### Syntax:

**Example:**

<?php

bool mysql\_free\_result ( resource $resultset )

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$conn) or die(mysql\_error()); while($row=mysql\_fetch\_array($resultset))

{

echo $row[0]."-".$row[1]."<br>";

}

mysql\_free\_result($resultset); mysql\_close($conn);

?>

## mysql\_result()

* The mysql\_result() function returns the value of a field in a result set. This function returns the field value on success, or FALSE on failure.
* Iit requires three arguments: first is the result set, second will be the row number and third is optional argument which is the field. Its default value is zero.

### Syntax:

**String mysql\_result ( resource $resultset, int $row [, mixed $field=0] )**

|  |  |
| --- | --- |
| **Name** | **Description** |
| resultset | Refers to the resource return by a valid mysql query (calling by mysql\_query()). |
| Row | The row number from the result that's being retrieved. Row numbers start at 0. |
| Field | The name or position of the field being retrieved. |

### Example:

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$link) or die(mysql\_error()); echo mysql\_result($resultset,2);//Outputs third student's rollno

echo mysql\_result($resultset,2,1);//Outputs third student's name mysql\_close($conn);//close a connection

?>

## mysql\_error()

* The mysql\_error() function returns the error description of the last MySQL operation.
* This function returns an empty string ("") if no error occurs.

### Syntax:

**String mysql\_error ( [resource $link] )**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| $link | Optional. Specifies the MySQL connection. If not specified, the last connection opened by mysql\_connect()is used. |

### Example:

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

if(!$conn)

{

die(mysql\_error());

}

mysql\_close($conn);

?>

## mysql\_fetch\_row()

* The mysql\_fetch\_row() function returns a row from a resultset as a numeric array.
* This function gets a row from the mysql\_query() function and returns an array on success, or FALSE on failure or when there are no more rows.

### Syntax:

**array mysql\_fetch\_row ( resource $resultset )**

|  |  |
| --- | --- |
| **Name** | **Description** |
| resultset | Refers to the resource return by a valid mysql query (calling by mysql\_query() function). |

### Example:

<?php

$link=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. "); mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$conn) or die(mysql\_error()); while($row=mysql\_fetch\_row($resultset)

{

echo $row[0]; //returns 1st record's first column value echo $row[1]; //returns 1st record's second column value

}

mysql\_close($conn);//close a connection

?>

## mysql\_fetch\_object()

* The mysql\_fetch\_object() function returns a row from a resultset as an object.
* This function gets a row from the mysql\_query() function and returns an object on success, or FALSE on failure or when there are no more rows.

### Syntax:

**object mysql\_fetch\_object ( resource $resultset )**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Resultset | Refers to the resource return by a valid mysql query (calling by mysql\_query() function). |

### Example:

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$query="select \* from emp\_per";

$resultset=mysql\_query($query,$conn);

$while($row=mysql\_fetch\_object($resultset)

{

echo $row->rollno."<br/>";

}

mysql\_close($conn);

?>

## mysql\_fetch\_assoc()

* The mysql\_fetch\_assoc() used to retrieve a row of data as an associative array from a MySQL result handle.
* It Returns an associative array that corresponds to the fetched row and moves the internal pointer ahead, or FALSE if there are no more rows.

### Syntax:

**array mysql\_fetch\_assoc ( resource $resultset )**

|  |  |
| --- | --- |
| Name | Description |
| resultset | Refers to the resource return by a valid mysql query (calling by mysql\_query() function). |

### Example:

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$link) or die(mysql\_error()); while($row=mysql\_fetch\_assoc($resultset))

{

echo $row['rollno']."<br>";

echo $row['name']."<br>";

}

mysql\_free\_result($resultset); mysql\_close($conn);

?>

## mysql\_num\_fields()

* + This function returns number of fields in result set or success , or FALSE on failure.

### Syntax:

**int mysql\_num\_fields ( resource $resultset )**

### It Example:

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per",$link) or die(mysql\_error()); echo "Total fields are:".mysql\_num\_fields($resultset);

mysql\_close($conn);

?>

## Order By

* + The order by keyword is used to sort the data in result set.
  + The order by keyword sorts the record in ascending order by default in order to sort record is descending order you can use DESC keyword.

### Syntax:

SELECT *column\_name(s)*

FROM *table\_name Order by column\_name(s)[asc |desc]*

**This example selects all the records stored in the student table and sorts them by age column.**

**Database:**Employee

**Table:** Emp\_per(eno,ename,age)

**Example:**

<?php

$conn=mysql\_connect("localhost","root","") or die("ERROR: Could not connect. ");

mysql\_select\_db("Employee",$conn) or die("ERROR: Could not find database. ");

$resultset=mysql\_query("select \* from emp\_per order by age",$conn) or die(mysql\_error());

while($row=mysql\_fetch\_array($resultset))

{

echo $row[0]. $row[1]. $row[2].<br>";

}

mysql\_free\_result($resultset); mysql\_close($conn);

?>

* + It is also possible to order by more than one column.
  + When ordering by more than one column the 2nd column is only used the values in the 1st column are equal.

### Syntax:

SELECT *column\_name(s)*

FROM *table\_name Order by column1,column2*

# What is AJAX?

AJAX = Asynchronous JavaScript and XML.

AJAX is a technique for creating fast and dynamic web pages.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

*Example:*  Google Maps, Gmail, Youtube, and Facebook tabs

Code explanation:

First, check if the input field is empty (str.length == 0). If it is, clear the content of the txtHint placeholder and exit the function.

However, if the input field is not empty, do the following:

* Create an XMLHttpRequest object
* Create the function to be executed when the server response is ready
* Send the request off to a PHP file (gethint.php) on the server
  + State=1request has been setup
  + State=2 request sent
  + state=3 request process
  + State=4 request complete
  + Status=404 for file not found
  + Status=200 for file found
* Notice that q parameter is added to the url (gethint.php?q="+str)
* And the str variable holds the content of the input field

*Ajaxcall.php*

<html>

<head>

<script>

function showUser(str)

{

if (str == "")

{

document.getElementById("txtHint").innerHTML = "";

return;

} else {

var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function()

{

if (this.readyState == 4 && this.status == 200) {

document.getElementById("txtHint").innerHTML = this.responseText;

}

};

xmlhttp.open("GET","getuser.php?q="+str,true);

//true for asynchronous data

xmlhttp.send();

}

}

</script>

</head>

<body>

<form>

<select name="users" onchange="showUser(this.value)">

<option value="">Select a person:</option>

<option value="1">Raj</option>

<option value="2">Radha</option>

</select>

</form>

<br>

<div id="txtHint"><b>Person info will be listed here...</b></div>

</body>

</html>

*Get data from this file-> getuser.php*

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<?php

$q = $\_GET['q'];

$con = mysql\_connect("localhost","root","");

mysql\_select\_db("studend",$con);

$result = mysql\_query("SELECT \* FROM stud\_per WHERE sno='$q'",$con);

echo "<table>

<tr>

<th>Firstname</th>

<th>City</th>

</tr>";

while($row = mysql\_fetch\_array($result)) {

echo "<tr>";

echo "<td>" . $row['sno'] . "</td>";

echo "<td>" . $row['city'] . "</td>";

echo "</tr>";

}

echo "</table>";

mysql\_close($con);

?>

</body>

</html>

* A variable xmlhttp is declared. Then, a new XMLHttpRequest object is created. If a your target audience use browsers older than Internet Explorer 8, ActiveXObject is used to create XMLHttpRequest.
* 'onreadystatechange' is a property of XMLHttpRequest object which is called whenever 'readyState' attribute is changed.
* We check whether the value of the 'readyState' property is 4, which denotes that the operation is complete.
* If the operation is completed, the status of the response to the request is checked. It returns the HTTP result code. Result code 200 states that the response to the request is successful.
* Now we set the value of the string to be displayed within the div whose id is 'suggestion' as 'responseText' property of the XMLHttpRequest object. 'responseText' is the response to the request as text.
* By using 'open' method of XMLHttpRequest object, a new request to the server is initialized. There are three parameters passed by this method. 'POST' determines the type of the httprequest. 'book-suggestion.php' sets the server side file and setting the third parameter 'true' states that the request should be handled asynchronously.
* 'send' method is used to send data contained in the 'data' variable to the server.