GIFT School of Engineering and Applied Sciences

Fall 2022

CS-244 Database Systems-Lab

Lab-08 Manual

Integration of Front-End with Database & Data Retrieval Using PHP

Introduction to Lab

This lab extends the use of basic database operations. This lab introduces the usage of PHP for connecting to database and to perform basic CRUD operations that was introduced in the previous labs. Students are also introduced to execute the basic SQL statements in PHP.

The main topics of this lab include:

1. PHP MySQL Create Table
2. PHP MySQL Insert
3. PHP MySQL Delete
4. PHP MySQL Update
5. PHP MySQL Select
6. PHP MySQL Where
7. PHP MySQL Order By

Objectives of this Lab

At the end of this lab, students should be able to:

a. Connect to the database using PHP built-in functions
b. To create database using the PHP Script
c. Perform basic CRUD Operations using PHP
d. To use basic SQL statements using the PHP

1. What is MySQL?

MySQL database stores data into tables like other relational database. A table is a collection of related data, and it is divided into rows and columns.

Each row in a table represents a data record that are inherently connected to each other such as information related to a particular person, whereas each column represents a specific field such as *id*, *first_name*, *last_name*, *email*, etc.

1.1. Talking to MySQL Databases with SQL

SQL, the Structured Query Language, is a simple, standardized language for communicating with relational databases like MySQL. With SQL you can perform any database-related task, such as creating databases and tables, saving data in database tables, query a database for specific records, deleting and updating data in databases.

2. PHP MySQL Connection

2.1. Ways of Connecting to MySQL through PHP

In order to store or access the data inside a MySQL database, you first need to connect to the MySQL database server. PHP offers two different ways to connect to MySQL server: **MySQLi** (Improved MySQL) and **PDO** (PHP Data Objects) extensions.

While the PDO extension is more portable and supports more than twelve different databases, MySQLi extension as the name suggests supports MySQL database only. MySQLi extension however provides an easier way to connect to, and execute queries on, a MySQL database server. Both PDO and MySQLi offer an object-oriented API, but MySQLi also offers a procedural API which is relatively easy for beginners to understand.

Tip: The PHP's MySQLi extension provides both speed and feature benefits over the PDO extension, so it could be a better choice for MySQL-specific projects.

2.2. Connecting to MySQL Database Server

In PHP you can easily do this using the `mysqli_connect()` function. All communication between PHP and the MySQL database server takes place through this connection. Here're the basic syntaxes for connecting to MySQL using MySQLi.

```
$link = mysqli_connect("hostname", "username", "password", "database");
```

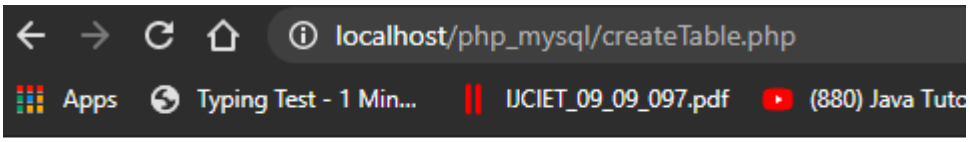
The *hostname* parameter in the above syntax specify the host name (e.g. `localhost`), or IP address of the MySQL server, whereas the *username* and *password* parameters specifies the credentials to access MySQL server, and the *database* parameter, if provided will specify the default MySQL database to be used when performing queries.

The following example shows how to connect to MySQL database server using MySQLi

```
<?php

$servername = "localhost";

$username = "root";

$password = "";

$database = "books";

//create connection to MySQL

$conn = mysqli_connect($servername , $username , $password ,
$database) or die("Cannot connect to
Database".mysqli_connect_error());

echo "Connected to database succesfully"."<br>";

 ?>
```

Note: The default username for MySQL database server is root and there is no password. However to prevent your databases from intrusion and unauthorized access you should set password for MySQL accounts.

Output of above Example in Web Browser:



Connected to database succesfully

3. PHP MySQL Create Table

In the previous topic we've learned how to connect to a MySQL database server. Now it's time to create some tables inside the database that will actually hold the data. A table organizes the information into rows and columns.

The SQL `CREATE TABLE` statement is used to create a table in database.

Let's make a SQL query using the `CREATE TABLE` statement, after that we will execute this SQL query through passing it to the PHP `query()` function to finally create our table.

```
<?php
include 'connect.php';
// SQL Create Table Query
$sql = "CREATE TABLE BookInfo(
      bid INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
      bname VARCHAR(30) NOT NULL,
      author VARCHAR(30) NOT NULL,
      price INT(6) NOT NULL,
      issuedate DATE
)";
if ($conn->query($sql) === TRUE) {
      # code...
      echo "Table created succesfully";
}else{
      echo "Error in Creating Table".$conn->error;
```

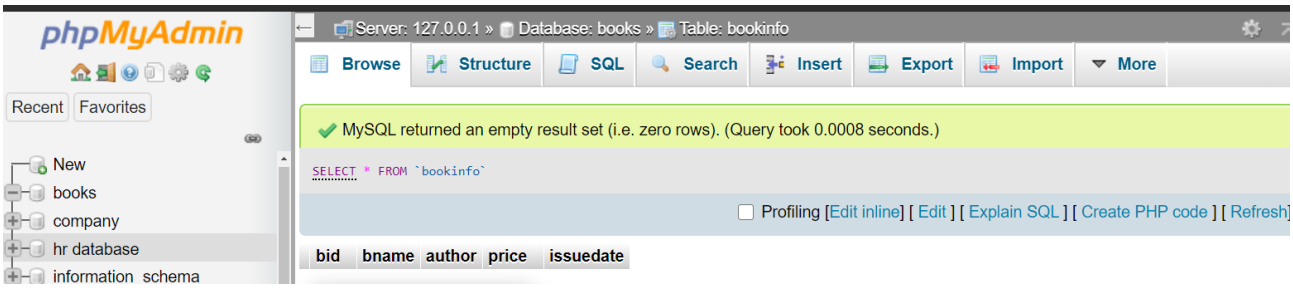
```
}
$conn->close();
 ?>
```

The PHP code in the above example creates a table named *BookInfo* with five columns b*id*, *bname*, *author, price* and *issuedate* inside the *Books* database.

Notice that each field name is followed by a data type declaration; this declaration specifies what type of data the column can hold, whether integer, string, date, etc.

There are a few additional constraints (also called *modifiers*) that are specified after the column name in the preceding SQL statement, like `NOT NULL`, `PRIMARY KEY`, `AUTO_INCREMENT`, etc. Constraints define rules regarding the values allowed in columns.

Output in phpMyAdmin:



4. PHP MySQL Insert

In this topic you will learn how to execute SQL query to insert records into a table.

The `INSERT INTO` statement is used to insert new rows in a database table.

Let's make a SQL query using the `INSERT INTO` statement with appropriate values, after that we will execute this insert query through passing it to the PHP `query()` function to insert data in table. Here's an example, which insert a new row to the *BookInfo* table by specifying values for the *bname*, *author, price* and *issuedate* fields.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
      <title>Add Book</title>
</head>
<body>
```

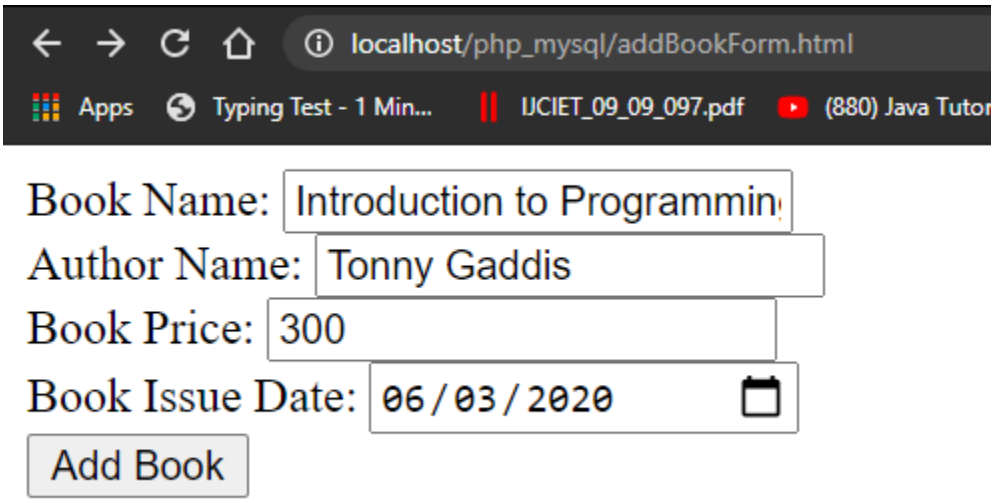
```
<form action="insert.php" method="post">

      Book Name: <input type="text" name="bname" placeholder="Enter Book name
here"><br>

      Author Name: <input type="text" name="author" placeholder="Enter Author
name here"><br>

      Book Price: <input type="number" name="price" placeholder="Enter Book
Price"><br>

      Book Issue Date: <input type="date" name="issuedate" placeholder="Enter
Issue date here"><br>

      <input type="submit" name="add" value="Add Book">

</form>

</body>

</html>
```

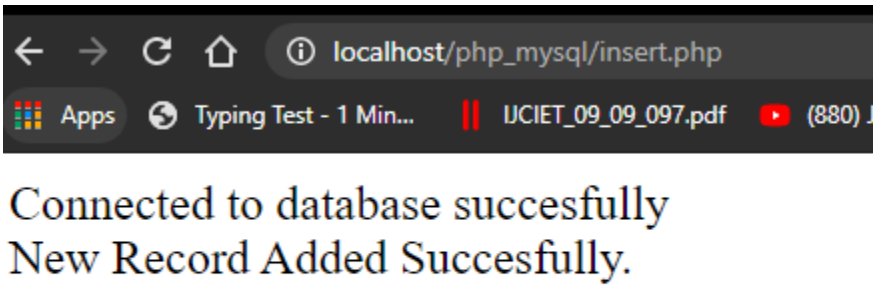
PHP Code

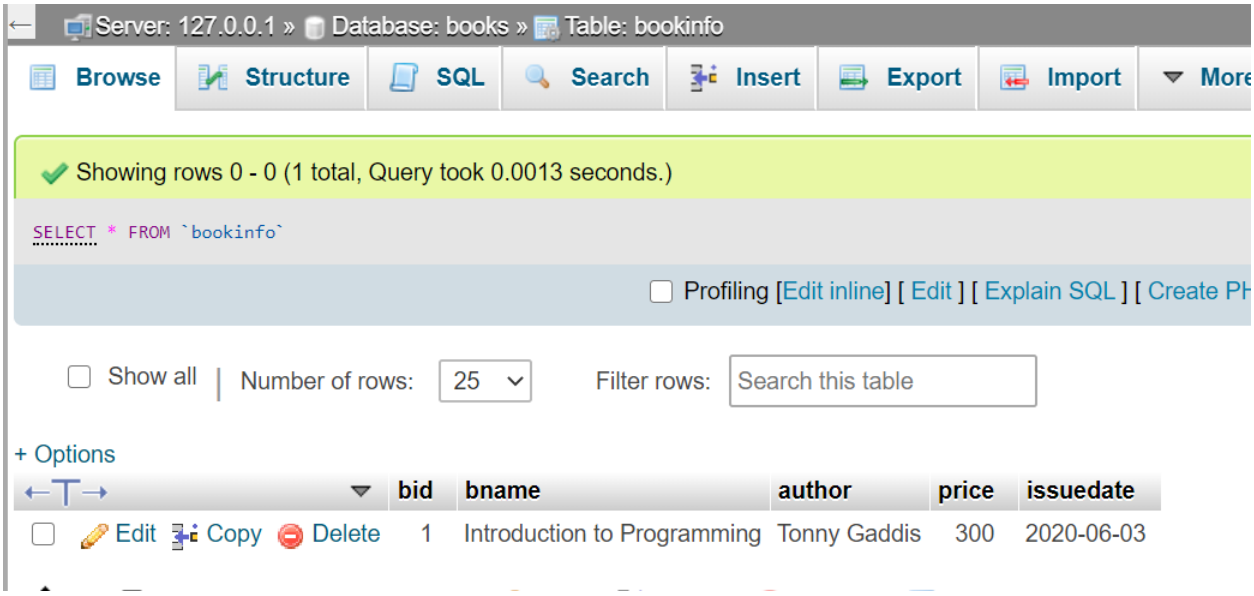
```
<?php

include 'connect.php';

$bname = $_REQUEST["bname"];

$author = $_REQUEST["author"];

$price = $_REQUEST["price"];

$issuedate = $_REQUEST["issuedate"];

$sql = "INSERT INTO BookInfo(bname,author,price,issuedate) VALUES('$bname'
, '$author', '$price' , '$issuedate')";

if ($conn->query($sql) === TRUE) {

      # code...

      echo "New Record Added Succesfully."."<br>";

}else{

      echo "Error:".$conn->error;

}

$conn->close();

?>
```

If you remember from the preceding topic, the b*id* field was marked with the `AUTO_INCREMENT` flag. This modifier tells the MySQL to automatically assign a value to this field if it is left unspecified, by incrementing the previous value by 1.

Output in the Web Browser:







5. PHP MySQL Delete Query

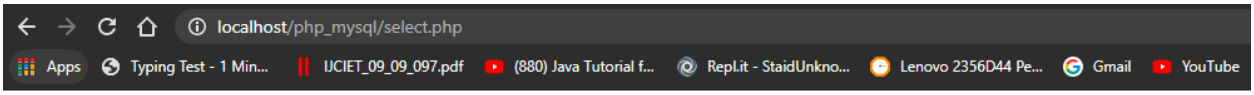
Just as you insert records into tables, you can delete records from a table using the SQL `DELETE` statement. It is typically used in conjugation with the `WHERE` clause to delete only those records that matches specific criteria or condition.

The basic syntax of the `DELETE` statement can be given with:

```
DELETE FROM table_name WHERE column_name=some_value
```

Let's make a SQL query using the `DELETE` statement and `WHERE` clause, after that we will execute this query through passing it to the PHP `$conn->query()` function to delete the tables records.

Consider the following *BookInfo* table inside the *Books* database:



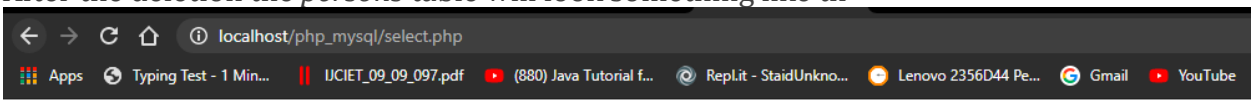
Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	390	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

The PHP code in the following example will delete the records of the book from the *BookInfo* table whose *Book id* is equal to "7".

```
<?php
include 'connect.php';


$sql = "DELETE FROM BookInfo WHERE bid=7";
if ($conn->query($sql) === TRUE) {
      # code...
      echo "Record Deleted Succesfully."."<br>";
}else{
      echo "Error:".$conn->error;
}
$conn->close();
 ?>
```

After the deletion the *persons* table will look something like th



Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	900	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

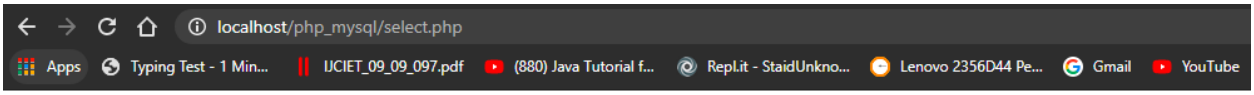
6. PHP MySQL Update Query

The `UPDATE` statement is used to change or modify the existing records in a database table. This statement is typically used in conjugation with the `WHERE` clause to apply the changes to only those records that matches specific criteria.

The basic syntax of the `UPDATE` statement can be given with:

```
UPDATE table_name SET column1=value,
column2=value2,... WHERE column_name=some_value
```

Let's make a SQL query using the `UPDATE` statement and `WHERE` clause, after that we will execute this query through passing it to the PHP `$conn->query()` function to update the tables records. Consider the following *BookInfo* table inside the *Books* database:



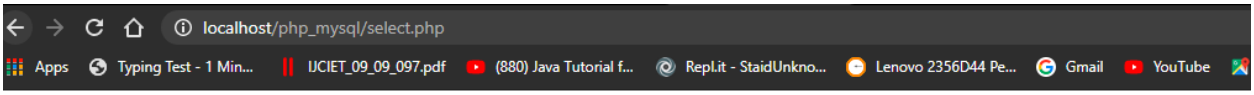
Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	390	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

The PHP code in the following example will update the price of a book in the BookInfo table whose Book *id* is equal to 3.

```
<?php
include 'connect.php';


$sql = "UPDATE BookInfo SET price = 900 WHERE bid = 3";
if ($conn->query($sql) === TRUE) {
      # code...
      echo "Record Updated Succesfully."."<br>";
}else{
      echo "Error:".$conn->error;
}
$conn->close();
 ?>
```

After update the BookInfo table will look something like this:



Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	900	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

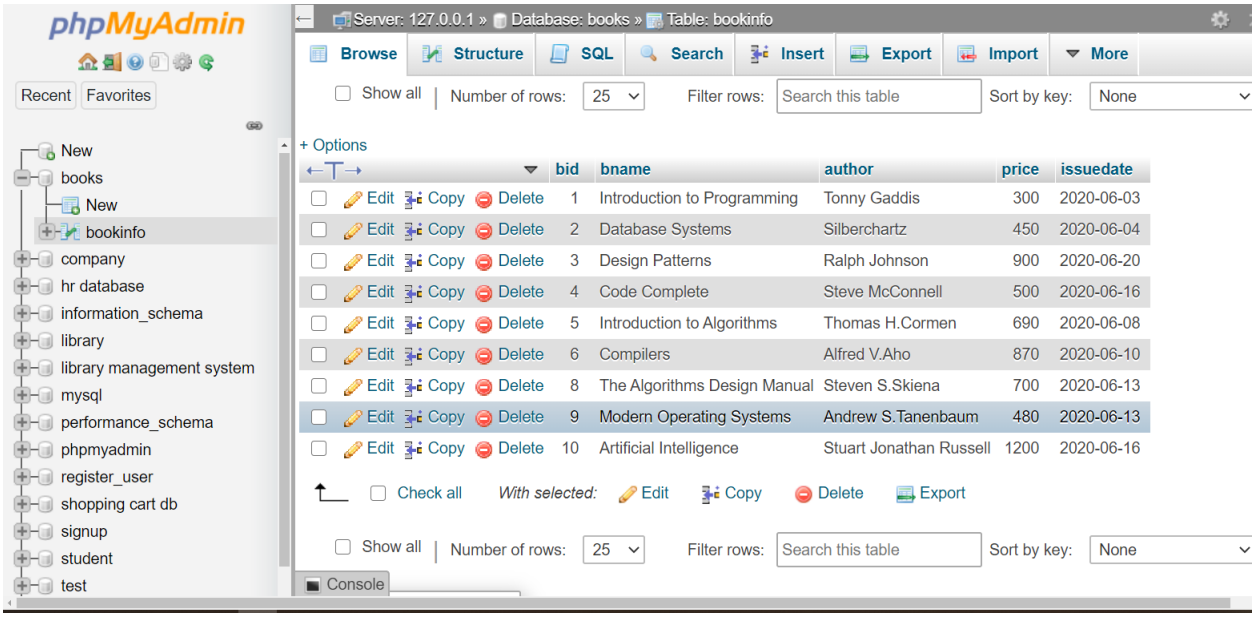
7. PHP MySQL Select

So far you have learnt how to connect to database and create table as well as inserting data. Now it's time to retrieve data what have inserted in the preceding tutorial. The SQL `SELECT` statement is used to select the records from database tables. Its basic syntax is as follows:

```
SELECT column1_name, column2_name, columnN_name FROM table_name;
```

Let's make a SQL query using the `SELECT` statement, after that we will execute this SQL query through passing it to the PHP `$conn->query()` function to retrieve the table data.

Consider our *BookInfo* database table has the following records:



The PHP code in the following example selects all the data stored in the *BookInfo* table (using the asterisk character (*) in place of column name selects all the data in the table).

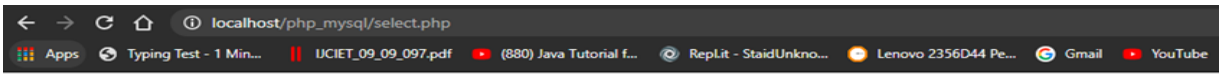
```
<?php
include 'connect.php';
$sql = "SELECT * FROM BookInfo";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
      echo "<table border=1>
                  <tr>
                        <th>Book ID</th>
                        <th>Book Name</th>
                        <th>Book Author</th>
                        <th>Book Price</th>
                        <th>Book Issue Date</th>
                  </tr>";
      while($row = $result->fetch_assoc()){
                  echo "<tr>
                        <td>".$row['bid']."</td>
                        <td>".$row['bname']."</td>
```

```
                    <td>".$row['author']."</td>
                    <td>".$row['price']."</td>
                    <td>".$row['issuedate']."</td>
                </tr>";
    }

echo "</table>";
}else{
    echo "No Result Found.";
}
$conn->close();

 ?>
```

Explanation of Code

In the example above, the data returned by the `$conn->query()` function is stored in the `$result` variable. Each time `mysqli_fetch_array()` is invoked, it returns the next row from the result set as an array. The while loop is used to loops through all the rows in the result set. Finally the value of individual field can be accessed from the row by passing the field name to the `$row` variable like `$row['bid']` , `$row['bname']`, `$row['author']`, `$row['price']`, and `$row['issuedate']` .

Output in the Web Browser:



Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	390	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

8. PHP MySQL Where

The `WHERE` clause is used to extract only those records that fulfill a specified condition.

The basic syntax of the `WHERE` clause can be given with:

```
SELECT column_name(s) FROM table_name WHERE column_name operator value
```

Let's make a SQL query using the `WHERE` clause in `SELECT` statement, after that we'll execute this query through passing it to the PHP `$conn->query()` function to get the filtered data.

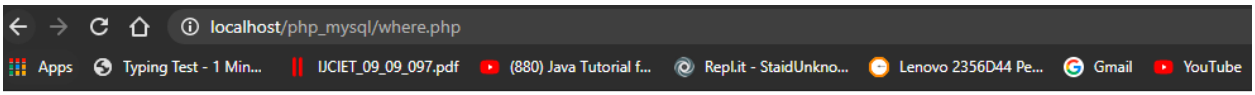
Consider we've a *BookInfo* table inside the *Books* database that has the records shown above:

Example-1: The following PHP code selects all the rows from the *BookInfo* table where price > 500:

```
<?php
include 'connect.php';
$sql = "SELECT * FROM BookInfo WHERE price > 500";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
      echo "<table border=1>
                  <tr>
                        <th>Book ID</th>
                        <th>Book Name</th>
                        <th>Book Author</th>
                        <th>Book Price</th>
                        <th>Book Issue Date</th>
                  </tr>";
      while($row = $result->fetch_assoc()){
                  echo "<tr>
                        <td>".$row['bid']."</td>
                        <td>".$row['bname']."</td>
                        <td>".$row['author']."</td>
                        <td>".$row['price']."</td>
```

```
                    <td>".$row['issuedate']."</td>
                </tr>";
        }

echo "</table>";
}else{
        echo "No Result Found.";
}
$conn->close();

 ?>
```

After filtration the result set will look something like this:



Connected to database succesfully

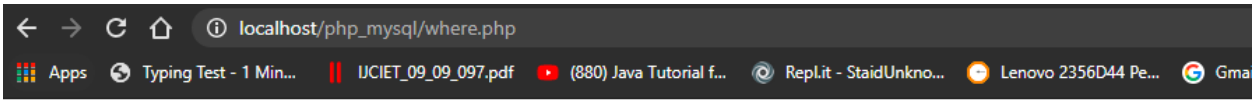
Book ID	Book Name	Book Author	Book Price	Book Issue Date
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

Example-2: The following PHP code selects all the rows from the *BookInfo* table whose Book name begins with 'A':

```
<?php
include 'connect.php';
$sql = "SELECT * FROM BookInfo WHERE bname LIKE 'A%'";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
        echo "<table border=1>
                    <tr>
```

```
                        <th>Book ID</th>
                        <th>Book Name</th>
                        <th>Book Author</th>
                        <th>Book Price</th>
                        <th>Book Issue Date</th>
                    </tr>";
        while($row = $result->fetch_assoc()){
                    echo "<tr>
                        <td>".$row['bid']."</td>
                        <td>".$row['bname']."</td>
                        <td>".$row['author']."</td>
                        <td>".$row['price']."</td>
                        <td>".$row['issuedate']."</td>
                    </tr>";
        }

echo "</table>";
}else{
        echo "No Result Found.";
}
$conn->close();


 ?>
```

After filtration the result set will look something like this:



Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

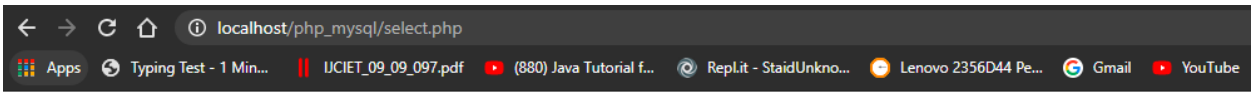
9. PHP MySQL Order By Clause

The `ORDER BY` clause can be used in conjugation with the `SELECT` statement to see the data from a table ordered by a specific field. The `ORDER BY` clause lets you define the field name to sort against and the sort direction either ascending or descending.

The basic syntax of this clause can be given with:

```
SELECT column_name(s) FROM table_name ORDER BY column_name(s) ASC|DESC
```

Let's make a SQL query using the `ORDER BY` clause in `SELECT` statement, after that we will execute this query through passing it to the PHP `$conn->query()` function to get the ordered data. Consider the following *BookInfo* table inside the *Books* database:



Connected to database succesfully

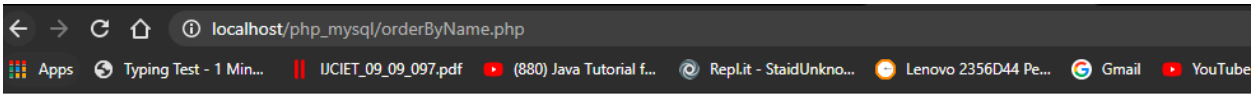
Book ID	Book Name	Book Author	Book Price	Book Issue Date
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	390	2020-06-20
4	Code Complete	Steve McConnell	500	2020-06-16
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
6	Compilers	Alfred V.Aho	870	2020-06-10
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16

The PHP code in the following example selects all rows from the *BookInfo* table and sorts the result by the *Book Name* column alphabetically in the ascending order.

```
<?php
include 'connect.php';
$sql = "SELECT * FROM BookInfo ORDER BY bname";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
      echo "<table border=1>
                  <tr>
                        <th>Book ID</th>
                        <th>Book Name</th>
                        <th>Book Author</th>
                        <th>Book Price</th>
                        <th>Book Issue Date</th>
                  </tr>";
      while($row = $result->fetch_assoc()){
                  echo "<tr>
                        <td>".$row['bid']."</td>
                        <td>".$row['bname']."</td>
                        <td>".$row['author']."</td>
                        <td>".$row['price']."</td>
                        <td>".$row['issuedate']."</td>
                  </tr>";
      }

echo "</table>";
}else{
      echo "No Result Found.";
}
$conn->close();
```

```
?>
```

After ordering the result, the result set will look something like this:



Connected to database succesfully

Book ID	Book Name	Book Author	Book Price	Book Issue Date
10	Artificial Intelligence	Stuart Jonathan Russell	1200	2020-06-16
4	Code Complete	Steve McConnell	500	2020-06-16
6	Compilers	Alfred V.Aho	870	2020-06-10
2	Database Systems	Silberchartz	450	2020-06-04
3	Design Patterns	Ralph Johnson	390	2020-06-20
5	Introduction to Algorithms	Thomas H.Cormen	690	2020-06-08
1	Introduction to Programming	Tonny Gaddis	300	2020-06-03
9	Modern Operating Systems	Andrew S.Tanenbaum	480	2020-06-13
7	Programming Pearls	Joh Louis Bentley	560	2020-06-17
8	The Algorithms Design Manual	Steven S.Skiena	700	2020-06-13

The End