Simple pagination in PHP with MySQL

In this article, you will learn how to create **simple pagination using PHP and MySQL**. Pagination mean retrieving and displaying your information into different pages rather than a single page. Pagination is important when we have to display large data on a single page. If you will be listed all the data on the same page, that will create issues like browser hang, high page loading time, long vertical scroll. It may also create confusion for readers. So, the best approach to split the records into chunks and displaying on multiple pages.

Here is the PHP pagination example, in which we have fetched the data from the database using the latest **PDO** (PHP Data Object) and written PHP logic to set pagination on the fetched records. All the coding flow is mentioned step by step, that will make you easier to understand and implement.

Database Connection

STEP 1: In the first step, we have written database connection code. For this, we have created a MySQL table '**students**' and inserted data in it as shown below. You can either use your existing database or copy & paste the below code in MySQL -

```
CREATE TABLE IF NOT EXISTS `students` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `first_name` varchar(100) NOT NULL,
  `last_name` varchar(100) NOT NULL,
  `email` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=11 ;



INSERT INTO `students` (`id`, `first_name`, `last_name`, `email`) VALUES
(1, 'John', 'Smith', 'john@example.com'),
(2, 'Soyam', 'Mithal', 's.mithal@example.com'),
(3, 'Sohn', 'Gaga', 'sohn@example.com'),
(4, 'Rita', 'Smith', 'rita@example.com'),
(5, 'Rohn', 'Mithal', 'r.mithal@example.com'),
(6, 'Sayam', 'Mitra', 'sayam.mitra@example.com'),
(7, 'Shyam', 'Mishra', 'shyam@example.com'),
(8, 'Ryan', 'Mithal', 'ryan@example.com'),
(9, 'Rohan', 'Soy', 'rohan.soy@example.com'),
(10, 'Mita', 'Dahl', 'mita@example.com');
```

Database Configuration File

Next, we have created a PHP file **configuration.php**, where we have written database connection code using **PHP PDO**. It is a lightweight interface for accessing databases and provides a data access abstraction layer for working with databases in PHP. Copy and paste this code in your configuration file. Only you will have to change the **database**, **hostname**, **username** and **password** with your database credentials and name.

configuration.php

```
<?php
   // define database related variables
   $database = 'db';
   $host = 'hostname';
   $user = 'username';
   $pass = 'password';

   // try to connect to database
   $db = new PDO("mysql:dbname={$database};host={$host};port={3306}", $user,
$pass);

   if(!$db){

      echo "Error in database connection";
   }
?>
```

data.php

STEP 2: In the second step, we have created another PHP file name '**data.php**'. In this page, we have set the data records limit on each page, start counter variable, next counter variable, previous counter variable. Like- on each page, we need to set the record limit to 4, so we have stored this in a variable **$per_page**, on the first page the page counter is 0, which is stored in **$page_counter**. If the user clicks on the next page, this counter increased by one and if the user clicks on the previous page, this counter decreased by one. We have passed these variables as **WHERE** clause to the **SELECT** statement. So that, exactly those amounts of data will fetch from the database that we have to display on the pagination page.

```
<?php
    //include configuration file
    require 'configuration.php';

    $start = 0;  $per_page = 4;
    $page_counter = 0;
    $next = $page_counter + 1;
    $previous = $page_counter - 1;

    if(isset($_GET['start'])){
     $start = $_GET['start'];
     $page_counter =  $_GET['start'];
     $start = $start *  $per_page;
     $next = $page_counter + 1;
     $previous = $page_counter - 1;
    }
    // query to get messages from messages table
    $q = "SELECT * FROM students LIMIT $start, $per_page";
    $query = $db->prepare($q);
    $query->execute();

    if($query->rowCount() > 0){
        $result = $query->fetchAll(PDO::FETCH_ASSOC);
    }
    // count total number of rows in students table
    $count_query = "SELECT * FROM students";
    $query = $db->prepare($count_query);
    $query->execute();
    $count = $query->rowCount();
    // calculate the pagination number by dividing total number of rows with per
page.
    $paginations = ceil($count / $per_page);
?>
```

index.php

STEP 3: In the third step, we have created an '**index.php**' file. This is the main file that we will call on the browser. This file contains mostly HTML code to display the records and pagination links with the pagination page numbers.

```
<html>
    <head>
        <title>Pagination</title>
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" />
    </head>
    <body>
        <?php include_once 'data.php'; ?>
        <div class="table-responsive" style="width: 600px; border: 1px solid
black; margin: 10px;">
            <table class="table table-striped" class="table table-hover">
                <thead class="table-info">
                 <th scope="col" class="bg-primary" >Id</th>
                 <th scope="col" class="bg-primary">First Name</th>
                 <th scope="col" class="bg-primary">Last Name</th>
                 <th scope="col" class="bg-primary">Email</th>
                </thead>
                <tbody>
                <?php
                    foreach($result as $data) {
                        echo '<tr>';
                        echo '<td>'.$data['id'].'</td>';
                        echo '<td>'.$data['first_name'].'</td>';
                        echo '<td>'.$data['last_name'].'</td>';
                        echo '<td>'.$data['email'].'</td>';
                        echo '</tr>';
                    }
                 ?>
                </tbody>
            </table>
            <center>
            <ul class="pagination">
            <?php
                if($page_counter == 0){
                    echo "<li><a href=?start='0' class='active'>0</a></li>";
                    for($j=1; $j < $paginations; $j++) {
                      echo "<li><a href=?start=$j>".$j."</a></li>";
                   }
                }else{
                    echo "<li><a href=?start=$previous>Previous</a></li>";
                    for($j=0; $j < $paginations; $j++) {
                     if($j == $page_counter) {
                        echo "<li><a href=?start=$j class='active'>".
$j."</a></li>";
                     }else{
                        echo "<li><a href=?start=$j>".$j."</a></li>";
                     }
                  }if($j != $page_counter+1)
                    echo "<li><a href=?start=$next>Next</a></li>";
                }
            ?>
            </ul>
            </center>
        </div>
    </body>
</html>
```

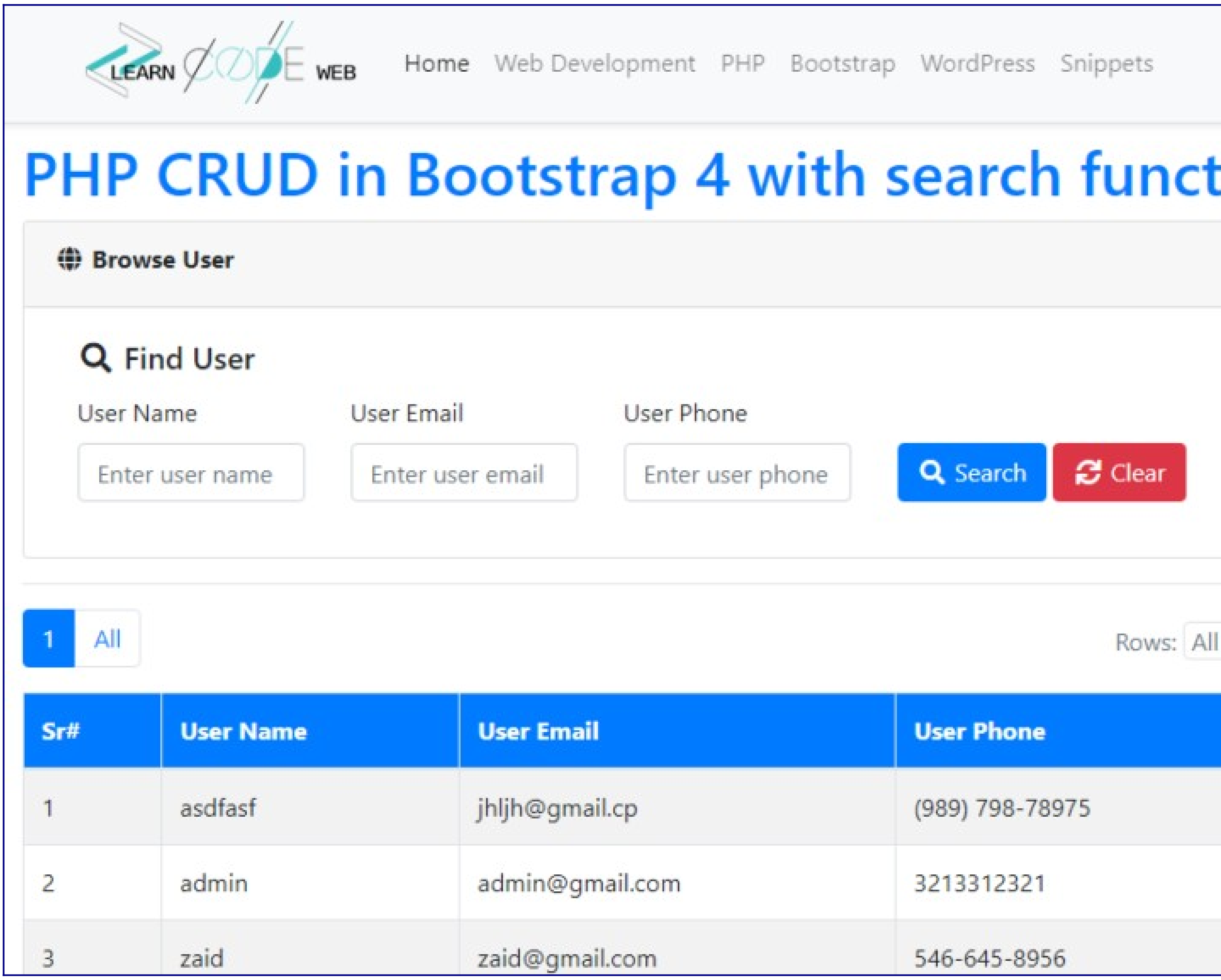
https://www.etutorialspoint.com/index.php/50-simple-pagination-in-php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Fan Club List</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
</head>
<body>

<?php

   $genders = [1 => 'PHD', 2 => 'Post Graduate', 3 => 'Graduate', 4 => 'HSC', 5 => 'SSC', 6 =>
'JSC'];


   $servername = "localhost";
   $username = "jago_fanclub";
   $password = "clubFanJago2018";
   $dbname = "jago_fanclub";

   $limit = 50;

   $db = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
   $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);


   $s = $db->prepare("SELECT * FROM club_fans");
   $s->execute();
   $allResp = $s->fetchAll(PDO::FETCH_ASSOC);
   // echo '<pre>';
   // var_dump($allResp);
   $total_results = $s->rowCount();
   $total_pages = ceil($total_results/$limit);

   if (!isset($_GET['page'])) {
      $page = 1;
   } else{
      $page = $_GET['page'];
   }


   $start = ($page-1)*$limit;

   $stmt = $db->prepare("SELECT * FROM club_fans ORDER BY id DESC LIMIT $start,
$limit");
   $stmt->execute();

   // set the resulting array to associative
```

```
    $stmt->setFetchMode(PDO::FETCH_OBJ);

    $results = $stmt->fetchAll();

    $conn = null;

    // var_dump($results);

    $no = $page > 1 ? $start+1 : 1;


?>

<div class="container">
  <h2 class="">Fan Club List <span class="badge">Total: <?= $total_results; ?></span></h2>

  <table class="table table-bordered">
    <thead>
      <tr>
        <th>#</th>
        <th>Name</th>
        <th>Email</th>
        <th>Phone</th>
        <th>Gender</th>
        <th>Education</th>
        <th>Ref Person</th>
      </tr>
    </thead>
    <tbody>
        <?php foreach($results as $result){?>
          <tr>
            <td><?= $no; ?></td>
            <td><?= $result->name; ?></td>
            <td><?= $result->email; ?></td>
            <td><?= $result->phone; ?></td>
            <td><?= $result->gender == 1 ? 'Male' : 'Female'; ?></td>
            <td><?= $genders[$result->education]; ?></td>
            <?php
                $refPerson = $db->prepare("SELECT name FROM club_fans where id=$result-
>reference_id");
                $refPerson->execute();
                $refPerson = $refPerson->fetch(PDO::FETCH_OBJ);

            ?>
            <td><?= $refPerson->name;?></td>
          </tr>
        <?php $no++; } ?>
    </tbody>
  </table>
    <ul class="pagination">
        <li><a href="?page=1">First</a></li>
        <?php for($p=1; $p<=$total_pages; $p++){?>

            <li class="<?= $page == $p ? 'active' : "; ?>"><a href="<?= '?page='.$p; ?>"><?= $p; ?></a></li>
        <?php }?>
        <li><a href="?page=<?= $total_pages; ?>">Last</a></li>
    </ul>
</div>

</body>
</html>
```

https://gist.github.com/RakibSiddiquee/9ea3578412d0ebf2cd9b2544d989fb91

PHP CRUD with Search and Pagination in Bootstrap 4



PHP CRUD in Bootstrap 4 with search funct

Browse User

Find User

User Name: Enter user name

User Email: Enter user email

User Phone: Enter user phone

Sr#	User Name	User Email	User Phone
1	asdfasf	jhljh@gmail.cp	(989) 798-78975
2	admin	admin@gmail.com	3213312321
3	zaid	zaid@gmail.com	546-645-8956

- 2 years ago
- Zaid Bin Khalid
- 18819 Views
- 12

In this tutorial, I am going to show you how you can create a simple ***CRUD*** in PHP with search and pagination functionality. For front end or user interface, I use Bootstrap 4 you can use any design framework as per your need.

Simple ***CRUD*** link without pagination is listed below.

1. PHP CRUD in Bootstrap 4 with search functionality
2. PHP CRUD in Bootstrap 3 with a search functionality

In this example, I will use the POD base DB class that holds all queries with a secure connection. To download DB class click here. And also download PHP pagination class from here. Add PHP pagination class file into the ***config.php*** with the help of ***include_once()*** function.

How to set up the config file.

Create a file with the name of ***config.php*** and then open the***config.php***file and paste below code in it.

Rememberinclude config file in all PHP files with the help ofinclude_once()function. It is the main file that holds a database connection. So without DB connection, you can not perform any DB operation into a database.

```
<?php
include_once('include/Database.php');
include_once('include/paginator.class.php');
define('SS_DB_NAME', 'test');
define('SS_DB_USER', 'root');
define('SS_DB_PASSWORD', '');
define('SS_DB_HOST', 'localhost');

$dsn = "mysql:dbname=".SS_DB_NAME.";host=".SS_DB_HOST."";
$pdo = "";
try {
$pdo = new PDO($dsn, SS_DB_USER, SS_DB_PASSWORD);
}catch (PDOException $e) {
echo "Connection failed: " . $e->getMessage();
}
$db = new Database($pdo);
$pages = new Paginator();
?>
```

In the config file, I already added the ***Database*** class with the help of ***include_once*** function. So you do not need to add this file again and again.

Paste below query to create table in ***mysql***.

```
CREATE TABLE `users` (
`id` INT NOT NULL AUTO_INCREMENT ,
`username` VARCHAR(64) NOT NULL ,
`useremail` VARCHAR(128) NOT NULL ,
`userphone` VARCHAR(24) NOT NULL ,
`dt` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP() ,
PRIMARY KEY (`id`)
) ENGINE = MyISAM;
```

Create the ***add-user.php*** file and paste below code just after the body tag.

```
<form method="post">
<div class="form-group">
<label>User Name <span class="text-danger">*</span></label>
<input type="text" name="username" id="username" class="form-
control" placeholder="Enter user name" required>
</div>
<div class="form-group">
<label>User Email <span class="text-danger">*</span></label>
<input type="email" name="useremail" id="useremail" class="form-
control" placeholder="Enter user email" required>
</div>
<div class="form-group">
<label>User Phone <span class="text-danger">*</span></label>
<input type="tel" name="userphone" id="userphone" class="form-
control" placeholder="Enter user phone" required>
</div>
<div class="form-group">
<button type="submit" name="submit" value="submit" id="submit"
class="btn btn-primary"><i class="fa fa-fw fa-plus-circle"></i>
Add User</button>
</div>
</form>
```

The above code snippet is based on Bootstrap 4. After creating a form paste below code just before the <***form***> tag. Below PHP code is used to take form data and submit into database.

```php
<?php
if(isset($_REQUEST['submit']) and $_REQUEST['submit']!=""){
extract($_REQUEST);
if($username==""){
header('location:'.$_SERVER['PHP_SELF'].'?msg=un');
exit;
}elseif($useremail==""){
header('location:'.$_SERVER['PHP_SELF'].'?msg=ue');
exit;
}elseif($userphone==""){
header('location:'.$_SERVER['PHP_SELF'].'?msg=up');
exit;
}else{
$data = array(
'username'=>$username,
'useremail'=>$useremail,
'userphone'=>$userphone,
);
$insert = $db->insert('users',$data);
if($insert){
header('location:'.$_SERVER['PHP_SELF'].'?msg=ras');
exit;
}else{
header('location:'.$_SERVER['PHP_SELF'].'?msg=rna');
exit;
}
}
}
?>
```

Success & ***Error*** message handling is listed below. You can past below code any where in the file where you want to show the message.

```php
<?php
if(isset($_REQUEST['msg']) and $_REQUEST['msg']=="un"){
echo '<div class="alert alert-danger"><i class="fa fa-
exclamation-triangle"></i> User name is mandatory field!</div>';
}elseif(isset($_REQUEST['msg']) and $_REQUEST['msg']=="ue"){
echo '<div class="alert alert-danger"><i class="fa fa-
exclamation-triangle"></i> User email is mandatory field!</div>';
}elseif(isset($_REQUEST['msg']) and $_REQUEST['msg']=="up"){
echo '<div class="alert alert-danger"><i class="fa fa-
exclamation-triangle"></i> User phone is mandatory field!</div>';
}elseif(isset($_REQUEST['msg']) and $_REQUEST['msg']=="ras"){
echo '<div class="alert alert-success"><i class="fa fa-thumbs-
up"></i> Record added successfully!</div>';
}elseif(isset($_REQUEST['msg']) and $_REQUEST['msg']=="rna"){
echo '<div class="alert alert-danger"><i class="fa fa-
exclamation-triangle"></i> Record not added <strong>Please try
again!</strong></div>';
}
```

```
?>
```

Create the ***browse-users.php*** file that shows user submitted data. So, in this file, we also need to set up pagination and search. So, first of all, we need to set up search conditions. Consider below code for search.

```
<?php
$condition = '';
if(isset($_REQUEST['username']) and $_REQUEST['username']!=""){
$condition .= ' AND username LIKE "%'.$_REQUEST['username'].'%" ';
}
if(isset($_REQUEST['useremail']) and $_REQUEST['useremail']!=""){
$condition .= ' AND useremail LIKE "%'.$_REQUEST['useremail'].'%" ';
}
if(isset($_REQUEST['userphone']) and $_REQUEST['userphone']!=""){
$condition .= ' AND userphone LIKE "%'.$_REQUEST['userphone'].'%" ';
}

//Main queries
$pages->default_ipp = 15; //total records show on per page
$sql = $db->getRecFrmQry("SELECT * FROM users WHERE 1 ".$condition."");
$pages->items_total = count($sql);
$pages->mid_range = 9;
$pages->paginate();

$userData = $db->getRecFrmQry("SELECT * FROM users WHERE 1 ".$condition." ORDER BY id DESC ".$pages->limit."");

?>
```

After above script, need to display user data in tabular format for that consider below script.

```
<table class="table table-striped table-bordered">
<thead>
<tr class="bg-primary text-white">
<th>Sr#</th>
<th>User Name</th>
<th>User Email</th>
<th>User Phone</th>
<th class="text-center">Action</th>
</tr>
</thead>
<tbody>
<?php
$s = '';
foreach($userData as $val){
$s++;
?>
<tr>
<td><?php echo $s;?></td>
<td><?php echo $val['username'];?></td>
<td><?php echo $val['useremail'];?></td>
<td><?php echo $val['userphone'];?></td>
<td align="center">
<a href="edit-users.php?editId=<?php echo $val['id'];?>"
class="text-primary"><i class="fa fa-fw fa-edit"></i> Edit</a> |
<a href="delete.php?delId=<?php echo $val['id'];?>" class="text-
danger"><i class="fa fa-fw fa-trash"></i> Delete</a>
</td>
</tr>
<?php } ?>
</tbody>
</table>
```

- 2 years ago
- Zaid Bin Khalid
- 18819 Views
- 12

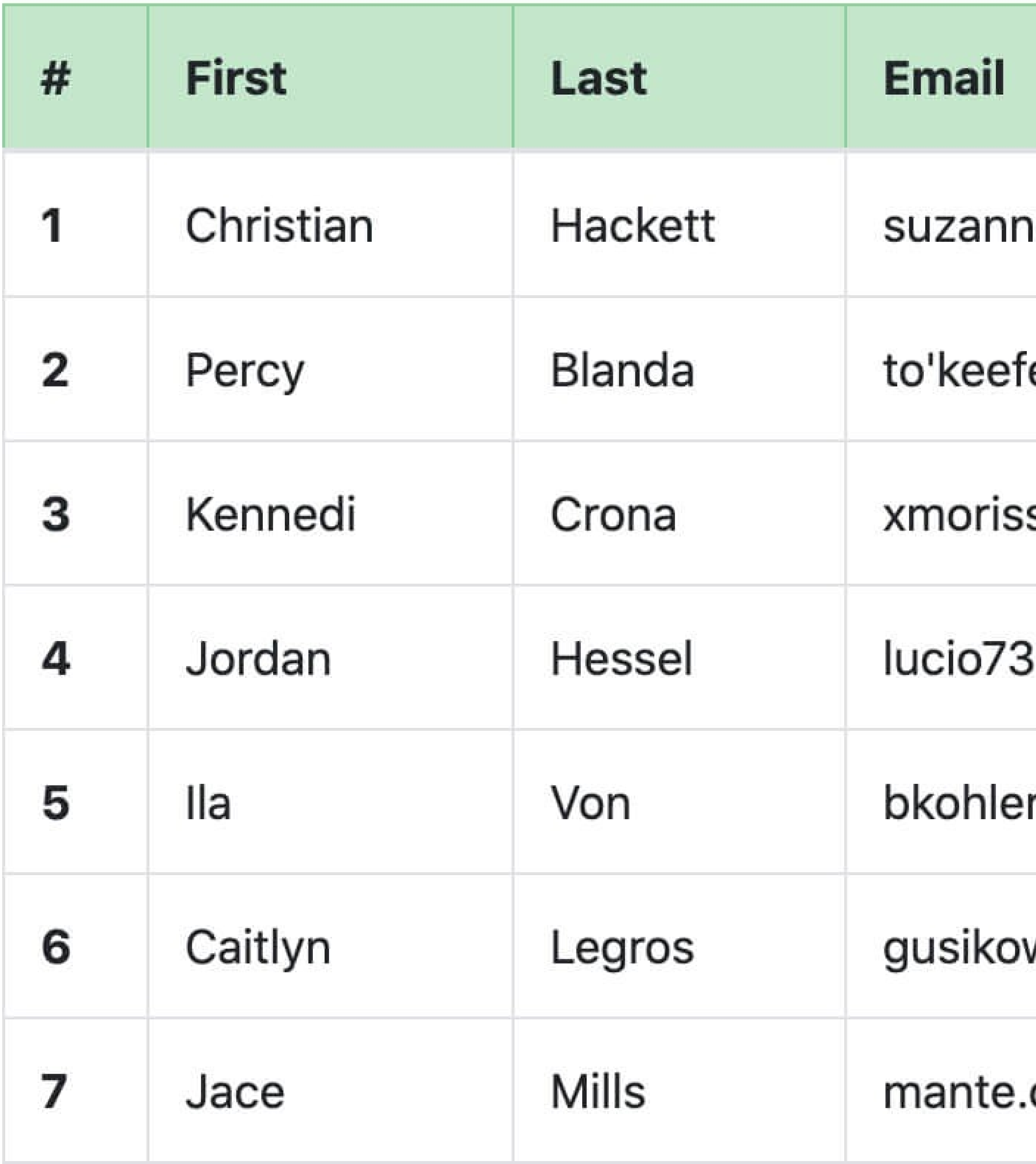
---

https://learncodeweb.com/web-development/php-crud-with-search-and-pagination-in-bootstrap-4/

Create Pagination in PHP 8 with MySQL and Bootstrap

Last updated on by Digamber

This tutorial explains how to create pagination in PHP 8 and MySQL using the Bootstrap 4 Pagination UI component. Also, learn how to set the dynamic limit in pagination with the session, create prev, next feature, active class in pagination to display results fetched from MySQL database.



#	First	Last	Email
1	Christian	Hackett	suzann
2	Percy	Blanda	to'keef
3	Kennedi	Crona	xmoriss
4	Jordan	Hessel	lucio73
5	Ila	Von	bkohler
6	Caitlyn	Legros	gusikow
7	Jace	Mills	mante.

LIVE DEMO

Pagination Features:

- Previous button in pagination to go back.
- Next button in pagination to go forward.
- Disable the previous button when reached the first item in pagination.
- Disable the next button when reached the last item in pagination.
- Add an active class in the pagination to set the active state.
- Set a dynamic limit for pagination items via the select dropdown.
- Set session for pagination limit.

What is Pagination?

Pagination is a graphical way of displaying a large set of data in smaller pieces, and It makes the data analysis process easy for the users.

If i try to make it simpler, so think about the search results you get when you write a search query in the Google search toolbar. You see the numerical list with previous and next links at the very bottom of your screen.

Google breaks down the million results in a smaller data set to show you the results in a more precise manner. It makes you quickly browse the information that you were looking for. This is known as the pagination.

Create Table & Insert Data

Create the `` `authors` `` table in the MySQL database.

```
CREATE TABLE `authors` (
  `id` int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT,
  `first_name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `last_name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `birthdate` date NOT NULL,
  `added` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

PL/SQL

Insert the following data inside the **authors'** table.

```
INSERT INTO `authors` (`id`, `first_name`, `last_name`, `email`, `birthdate`,
`added`) VALUES
(1, 'Christian', 'Hackett', 'suzanne41@example.com', '1983-12-30', '1992-02-05
13:21:46'),
(2, 'Percy', 'Blanda', 'to\'keefe@example.org', '2011-09-19', '1990-04-24
01:17:02'),
(3, 'Kennedi', 'Crona', 'xmorissette@example.com', '2013-12-17', '1973-03-17
13:21:12'),
(4, 'Jordan', 'Hessel', 'lucio73@example.com', '1975-04-17', '1970-10-18
14:43:11'),
(5, 'Ila', 'Von', 'bkohler@example.net', '1989-10-04', '2004-08-15 06:25:33'),
(6, 'Caitlyn', 'Legros', 'gusikowski.alycia@example.com', '2020-02-05', '1978-
01-05 20:54:52'),
(7, 'Jace', 'Mills', 'mante.claud@example.org', '2017-04-30', '1999-12-06
17:56:43'),
(8, 'Kiley', 'Hickle', 'megane34@example.net', '1999-09-16', '2014-05-27
22:54:34'),
(9, 'Keshaun', 'Swift', 'ahickle@example.com', '1984-05-27', '1979-06-15
02:41:44'),
(10, 'Bernhard', 'Hudson', 'ramiro46@example.com', '1996-09-30', '1987-10-15
19:29:03'),
(11, 'Brando', 'Maggio', 'katarina90@example.org', '2001-10-16', '1989-08-31
08:25:57'),
(12, 'Kariane', 'Dicki', 'hwilliamson@example.net', '2006-03-25', '2018-10-07
06:23:34'),
(13, 'Earnestine', 'Ankunding', 'nwindler@example.org', '1975-11-11', '2019-08-
20 17:12:29'),
(14, 'Nayeli', 'Schiller', 'camden.kemmer@example.net', '2005-01-28', '2008-02-
28 19:42:52'),
(15, 'Tressie', 'Willms', 'randerson@example.com', '1995-11-24', '2000-05-19
09:48:39'),
(16, 'Shaun', 'Walsh', 'howell.brenna@example.net', '1991-11-01', '1976-03-24
11:54:20'),
(17, 'Roosevelt', 'Leuschke', 'janiya.kub@example.com', '1984-12-16', '2004-10-
01 00:21:22'),
(18, 'Bill', 'Farrell', 'bins.moses@example.net', '1986-03-18', '1994-01-12
02:22:08'),
(19, 'Maurice', 'Johns', 'katelyn.friesen@example.org', '2000-12-07', '2004-07-
16 02:59:16'),
(20, 'Taya', 'Towne', 'vbauch@example.net', '1972-01-14', '2018-04-19
22:00:33'),
(21, 'Ivah', 'Kuhlman', 'vswaniawski@example.org', '2003-10-30', '2004-08-28
08:01:06'),
(22, 'Virgie', 'Quitzon', 'terrell.ratke@example.net', '1977-06-30', '1990-08-13
05:30:49'),
(23, 'Laurel', 'Lueilwitz', 'karen02@example.com', '1973-03-10', '2006-06-24
15:01:07'),
(24, 'Colton', 'Wisoky', 'ivory40@example.com', '2004-03-13', '1972-04-13
10:39:32'),
(25, 'Frankie', 'Kutch', 'schuster.adrianna@example.com', '1983-07-16', '1993-
03-27 06:29:23'),
(26, 'Noelia', 'Kertzmann', 'dubuque.blanca@example.org', '1990-10-18', '1989-
02-02 16:52:51'),
(27, 'Aida', 'Durgan', 'brendan05@example.org', '1979-05-30', '1996-08-20
08:45:41'),
(28, 'Vesta', 'Stiedemann', 'jo\'kon@example.net', '2019-03-18', '1977-11-04
12:13:54'),
(29, 'Emmy', 'Armstrong', 'schuster.adrienne@example.org', '1971-07-24', '1997-
08-23 02:34:33'),
(30, 'Melany', 'Kris', 'antonio.towne@example.net', '1970-05-03', '1993-01-11
04:26:59'),
(31, 'Valentine', 'Boyle', 'swift.joana@example.net', '1988-02-08', '2012-11-15
12:54:23'),
(32, 'Trisha', 'Gutmann', 'jdickinson@example.net', '1992-07-21', '1989-10-25
21:52:17'),
(33, 'Angela', 'Stoltenberg', 'walter.leta@example.com', '1973-08-15', '2008-11-
21 16:16:02'),
(34, 'Dulce', 'Bartoletti', 'mosciski.nolan@example.com', '2011-04-03', '2015-
10-07 05:27:01'),
(35, 'Haylie', 'Rohan', 'edna.maggio@example.net', '2003-07-15', '2005-05-10
00:13:04'),
(36, 'Daphney', 'Nikolaus', 'tdibbert@example.org', '1978-02-19', '1984-02-12
08:32:02'),
(37, 'Gabriella', 'Wolf', 'egutmann@example.org', '2009-11-28', '2001-10-20
06:25:35'),
(38, 'Elvie', 'Pfannerstill', 'aorn@example.org', '2014-08-14', '2015-10-19
13:48:05'),
(39, 'Elliot', 'Denesik', 'borer.tierra@example.net', '2005-02-28', '2015-01-29
07:09:30'),
(40, 'Jermaine', 'Cartwright', 'lhane@example.org', '2013-07-05', '1970-03-26
02:34:32'),
(41, 'Herminio', 'Rosenbaum', 'shanahan.gilda@example.com', '1997-10-06', '2010-
07-25 08:32:11'),
(42, 'Mateo', 'Raynor', 'esmeralda.yost@example.com', '2006-11-04', '2017-08-25
06:13:30'),
(43, 'Maymie', 'Runte', 'kwhite@example.com', '2000-06-19', '2018-06-01
05:42:58'),
(44, 'Demond', 'Skiles', 'schinner.westley@example.com', '1983-02-22', '2013-08-
11 14:39:05'),
(45, 'Arvel', 'Jones', 'udietrich@example.net', '1975-03-20', '1974-10-04
10:44:12'),
(46, 'Donavon', 'Thiel', 'smitham.keven@example.org', '1994-12-25', '2019-05-05
13:08:57'),
(47, 'Aiyana', 'Ziemann', 'katlyn.shields@example.com', '1987-02-18', '1982-12-
16 09:38:25'),
(48, 'Gillian', 'Streich', 'zmertz@example.com', '1976-07-07', '1990-09-03
09:25:48'),
(49, 'Bryon', 'Roob', 'rosanna03@example.com', '1979-06-21', '1979-03-28
01:58:17'),
(50, 'Wendy', 'McLaughlin', 'katelyn.howell@example.com', '2018-06-06', '2002-
10-11 21:50:33');
```

SQL

Make Database Connection

Open **config/db.php** file and place the following code to connect PHP project with MySQL database.

```
<?php

    $hostname = "localhost";
    $username = "root";
    $password = "";
    try {
        $connection = new PDO("mysql:host=$hostname;dbname=php_crud", $username, 
$password);
        // set the PDO error mode to exception
        $connection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    } catch(PDOException $e) {
        echo "Database connection failed: " . $e->getMessage();
    }

?>
```

PHP

We are using the Bootstrap library to create the table and pagination layout to display the Authors' results. However, you can use custom CSS to build pagination and table layout.

Calculate Total Pages with Dynamic Limit

The `$limit` variable sets the dynamic limit for displaying the result via the pagination and select dropdown. We wrap the results limit in session so if the user selects the limit from the dropdown, and the selected limit won't go away on browser refresh.

Grab the total id using the **SELECT** `count()` method to find out the total records in the table.

The **ceil()** function rounds the number up to the nearest integer.

```
// Dynamic limit
$limit = isset($_SESSION['records-limit']) ? $_SESSION['records-limit'] : 5;

// Get total records
$sql = $connection->query("SELECT count(id) AS id FROM authors")->fetchAll();
$allRecrods = $sql[0]['id'];

// Calculate total pages
$totoalPages = ceil($allRecrods / $limit);
```

PHP

PHP Pagination & SQL Query

MySQL gives a LIMIT clause that is used to define the number of records to return. The LIMIT clause allows displaying multi-page results via pagination with SQL and is very helpful with large tables.

To get the pagination number, we will define the page parameter later with pagination.

To get the offset or paginationStart we deduct the current page from 1 and divide it by the page limit.

```
$limit = isset($_SESSION['records-limit']) ? $_SESSION['records-limit'] : 5;

// Current pagination page number
$page = (isset($_GET['page']) && is_numeric($_GET['page']) ) ? $_GET['page'] : 
1;

// Offset
$paginationStart = ($page - 1) * $limit;

// Limit query
```

```
$authors = $connection->query("SELECT * FROM authors LIMIT $paginationStart,
$limit")->fetchAll();
```

PHP

Pagination Implementation with PHP

We have defined the formula and necessary variables, now we create the pagination and display the result based on data limitation. A user can go backward and forward using the pagination, see the active class for the current page.

```
<!-- Pagination -->
<nav aria-label="Page navigation example mt-5">
    <ul class="pagination justify-content-center">
        <li class="page-item <?php if($page <= 1){ echo 'disabled'; } ?>">
            <a class="page-link"
                href="<?php if($page <= 1){ echo '#'; } else { echo "?page=" .
$prev; } ?>">Previous</a>
        </li>

        <?php for($i = 1; $i <= $totoalPages; $i++ ): ?>
        <li class="page-item <?php if($page == $i) {echo 'active'; } ?>">
            <a class="page-link" href="index.php?page=<?= $i; ?>"> <?= $i; ?>
</a>
        </li>
        <?php endfor; ?>

        <li class="page-item <?php if($page >= $totoalPages) { echo
'disabled'; } ?>">
            <a class="page-link"
                href="<?php if($page >= $totoalPages){ echo '#'; } else {echo "?
page=". $next; } ?>">Next</a>
        </li>
    </ul>
</nav>
```

PHP

Set Dynamic Records Limit

First import the jQuery CDN link, we need to get the value from the select dropdown.

Run a loop and pass the values that we want to use for setting up the records limit. We are taking the values from the session and set the same value as a selected.

```
<form action="index.php" method="post">
        <select name="records-limit" id="records-limit" class="custom-select">
            <option disabled selected>Records Limit</option>
            <?php foreach([5,7,10,12] as $limit) : ?>
            <option
                <?php if(isset($_SESSION['records-limit']) &&
$_SESSION['records-limit'] == $limit) echo 'selected'; ?>
                value="<?= $limit; ?>">
                <?= $limit; ?>
            </option>
            <?php endforeach; ?>
        </select>
</form>
```

```
<!-- jQuery + Bootstrap JS -->
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script>
    $(document).ready(function () {
        $('#records-limit').change(function () {
            $('form').submit();
        })
    });
</script>
```

PHP

Pagination in PHP Code Example

The following is the final code for pagination example in PHP with previous and next buttons.

```
<?php
  // Database
  include('config/db.php');

  // Set session
  session_start();
  if(isset($_POST['records-limit'])){
      $_SESSION['records-limit'] = $_POST['records-limit'];
  }

  $limit = isset($_SESSION['records-limit']) ? $_SESSION['records-limit'] : 5;
  $page = (isset($_GET['page']) && is_numeric($_GET['page']) ) ? $_GET['page'] :
1;
  $paginationStart = ($page - 1) * $limit;
  $authors = $connection->query("SELECT * FROM authors LIMIT $paginationStart,
$limit")->fetchAll();

  // Get total records
  $sql = $connection->query("SELECT count(id) AS id FROM authors")->fetchAll();
  $allRecrods = $sql[0]['id'];

  // Calculate total pages
  $totoalPages = ceil($allRecrods / $limit);

  // Prev + Next
  $prev = $page - 1;
  $next = $page + 1;
?>

<!doctype html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">
    <title>PHP Pagination Example</title>
    <style>
        .container {
            max-width: 1000px
        }

        .custom-select {
            max-width: 150px
        }
    </style>
</head>

<body>
    <div class="container mt-5">
        <h2 class="text-center mb-5">Simple PHP Pagination Demo</h2>


        <!-- Select dropdown -->
        <div class="d-flex flex-row-reverse bd-highlight mb-3">
            <form action="index.php" method="post">
                <select name="records-limit" id="records-limit" class="custom-
select">
                    <option disabled selected>Records Limit</option>
                    <?php foreach([5,7,10,12] as $limit) : ?>
                    <option
                        <?php if(isset($_SESSION['records-limit']) &&
$_SESSION['records-limit'] == $limit) echo 'selected'; ?>
                        value="<?= $limit; ?>">
                        <?= $limit; ?>
                    </option>
                    <?php endforeach; ?>
                </select>
            </form>
        </div>

        <!-- Datatable -->
        <table class="table table-bordered mb-5">
            <thead>
                <tr class="table-success">
                    <th scope="col">#</th>
                    <th scope="col">First</th>
                    <th scope="col">Last</th>
                    <th scope="col">Email</th>
                    <th scope="col">DOB</th>
                </tr>
            </thead>
            <tbody>
                <?php foreach($authors as $author): ?>
                <tr>
                    <th scope="row"><?php echo $author['id']; ?></th>
                    <td><?php echo $author['first_name']; ?></td>
                    <td><?php echo $author['last_name']; ?></td>
                    <td><?php echo $author['email']; ?></td>
                    <td><?php echo $author['birthdate']; ?></td>
                </tr>
                <?php endforeach; ?>
            </tbody>
        </table>

        <!-- Pagination -->
        <nav aria-label="Page navigation example mt-5">
            <ul class="pagination justify-content-center">
                <li class="page-item <?php if($page <= 1){ echo 'disabled'; } ?
>">
                    <a class="page-link"
                        href="<?php if($page <= 1){ echo '#'; } else { echo "?
page=" . $prev; } ?>">Previous</a>
                </li>

                <?php for($i = 1; $i <= $totoalPages; $i++ ): ?>
                <li class="page-item <?php if($page == $i) {echo 'active'; } ?
>">
                    <a class="page-link" href="index.php?page=<?= $i; ?>"> <?=
$i; ?> </a>
                </li>
                <?php endfor; ?>

                <li class="page-item <?php if($page >= $totoalPages) { echo
'disabled'; } ?>">
                    <a class="page-link"
                        href="<?php if($page >= $totoalPages){ echo '#'; } else
{echo "?page=". $next; } ?>">Next</a>
                </li>
            </ul>
        </nav>
    </div>

    <!-- jQuery + Bootstrap JS -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

    <script>
        $(document).ready(function () {
            $('#records-limit').change(function () {
                $('form').submit();
            })
        });
    </script>
</body>

</html>
```

PHP

Conclusion

We learned to divide the broad set of data into multiple pages using the pagination. I hope you liked this tutorial, don’t forget to share it with others.

You can also download the full code of this project from GitHub.

https://www.positronx.io/create-pagination-in-php-with-mysql-and-bootstrap/

How to Paginate Data With PHP

by Jason
Difficulty:IntermediateLength:LongLanguages:
PHPWeb Development

I can remember years ago, when I first began coding in PHP and MySQL, how excited I was the first time I got information from a database to show up in a web browser.

For someone who had little database and programming knowledge, seeing those table rows show up onscreen based on the code I wrote (okay, so I copied an example from a book) gave me a triumphant feeling. I may not have fully understood all the magic at work back then, but that first success spurred me on to bigger and better projects.

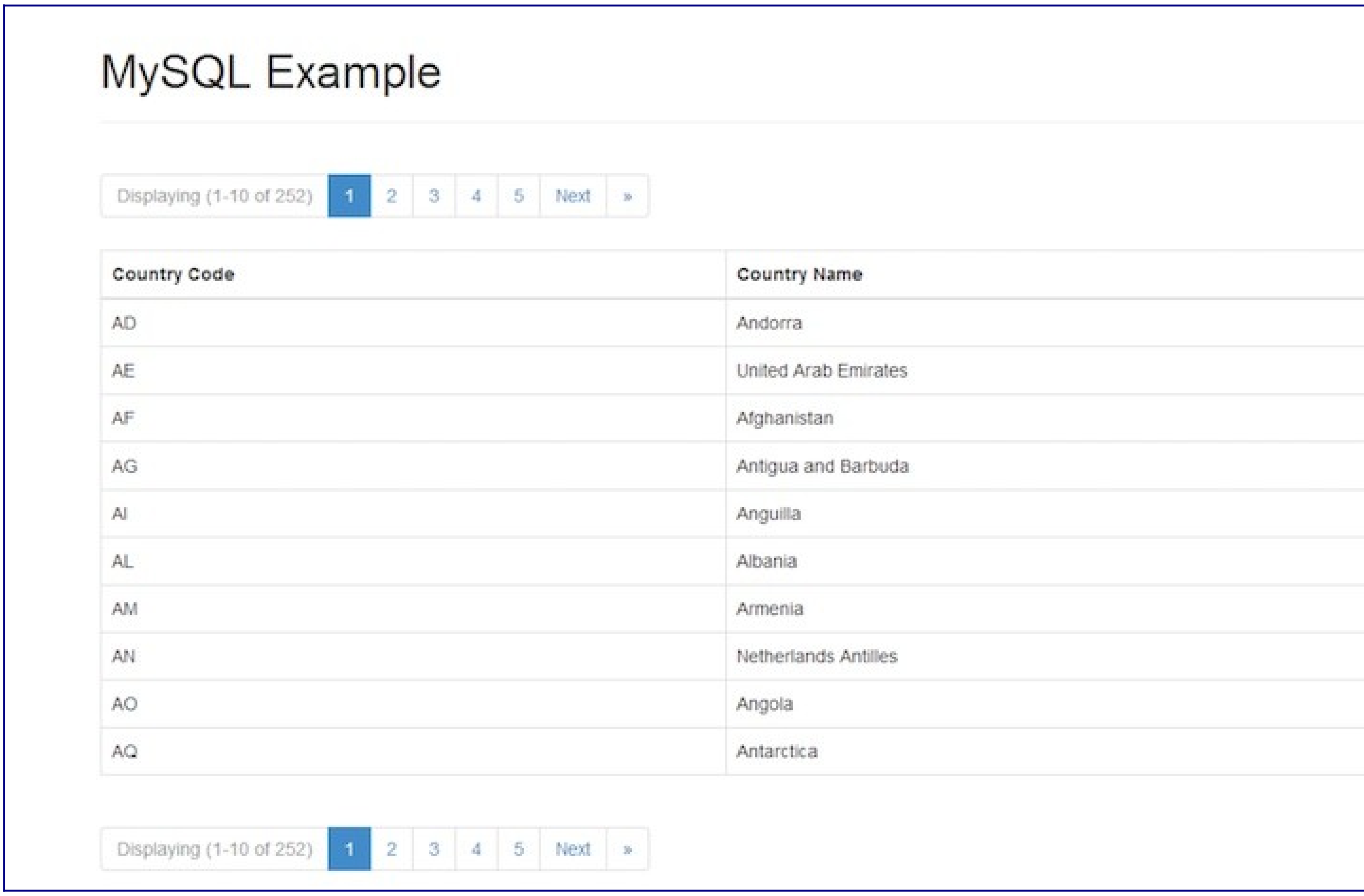
While my level of exuberance over databases may not be the same as it once was, ever since my first 'hello world' encounter with PHP and MySQL I've been hooked on the power of making things simple and easy to use.

As a developer, one problem I'm constantly faced with is taking a large set of information and making it easy to digest. Whether it's a large company's client list or a personal MP3 catalog, having to sit and stare at rows upon rows upon rows of data can be discouraging and frustrating. What can a good developer do? Paginate!

Looking for a Quick Solution?

If you're looking for a quick solution, there's a great collection of pagination scripts and helpers over at Envato Market.

For example, try out the ZPager PHP pagination class. It makes it easy to generate listing pagination links and customize them in just a few clicks. ZPager supports Bootstrap, MySQL, and Smarty Template Engine, and it also lets you use SEO-friendly URLs.



MySQL Example

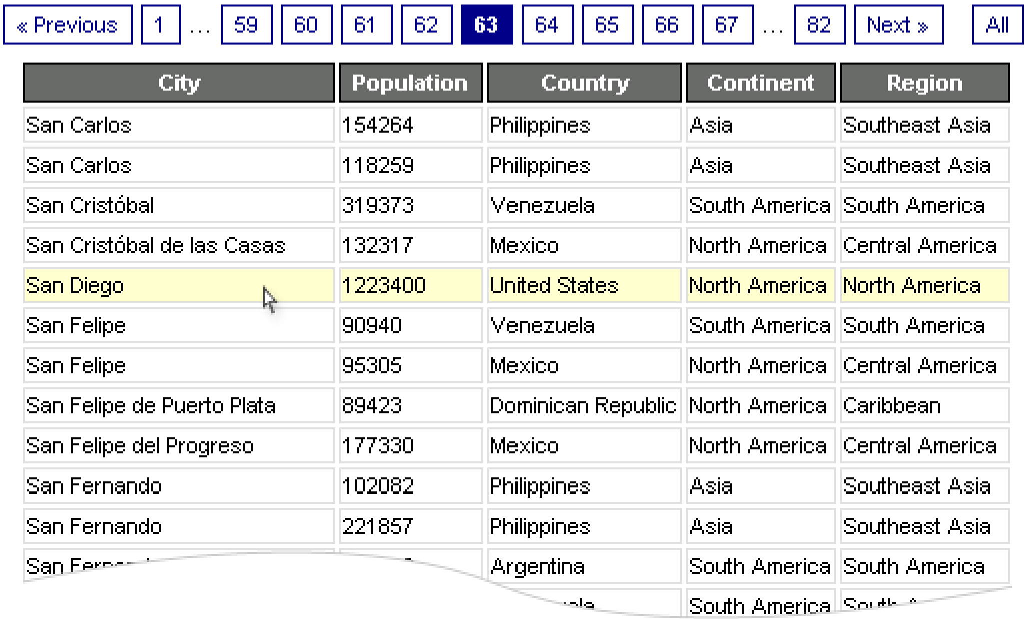
Displaying (1-10 of 252) 1 2 3 4 5 Next »

Country Code	Country Name
AD	Andorra
AE	United Arab Emirates
AF	Afghanistan
AG	Antigua and Barbuda
AI	Anguilla
AL	Albania
AM	Armenia
AN	Netherlands Antilles
AO	Angola
AQ	Antarctica

Displaying (1-10 of 252) 1 2 3 4 5 Next »

1. Pagination

Pagination is essentially the process of taking a set of results and spreading them out over pages to make them easier to view.



« Previous 1 … 59 60 61 62 63 64 65 66 67 … 82 Next » All

City	Population	Country	Continent	Region
San Carlos	154264	Philippines	Asia	Southeast Asia
San Carlos	118259	Philippines	Asia	Southeast Asia
San Cristóbal	319373	Venezuela	South America	South America
San Cristóbal de las Casas	132317	Mexico	North America	Central America
San Diego	1223400	United States	North America	North America
San Felipe	90940	Venezuela	South America	South America
San Felipe	95305	Mexico	North America	Central America
San Felipe de Puerto Plata	89423	Dominican Republic	North America	Caribbean
San Felipe del Progreso	177330	Mexico	North America	Central America
San Fernando	102082	Philippines	Asia	Southeast Asia
San Fernando	221857	Philippines	Asia	Southeast Asia
San Fern		Argentina	South America	South America
			South America	South

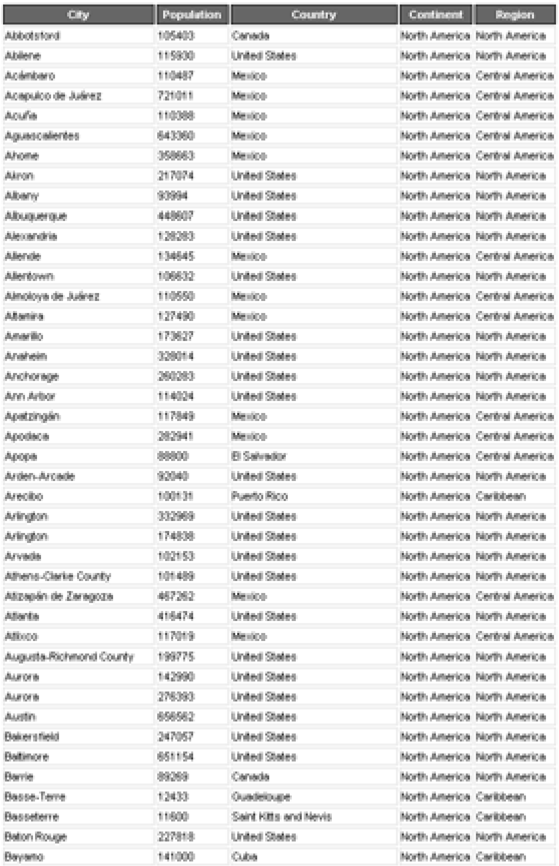
I realized early on that if I had 5,000 rows of information to display, not only would it be a headache for someone to try and read, but most browsers would take an Internet eternity (i.e. more than about five seconds) to display it.

To solve this, I created a simple, flexible, and easy-to-use PHP class that I could use for pagination in all my projects.

The Database

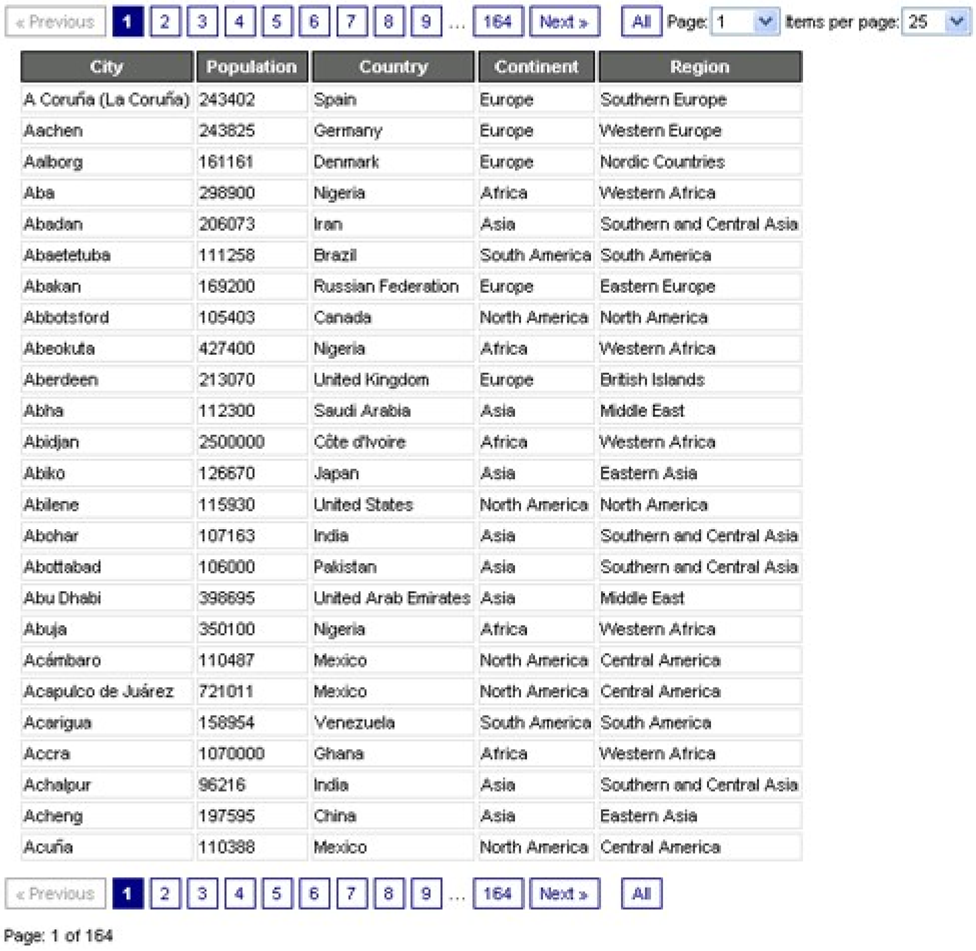
Gotta love MySQL. No offense to the other database systems out there, but for me, all I need is MySQL. And one great feature of MySQL is that they give you some free sample databases to play with at https://dev.mysql.com/doc/#sampledb.

For my examples, I'll be using the world database (~90k zipped), which contains over 4,000 records to play with, but the beauty of the PHP script we'll be creating is that it can be used with any database. Now I think we can all agree that if we decided not to paginate our results, we would end up with some very long and unwieldy results like the following:



City	Population	Country	Continent	Region
Abbotsford	105403	Canada	North America	North America
Abilene	115930	United States	North America	North America
Acámbaro	110487	Mexico	North America	Central America
Acapulco de Juárez	721011	Mexico	North America	Central America
Acuña	110388	Mexico	North America	Central America
Aguascalientes	643360	Mexico	North America	Central America
Ahome	358663	Mexico	North America	Central America
Akron	217074	United States	North America	North America
Albany	93994	United States	North America	North America
Albuquerque	448607	United States	North America	North America
Alexandria	128283	United States	North America	North America
Allende	134645	Mexico	North America	Central America
Allentown	106632	United States	North America	North America
Almoloya de Juárez	110550	Mexico	North America	Central America
Altamira	127490	Mexico	North America	Central America
Amarillo	173627	United States	North America	North America
Anaheim	328014	United States	North America	North America
Anchorage	260283	United States	North America	North America
Ann Arbor	114024	United States	North America	North America
Apatzingán	117849	Mexico	North America	Central America
Apodaca	282941	Mexico	North America	Central America
Apopa	88800	El Salvador	North America	Central America
Arden-Arcade	92040	United States	North America	North America
Arecibo	100131	Puerto Rico	North America	Caribbean
Arlington	332969	United States	North America	North America
Arlington	174838	United States	North America	North America
Arvada	102153	United States	North America	North America
Athens-Clarke County	101489	United States	North America	North America
Atizapán de Zaragoza	467262	Mexico	North America	Central America
Atlanta	416474	United States	North America	North America
Atlixco	117019	Mexico	North America	Central America
Augusta-Richmond County	199775	United States	North America	North America
Aurora	142990	United States	North America	North America
Aurora	276393	United States	North America	North America
Austin	656562	United States	North America	North America
Bakersfield	247057	United States	North America	North America
Baltimore	651154	United States	North America	North America
Barrie	89269	Canada	North America	North America
Basse-Terre	12433	Guadeloupe	North America	Caribbean
Basseterre	11600	Saint Kitts and Nevis	North America	Caribbean
Baton Rouge	227818	United States	North America	North America
Bayamo	141000	Cuba	North America	Caribbean

So let's gets down to breaking up our data into easy-to-digest bites like this:



Page: 1 Items per page: 25

City	Population	Country	Continent	Region
A Coruña (La Coruña)	243402	Spain	Europe	Southern Europe
Aachen	243825	Germany	Europe	Western Europe
Aalborg	161161	Denmark	Europe	Nordic Countries
Aba	298900	Nigeria	Africa	Western Africa
Abadan	206073	Iran	Asia	Southern and Central Asia
Abaetetuba	111258	Brazil	South America	South America
Abakan	169200	Russian Federation	Europe	Eastern Europe
Abbotsford	105403	Canada	North America	North America
Abeokuta	427400	Nigeria	Africa	Western Africa
Aberdeen	213070	United Kingdom	Europe	British Islands
Abha	112300	Saudi Arabia	Asia	Middle East
Abidjan	2500000	Côte d'Ivoire	Africa	Western Africa
Abiko	126670	Japan	Asia	Eastern Asia
Abilene	115930	United States	North America	North America
Abohar	107163	India	Asia	Southern and Central Asia
Abottabad	106000	Pakistan	Asia	Southern and Central Asia
Abu Dhabi	398695	United Arab Emirates	Asia	Middle East
Abuja	350100	Nigeria	Africa	Western Africa
Acámbaro	110487	Mexico	North America	Central America
Acapulco de Juárez	721011	Mexico	North America	Central America
Acarigua	158954	Venezuela	South America	South America
Accra	1070000	Ghana	Africa	Western Africa
Achalpur	96216	India	Asia	Southern and Central Asia
Acheng	197595	China	Asia	Eastern Asia
Acuña	110388	Mexico	North America	Central America

Page: 1 of 164

Beautiful, isn't it? Once you drop the pagination class into your code, you can quickly and easily transform a huge set of data into easy-to-navigate pages with just a few lines of code. Really.

3. The Paginator

This example will be composed of two scripts: the reusable paginator class and the index file that will display the table items and controls.

Paginator.class.php

The paginator class will have only two methods and the constructor. We will build it gradually, explaining each step as we move forward.

```
<?php

class Paginator {

    private $_conn;
        private $_limit;
        private $_page;
        private $_query;
        private $_total;

}
```

This definition only sets the paginator required member variables. Since this is a helper class and it's destined for pagination only, it will rely on a valid connection to the MySQL server and an already defined query, to which we will append the parameters necessary to paginate the results. We'll start with the constructor method.

```
<?php

public function __construct( $conn, $query ) {

    $this->_conn = $conn;
    $this->_query = $query;

    $rs= $this->_conn->query( $this->_query );
    $this->_total = $rs->num_rows;

}
```

Quite simple, right? This method only sets the object's database connection and the necessary query. After that, it calculates the total number of rows retrieved by that query without any limit or skip parameters. This total is necessary to create the links for the paginator.

Note that, for simplicity, we are not doing error checking or any other validation of the given parameters, but in a real-world application, these checks will be necessary.

Retrieving Results

Now, let's create the method that will actually paginate the data and return the results.

```php
<?php
public function getData( $limit = 10, $page = 1 ) {

    $this->_limit   = $limit;
    $this->_page    = $page;

    if ( $this->_limit == 'all' ) {
        $query   = $this->_query;
    } else {
        $query   = $this->_query . " LIMIT " . ( ( $this->_page - 1 ) * $this->_limit ) . ", $this->_limit";
    }
    $rs     = $this->_conn->query( $query );

    while ( $row = $rs->fetch_assoc() ) {
        $results[]  = $row;
    }

    $result    = new stdClass();
    $result->page   = $this->_page;
    $result->limit  = $this->_limit;
    $result->total  = $this->_total;
    $result->data   = $results;

    return $result;
}
```

Let's analyze this one step at a time. First, we set the limit and page parameters, which by default are set to 10 and 1 respectively. Then, we check if the user requires a given number of rows or all of them. Based on this and the page parameter, we set the `LIMIT` term of the query. Note that we take one from the page number because our script counts from 1 instead of from 0.

After this, we simply evaluate the query and get the results. Finally, we create a new results object which contains the limit, page, and total parameters of the executed query, as well as the data for each of the retrieved rows.

Displaying Pagination Links

Now, let's write the method used to get the pagination links.

```
<?php
public function createLinks( $links, $list_class ) {
    if ( $this->_limit == 'all' ) {
        return '';
    }

    $last    = ceil( $this->_total / $this->_limit );

    $start   = ( ( $this->_page - $links ) > 0 ) ? $this->_page -
$links : 1;
    $end     = ( ( $this->_page + $links ) < $last ) ? $this->_page
+ $links : $last;

    $html    = '<ul class="' . $list_class . '">';

    $class   = ( $this->_page == 1 ) ? "disabled" : "";
    $html    .= '<li class="' . $class . '"><a href="?limit=' .
$this->_limit . '&page=' . ( $this->_page - 1 ) .
'">&laquo;</a></li>';

    if ( $start > 1 ) {
        $html  .= '<li><a href="?limit=' . $this->_limit .
'&page=1">1</a></li>';
        $html  .= '<li class="disabled"><span>...</span></li>';
    }

    for ( $i = $start ; $i <= $end; $i++ ) {
        $class = ( $this->_page == $i ) ? "active" : "";
        $html  .= '<li class="' . $class . '"><a href="?limit=' .
$this->_limit . '&page=' . $i . '">' . $i . '</a></li>';
    }

    if ( $end < $last ) {
        $html  .= '<li class="disabled"><span>...</span></li>';
        $html  .= '<li><a href="?limit=' . $this->_limit .
'&page=' . $last . '">' . $last . '</a></li>';
    }

    $class   = ( $this->_page == $last ) ? "disabled" : "";
    $html    .= '<li class="' . $class . '"><a href="?limit=' .
$this->_limit . '&page=' . ( $this->_page + 1 ) .
'">&raquo;</a></li>';

    $html    .= '</ul>';

    return $html;
}
```

This is a rather long method, so let's look over it in more detail.

First, we evaluate if the user requires a given number of links or all of them. If the user is requesting all links, then we simply return an empty string, since no pagination is required.

After this, we calculate the last page based on the total number of rows available and the items required per page.

Then, we take the links parameter, which represents the number of links to display below and above the current page, and calculate the start and end link to create.

Next, we create the opening tag for the list and set the class of it with the list class parameter. We also add the "previous page" link—note that for this link, we check if the current page is the first, and if so, we set the disabled property of the link. We also display a link to the first page and an ellipsis symbol in case the start link is not the first one.

Next, we add the links below and above the current page, based on the previously calculated start and end parameters. In each step, we evaluate the current page against the link page displayed and set the active class accordingly.

After this, we display another ellipsis symbol and the link to the last page, in case the end link is not the last one.

Finally, we display the "next page" link and set the disabled state when the user is viewing the last page, close the list, and return the generated HTML string.

That's all there is to the `Paginator` class! Of course, we could add setters and getters for the database connection, limit, page, query, and total parameters, but for simplicity we'll keep it this way.

4. The Index Page

Now we'll create the **index.php** file, which is in charge of using the `Paginator` class and displaying the data. First, let me show you the base HTML.

```
<!DOCTYPE html>
    <head>
        <title>PHP Pagination</title>
        <link rel="stylesheet" href="css/bootstrap.min.css">
    </head>
    <body>
        <div class="container">
                <div class="col-md-10 col-md-offset-1">
                <h1>PHP Pagination</h1>
                <table class="table table-striped table-condensed
table-bordered table-rounded">
                        <thead>
                                <tr>
                                <th>City</th>
                                <th width="20%">Country</th>
                                <th width="20%">Continent</th>
                                <th width="25%">Region</th>
                        </tr>
                        </thead>
                        <tbody></tbody>
                </table>
                </div>
        </div>
        </body>
</html>
```

Quite simple—this file only displays a table that we will populate with the information retrieved from the database. Note that for this example I'm using Bootstrap for basic page styling.

Using the Paginator

```
<?php for( $i = 0; $i < count( $results->data ); $i++ ) : ?>
        <tr>
                <td><?php echo $results->data[$i]['Name']; ?></td>
                <td><?php echo $results->data[$i]['Country']; ?
></td>
                <td><?php echo $results->data[$i]['Continent']; ?
></td>
                <td><?php echo $results->data[$i]['Region'];
?></td>
        </tr>
<?php endfor; ?>
```

Now, to make use of our `Paginator` class, add the following PHP code at the top of the document.

```
<?php
    require_once 'Paginator.class.php';

    $conn   = new mysqli( '127.0.0.1', 'root', 'root', 'world' );

    $limit   = ( isset( $_GET['limit'] ) ) ? $_GET['limit'] : 25;
    $page   = ( isset( $_GET['page'] ) ) ? $_GET['page'] : 1;
    $links   = ( isset( $_GET['links'] ) ) ? $_GET['links'] : 7;
    $query   = "SELECT City.Name, City.CountryCode, Country.Code,
Country.Name AS Country, Country.Continent, Country.Region FROM
City, Country WHERE City.CountryCode = Country.Code";

    $Paginator = new Paginator( $conn, $query );

    $results  = $Paginator->getData( $limit, $page );
?>
```

This script is quite simple—we just required our `Paginator` class. Note that this code assumes that this file is in the same directory as the **index.php** file—if this is not the case, you should update the path accordingly.

Then, we create the connection to our database using the MySQLi library, retrieve the paginator parameters from the GET request, and set the query. Since this is not an article on MySQL, I will not get into details about the connection or the query used here.

Lastly, we create the `Paginator` object and retrieve the results for the current page.

Displaying the Results

Now, to display the obtained results, add the following code to the table body.

```
<?php for( $i = 0; $i < count( $results->data ); $i++ ) : ?>
        <tr>
                <td><?php echo $results->data[$i]['Name']; ?></td>
                <td><?php echo $results->data[$i]['Country']; ?
></td>
                <td><?php echo $results->data[$i]['Continent']; ?
></td>
                <td><?php echo $results->data[$i]['Region'];
?></td>
        </tr>
<?php endfor; ?>
```

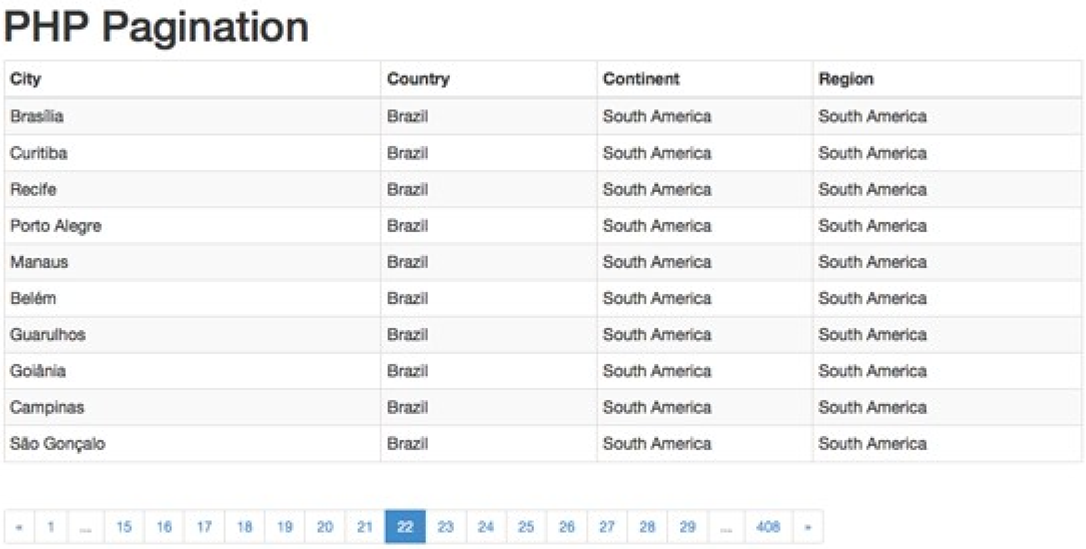
Here, we are simply iterating through the results data attribute containing the city records and creating a table row for each one of them.

Pagination Links

Now, to display the paginator links, add the following code below the table.

```
<?php echo $Paginator->createLinks( $links, 'pagination
 pagination-sm' ); ?>
```

To the paginator `createLinks` method, we pass the obtained `links` parameter and the CSS class for the pagination links used from Bootstrap. Here is the result of the created page.



PHP Pagination

City	Country	Continent	Region
Brasília	Brazil	South America	South America
Curitiba	Brazil	South America	South America
Recife	Brazil	South America	South America
Porto Alegre	Brazil	South America	South America
Manaus	Brazil	South America	South America
Belém	Brazil	South America	South America
Guarulhos	Brazil	South America	South America
Goiânia	Brazil	South America	South America
Campinas	Brazil	South America	South America
São Gonçalo	Brazil	South America	South America

How to Implement AJAX-Based Pagination

In this section, we'll quickly go through how you could convert the above PHP pagination example into AJAX-based pagination. We'll revise all the files and discuss the changes that we will need to make.

Firstly, let's update the **index.php** file, as shown in the following snippet.

```
<!DOCTYPE html>
    <head>
        <title>PHP Pagination</title>
        <link rel="stylesheet" href="css/bootstrap.min.css">
    </head>

    <body>
        <div class="container">
                <div class="col-md-10 col-md-offset-1">
                    <h1>PHP Pagination</h1>
                    <div id="ajax_wrapper">
                        <?php
                            require_once 'ajax_pagination.php';
                        ?>
                    </div>
                </div>
        </div>
        <script src="https://code.jquery.com/jquery-3.5.1.min.js"
integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0="
crossorigin="anonymous"></script>
```

```
        <script src="pagination.js"></script>
    </body>
</html>
```

First of all, we've moved the HTML code which is required to build the pagination into the **ajax_pagination.php** file so that we can reuse it. Next, we've loaded the jQuery library and **pagination.js** file at the end of the file.

The **ajax_pagination.php** file looks like this.

```
<?php
require_once 'paginator.php';

$conn   = new mysqli( '127.0.0.1', 'root', 'root', 'world' );

$limit   = ( isset( $_GET['limit'] ) ) ? $_GET['limit'] : 25;
$page   = ( isset( $_GET['page'] ) ) ? $_GET['page'] : 1;
$links   = ( isset( $_GET['links'] ) ) ? $_GET['links'] : 7;
$query   = "SELECT City.Name, City.CountryCode, Country.Code,
Country.Name AS Country, Country.Continent, Country.Region FROM
City, Country WHERE City.CountryCode = Country.Code";

$Paginator = new Paginator( $conn, $query );
$results  = $Paginator->getData( $limit, $page );
?>

<table class="table table-striped table-condensed table-bordered
table-rounded">
    <thead>
        <tr>
            <th>City</th>
            <th width="20%">Country</th>
            <th width="20%">Continent</th>
            <th width="25%">Region</th>
        </tr>
    </thead>
    <tbody>
    <?php for( $i = 0; $i < count( $results->data ); $i++ ) : ?>
        <tr>
                <td><?php echo $results->data[$i]
['first_name']; ?></td>
                <td><?php echo $results->data[$i]['last_name']; ?
></td>
                <td><?php echo $results->data[$i]['email'];
?></td>
                <td><?php echo $results->data[$i]['phone'];
?></td>
        </tr>
    <?php endfor; ?>
    </tbody>
</table>
```

```
<?php echo $Paginator->createLinks( $links, 'pagination
pagination-sm' ); ?>
```

It's pretty much the same.

Next, let's have a look at the **pagination.js** file.

```
$(document).on( "click", ".pagination a", function(e) {
    var pageValue = $(this).attr("data-page");

    $.ajax({
        url: '/ajax_pagination.php?limit=25&page='+pageValue,
        type: "GET",
        success: function(data){
            $("#ajax_wrapper").html(data);
        }
    });

    e.preventDefault();
});
```

It does a couple of things here. Firstly, it binds the click event to every pagination link. Next, when the link is clicked, it makes an AJAX call to fetch the listing data of the corresponding page and updates the page.

Finally, you need to update the `createLinks` method as shown in the following snippet.

```
public function createLinks( $links, $list_class ) {
    if ( $this->_limit == 'all' ) {
        return '';
    }

    $last   = ceil( $this->_total / $this->_limit );

    $start   = ( ( $this->_page - $links ) > 0 ) ? $this->_page -
$links : 1;
    $end   = ( ( $this->_page + $links ) < $last ) ? $this->_page
+ $links : $last;

    $html   = '<ul class="' . $list_class . '">';

    $class   = ( $this->_page == 1 ) ? "disabled" : "";
    $html   .= '<li class="' . $class . '"><a data-page="' .
( $this->_page - 1 ) . '" href="?limit=' . $this->_limit .
'&page=' . ( $this->_page - 1 ) . '">&laquo;</a></li>';

    if ( $start > 1 ) {
        $html .= '<li><a data-page="1" href="?limit=' . $this-
>_limit . '&page=1">1</a></li>';
        $html .= '<li class="disabled"><span>...</span></li>';

    }

    for ( $i = $start ; $i <= $end; $i++ ) {
        $class = ( $this->_page == $i ) ? "active" : "";
        $html .= '<li class="' . $class . '"><a  data-page="' . $i
. '" href="?limit=' . $this->_limit . '&page=' . $i . '">' . $i .
'</a></li>';
    }

    if ( $end < $last ) {
        $html .= '<li class="disabled"><span>...</span></li>';
        $html .= '<li><a data-page="' . $last . '"href="?limit=' .
$this->_limit . '&page=' . $last . '">' . $last . '</a></li>';
    }

    $class   = ( $this->_page == $last ) ? "disabled" : "";
    $html    .= '<li class="' . $class . '"><a data-page="' .
( $this->_page + 1 ) . '" href="?limit=' . $this->_limit .
'&page=' . ( $this->_page + 1 ) . '">&raquo;</a></li>';

    $html    .= '</ul>';

    return $html;
}
```

Basically, we've added the `data-page` attribute to every link, so that we can get the page number when the user clicks on the link.

And with these changes in place, you've turned your pagination into AJAX-based pagination!

Conclusion

This should provide you with everything that you need to know in order to get up and running with pagination in your application.

Learn PHP With a Free Online Course

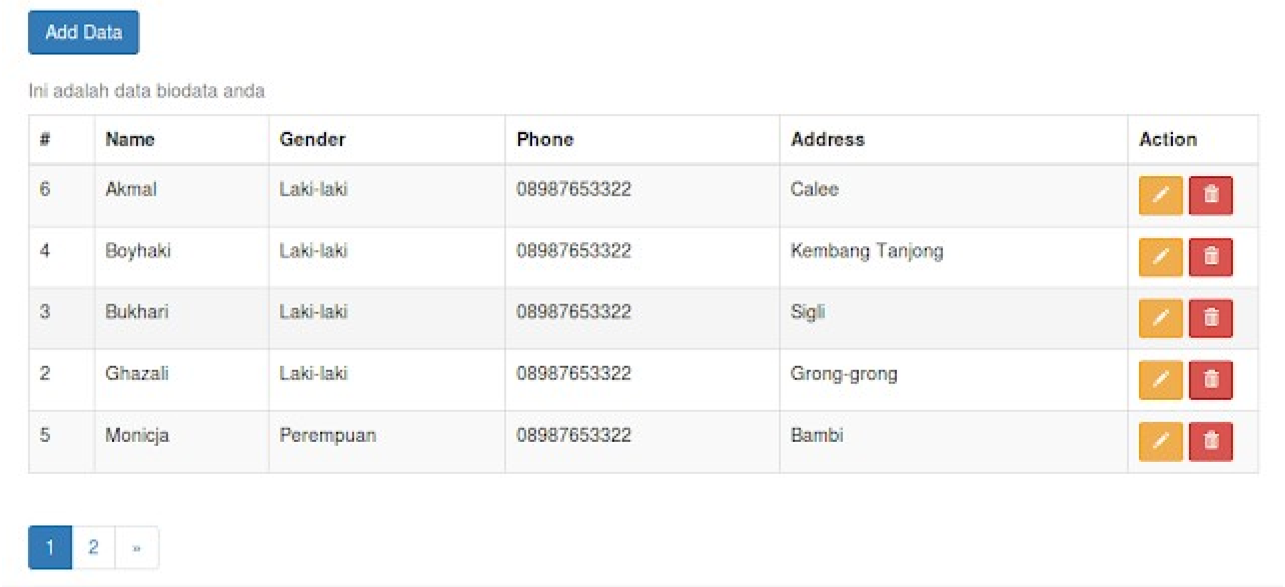
If you want to learn PHP, check out our free online course on PHP fundamentals!

In this course, you'll learn the fundamentals of PHP programming. You'll start with the basics, learning how PHP works and writing simple PHP loops and functions. Then you'll build up to coding classes for simple object-oriented programming (OOP). Along the way, you'll learn all the most important skills for writing apps for the web: you'll get a chance to practice responding to GET and POST requests, parsing JSON, authenticating users, and using a MySQL database.

https://code.tutsplus.com/tutorials/how-to-paginate-data-with-php--net-2928

CRUD (Create, Read, Update, Delete, Pagination) Using PHP PDO and Bootstrap

00:34 Php pdo



Ini adalah data biodata anda

#	Name	Gender	Phone	Address	Action
6	Akmal	Laki-laki	08987653322	Calee	✎ 🗑
4	Boyhaki	Laki-laki	08987653322	Kembang Tanjong	✎ 🗑
3	Bukhari	Laki-laki	08987653322	Sigli	✎ 🗑
2	Ghazali	Laki-laki	08987653322	Grong-grong	✎ 🗑
5	Monicja	Perempuan	08987653322	Bambi	✎ 🗑

In this article I will discuss how to create an application CRUD (Create, Read, Update, Delete and Pagination) using pdo php and mysql as the database server. To make the application php pdo we should not have to use mysql but we can use other dbms like postgresql or sqlite. In this tutorial I use the database from mysql because I 'm used to using mysql. If you are not using mysql then you just compose the connection just become your favorite dbms.

To make sure the webserver application in your computer is already live. To use better use apache2 webserver or xampp. If you do not like the xampp you can use other web server such as IIS and others. Create a folder inside the webroot and with the name "crudpdo" or up to you. Then create a new database and a new table or as syntax sql below.

```
--
-- Database: `biodata`
--

CREATE TABLE IF NOT EXISTS `crudpdo` (
`id_pdo` int(11) NOT NULL COMMENT 'Identitas',
  `nm_pdo` varchar(45) NOT NULL COMMENT 'Nama',
  `gd_pdo` varchar(20) NOT NULL COMMENT 'Jenis Kelamin',
  `tl_pdo` varchar(25) NOT NULL COMMENT 'Phone',
  `ar_pdo` text NOT NULL COMMENT 'Alamat'
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `crudpdo`
```

```
 ADD PRIMARY KEY (`id_pdo`);

ALTER TABLE `crudpdo`
MODIFY `id_pdo` int(11) NOT NULL AUTO_INCREMENT COMMENT 'Identitas';
```

Download and extract the plugin bootstrap into the folder "crudpdo". So there will be a folder css, js, and fonts. Then create a new folder named "includes" which contains the config.php file, data.inc.php and pagination.inc.php or like syntax below.

config.php

```
<?php
class Config{

 // specify your own database credentials
 private $host = "localhost";
 private $db_name = "biodata";
 private $username = "root";
 private $password = "pidie";
 public $conn;

 // get the database connection
 public function getConnection(){

  $this->conn = null;

  try{
   $this->conn = new PDO("mysql:host=" . $this->host . ";dbname=" . $this->db_name, $this-
>username, $this->password);
  }catch(PDOException $exception){
   echo "Connection error: " . $exception->getMessage();
  }

  return $this->conn;
 }
}
?>
```

data.inc.php

```
<?php
class Data{

 // database connection and table name
 private $conn;
 private $table_name = "crudpdo";
```

```
// object properties
public $id;
public $nm;
public $gd;
public $tl;
public $ar;

public function __construct($db){
 $this->conn = $db;
}

// create product
function create(){

 //write query
 $query = "INSERT INTO " . $this->table_name . " values(",?,?,?,?)";

 $stmt = $this->conn->prepare($query);

 $stmt->bindParam(1, $this->nm);
 $stmt->bindParam(2, $this->gd);
 $stmt->bindParam(3, $this->tl);
 $stmt->bindParam(4, $this->ar);

 if($stmt->execute()){
  return true;
 }else{
  return false;
 }

}

// read products
function readAll($page, $from_record_num, $records_per_page){

 $query = "SELECT
    *
   FROM
    " . $this->table_name . "
   ORDER BY
    nm_pdo ASC
   LIMIT
    {$from_record_num}, {$records_per_page}";

 $stmt = $this->conn->prepare( $query );
 $stmt->execute();

 return $stmt;
}

// used for paging products
public function countAll(){

 $query = "SELECT id_pdo FROM " . $this->table_name . "";

 $stmt = $this->conn->prepare( $query );
 $stmt->execute();

 $num = $stmt->rowCount();

 return $num;
}

// used when filling up the update product form
function readOne(){

 $query = "SELECT
    *
   FROM
    " . $this->table_name . "
   WHERE
    id_pdo = ?
   LIMIT
    0,1";

 $stmt = $this->conn->prepare( $query );
 $stmt->bindParam(1, $this->id);
 $stmt->execute();

 $row = $stmt->fetch(PDO::FETCH_ASSOC);

 $this->nm = $row['nm_pdo'];
 $this->gd = $row['gd_pdo'];
 $this->tl = $row['tl_pdo'];
 $this->ar = $row['ar_pdo'];
}

// update the product
function update(){

 $query = "UPDATE
```

```
     " . $this->table_name . "
    SET
     nm_pdo = :nm,
     gd_pdo = :gd,
     tl_pdo = :tl,
     ar_pdo = :ar
    WHERE
     id_pdo = :id";

  $stmt = $this->conn->prepare($query);

  $stmt->bindParam(':nm', $this->nm);
  $stmt->bindParam(':gd', $this->gd);
  $stmt->bindParam(':tl', $this->tl);
  $stmt->bindParam(':ar', $this->ar);
  $stmt->bindParam(':id', $this->id);

  // execute the query
  if($stmt->execute()){
   return true;
  }else{
   return false;
  }
 }

 // delete the product
 function delete(){

  $query = "DELETE FROM " . $this->table_name . " WHERE id_pdo = ?";

  $stmt = $this->conn->prepare($query);
  $stmt->bindParam(1, $this->id);

  if($result = $stmt->execute()){
   return true;
  }else{
   return false;
  }
 }
}
?>
```

pagination.inc.php

```
<?php
// the page where this paging is used
```

```
echo "<nav><ul class=\"pagination\">";

// button for first page
if($page>1){
    echo "<li><a href='{$page_dom}' title='Go to the first page.'>";
        echo "«";
    echo "</a></li>";
}

// count all products in the database to calculate total pages
$total_rows = $product->countAll();
$total_pages = ceil($total_rows / $records_per_page);

// range of links to show
$range = 2;

// display links to 'range of pages' around 'current page'
$initial_num = $page - $range;
$condition_limit_num = ($page + $range)  + 1;

for ($x=$initial_num; $x<$condition_limit_num; $x++) {

    // be sure '$x is greater than 0' AND 'less than or equal to the $total_pages'
    if (($x > 0) && ($x <= $total_pages)) {

        // current page
        if ($x == $page) {
            echo "<li class='active'><a href=\"#\">$x</a></li>";
        }

        // not current page
        else {
            echo "<li><a href='{$page_dom}?page=$x'>$x</a></li>";
        }
    }
}

// button for last page
if($page<$total_pages){
    echo "<li><a href='" .$page_dom . "?page={$total_pages}' title='Last page is {$total_pages}.'>";
        echo "»";
    echo "</a></li>";
}

echo "</ul></nav>";
```

```
?>
```

Then go back to the folder "crudpdo" and create files like index.php, add.php, update.php and delete.php. This file is a file that contains the template files from the bootstrap and a crud application. For syntax as below.

index.php

```
<?php
$page = isset($_GET['page']) ? $_GET['page'] : 1;

$records_per_page = 5;

$from_record_num = ($records_per_page * $page) - $records_per_page;

include_once 'includes/config.php';
include_once 'includes/data.inc.php';

$database = new Config();
$db = $database->getConnection();

$product = new Data($db);

$stmt = $product->readAll($page, $from_record_num, $records_per_page);
$num = $stmt->rowCount();

?>
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Data CRUD PDO</title>

    <!-- Bootstrap -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
      <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
    <![endif]-->
  </head>
  <body>
  <p>
</p>
    <div class="container">
      <p>
 <a class="btn btn-primary" href="add.php" role="button">Add Data</a>
      </p>
<?php
if($num>0){
?>
 <table class="table table-bordered table-hover table-striped">
 <caption>Ini adalah data biodata anda</caption>
 <thead>
  <tr>
          <th>#</th>
          <th>Name</th>
          <th>Gender</th>
          <th>Phone</th>
          <th>Address</th>
          <th>Action</th>
        </tr>
 </thead>
 <tbody>
<?php
while ($row = $stmt->fetch(PDO::FETCH_ASSOC)){
extract($row);
?>
<tr>
 <?php echo "<td>{$id_pdo}</td>" ?>
 <?php echo "<td>{$nm_pdo}</td>" ?>
 <?php echo "<td>{$gd_pdo}</td>" ?>
 <?php echo "<td>{$tl_pdo}</td>" ?>
 <?php echo "<td>{$ar_pdo}</td>" ?>
 <?php echo "<td width='100px'>
    <a class='btn btn-warning btn-sm' href='update.php?id={$id_pdo}' role='button'><span
class='glyphicon glyphicon-pencil' aria-hidden='true'></span></a>
    <a class='btn btn-danger btn-sm' href='delete.php?id={$id_pdo}' role='button'><span
class='glyphicon glyphicon-trash' aria-hidden='true'></span></a>
          </td>" ?>
</tr>
<?php
}
?>
 </tbody>
      </table>
<?php
$page_dom = "index.php";
include_once 'includes/pagination.inc.php';
}
else{
?>
<div class="alert alert-warning alert-dismissible" role="alert">
  <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">×</span></button>
  <strong>Warning!</strong> Data Masih Kosong Tolong Diisi.
</div>
<?php
}
?>
    </div>

    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="js/jquery.min.js"></script>
    <!-- Include all compiled plugins (below), or include individual files as needed -->
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

add.php

```
<?php
include_once 'includes/config.php';

$database = new Config();
$db = $database->getConnection();

include_once 'includes/data.inc.php';
$product = new Data($db);
?>
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Data CRUD PDO</title>

    <!-- Bootstrap -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
```

```
      <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
      <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
    <![endif]-->
  </head>
  <body>
  <p>
</p>
    <div class="container">
      <p>
 <a class="btn btn-primary" href="index.php" role="button">Back View Data</a>
      </p>

<?php
if($_POST){

 $product->nm = $_POST['nm'];
 $product->gd = $_POST['gd'];
 $product->tl = $_POST['tl'];
 $product->ar = $_POST['ar'];

 if($product->create()){
?>
<div class="alert alert-success alert-dismissible" role="alert">
  <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">×</span></button>
  <strong>Success!</strong> Anda Berhasil, <a href="index.php">View Data</a>.
</div>
<?php
 }else{
?>
<div class="alert alert-danger alert-dismissible" role="alert">
  <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">×</span></button>
  <strong>Fail!</strong> Anda Gagal, Coba Lagi.
</div>
<?php
 }
}
?>
<form method="post">
  <div class="form-group">
    <label for="nm">Name</label>
    <input type="text" class="form-control" id="nm" name="nm">
  </div>
  <div class="form-group">
    <label for="gd">Gender</label>
    <input type="text" class="form-control" id="gd" name="gd">
  </div>
  <div class="form-group">
    <label for="tl">Phone</label>
    <input type="text" class="form-control" id="tl" name="tl">
  </div>
  <div class="form-group">
    <label for="ar">Alamat</label>
    <textarea class="form-control" rows="3" id="ar" name="ar"></textarea>
  </div>
  <button type="submit" class="btn btn-success">Submit</button>
</form>
    </div>

    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="js/jquery.min.js"></script>
    <!-- Include all compiled plugins (below), or include individual files as needed -->
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

update.php

```
<?php
include_once 'includes/config.php';

$id = isset($_GET['id']) ? $_GET['id'] : die('ERROR: missing ID.');

$database = new Config();
$db = $database->getConnection();

include_once 'includes/data.inc.php';
$product = new Data($db);

$product->id = $id;
$product->readOne();
?>
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Data CRUD PDO</title>

    <!-- Bootstrap -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
      <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
    <![endif]-->
  </head>
  <body>
  <p>
</p>
    <div class="container">
      <p>
 <a class="btn btn-primary" href="index.php" role="button">Back View Data</a>
      </p>

<?php
if($_POST){

 $product->nm = $_POST['nm'];
 $product->gd = $_POST['gd'];
 $product->tl = $_POST['tl'];
 $product->ar = $_POST['ar'];

 if($product->update()){
?>
<script>window.location.href='index.php'</script>
<?php
 }else{
?>
<div class="alert alert-danger alert-dismissible" role="alert">
  <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-
hidden="true">×</span></button>
  <strong>Fail!</strong> Anda Gagal, Coba Lagi.
</div>
<?php
 }
}
?>
<form method="post">
  <div class="form-group">
    <label for="nm">Name</label>
    <input type="text" class="form-control" id="nm" name="nm" value='<?php echo $product->nm;
?>'>
  </div>
  <div class="form-group">
    <label for="gd">Gender</label>
    <input type="text" class="form-control" id="gd" name="gd" value='<?php echo $product->gd; ?
>'>
  </div>
  <div class="form-group">
    <label for="tl">Phone</label>
    <input type="text" class="form-control" id="tl" name="tl" value='<?php echo $product->tl; ?>'>
  </div>
  <div class="form-group">
    <label for="ar">Alamat</label>
    <textarea class="form-control" rows="3" id="ar" name="ar"><?php echo $product->ar;
?></textarea>
  </div>
  <button type="submit" class="btn btn-success">Submit</button>
</form>
    </div>

    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="js/jquery.min.js"></script>
    <!-- Include all compiled plugins (below), or include individual files as needed -->
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

delete.php

```
// check if value was posted
// include database and object file
 include_once 'includes/config.php';
 include_once 'includes/data.inc.php';

 // get database connection
 $database = new Config();
 $db = $database->getConnection();

 // prepare product object
 $product = new Data($db);

 // set product id to be deleted
 $product->id = isset($_GET['id']) ? $_GET['id'] : die('ERROR: missing ID.');

 // delete the product
 if($product->delete()){
  echo "<script>location.href='index.php'</script>";
 }
 // if unable to delete the product
 else{
  echo "<script>alert('Gagal menghapus data')</script>";

 }
?>
```

Complete, run in your favorite browser such as Mozilla Firefox, Google Chrome, Opera web browser, Apple Safari and Microsoft Internet Explorer.
http://tutoriallancer.blogspot.com/2016/06/crud-create-read-update-delete.html