

LAPORAN PRATIKUM
PEMROGRAMAN WEB DAN MOBILE I



NAMA : Hizbullah Haidar A A
NIM : 193010503011
MODUL : III
KELAS : A

JURUSAN/PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2021

BAB I

TUJUAN DAN LANDASAN TEORI

1. Tujuan Praktikum

- 1.1. Mahasiswa mampu membuat program yang bisa menyimpan data dalam jumlah yang banyak.
- 1.2. Mahasiswa mampu membuat program yang bisa mengolah data yang tersimpan dalam database.

2. Landasan Teori

Data dalam database MySQL disimpan dalam tabel-tabel. Sebuah tabel adalah koleksi dari data yang berelasi dan mengandung kolom dan baris. Database sangat bermanfaat untuk menyimpan informasi secara kategori. Contoh yang akan diberikan pada modul praktikum ini adalah tabel yang mengandung data Employees (Pekerja), Products (Produk), Customers (Pelanggan) dan Orders (Pesanan).

Membuka Koneksi

Sebelum mengakses data dalam database MySQL, kita harus terhubung ke server database MySQL. Berikut adalah contoh kode program agar terhubung dengan server mySQL:

```
<?php
    $servername = "localhost";
    $username = "username";
    $password = "password";

    // Membuat hubungan
    $conn = new mysqli($servername, $username, $password);

    // Memeriksa hubungan
    if ($conn->connect_error){
        die("Connection failed: " . $conn->connect_error);
    }
    echo "Connected successfully";
?>
```

Gambar 1.1 Koneksi

Jika kode diatas tidak berhasil, kemungkinan variabel \$connect_error sudah tidak tersedia pada versi PHP yang kita gunakan, maka gunakan kode program berikut:

```
<?php

$servername = "localhost";
$username = "username";
$password = "password";

// Membuat Hubungan
$conn = mysqli_connect($servername, $username, $password);

// Memeriksa
Hubungan if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}

echo "Connected successfully";

?>
```

Gambar 1.2 Koneksi

Ketika selesai menggunakan data dari database, sebaiknya koneksi atau hubungan ke server ditutup, caranya dengan menggunakan kode program berikut: **mysqli_close(\$conn);**

Membuat Database

Database pada MySQL bisa juga dibuat menggunakan kode program PHP. Program tersebut akan berisi statement SQL “CREATE DATABASE”. Berikut adalah contoh program yang digunakan untuk membuat database “myDB”:

```

<?php
    $servername = "localhost";
    $username = "username";
    $password = "password";

    // Membuat Koneksi
    $conn = mysqli_connect($servername, $username, $password);
    // Memeriksa Koneksi
    if (!$conn){
        die("Connection failed: " . mysqli_connect_error());
    }

    // Membuat Database
    $sql = "CREATE DATABASE myDB";
    if (mysqli_query($conn, $sql)) {

```

Gambar 1,3 Membuat Database

```

        echo "Database created successfully";
    } else {
        echo "Error creating database: " . mysqli_error($conn);
    }

    mysqli_close($conn);
?>

```

Gambar 1.4 Membuat Database

Membuat Tabel

Pembuatan tabel pada bahasa pemrograman PHP juga menggunakan statement SQL, yaitu statement “CREATE TABLE”. Contoh yang akan diberikan adalah pembuatan tabel MyGuests. Statement pembuatan tabel MyGuests adalah sebagai berikut:

```

CREATE TABLE MyGuests (
    id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    firstname VARCHAR(30) NOT NULL,
    lastname VARCHAR(30) NOT NULL,
    email VARCHAR(50),
    reg_date TIMESTAMP )

```

Berikut adalah contoh program untuk membuat tabel diatas:

```

<?php
    $servername = "localhost";
    $username = "username";
    $password = "password";
    $dbname = "myDB";

    // Membuat Koneksi
    $conn = mysqli_connect($servername, $username, $password, $dbname);
    // Memeriksa koneksi
    if (!$conn){
        die("Connection failed: " . mysqli_connect_error());
    }

    // SQL untuk membuat tabel
    $sql = "CREATE TABLE MyGuests (

```

Gambar 1.5 Membuat Tabel

```

        id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY
        KEY, firstname VARCHAR(30) NOT NULL, lastname
        VARCHAR(30) NOT NULL,
        email VARCHAR(50),
        reg_date TIMESTAMP
    );

    if (mysqli_query($conn, $sql)) {
        echo "Table MyGuests created successfully";
    } else {
        echo "Error creating table: " . mysqli_error($conn);
    }

    mysqli_close($conn);
?>

```

Gambar 1.6 Membuat Tabel

Memasukkan Data ke Database

Untuk memasukkan data kedalam tabel di database, terdapat beberapa aturan sintaks yang harus diikuti:

- Query SQL harus diberikan kutip dalam PHP.
- Nilai string didalam query SQL harus diberikan kutip.
- Nilai numeris tidak harus diberikan kutip.
- Nilai NULL tidak harus diberikan kutip.

Statement INSERT INTO digunakan untuk menambahkan baris pada tabel MySQL, query untuk menambahkan data tersebut adalah sebagai berikut:

INSERT INTO table_name (column1, column2, column3,...) VALUES (value1, value2, value3,...)

adapun contoh kode program PHP untuk menambahkan data tersebut adalah sebagai berikut:

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

$dbname = "myDB";

// Membuat Koneksi
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Memeriksa Koneksi
if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

Gambar 1.7 Memasukan Data

Jika kita melakukan perintah INSERT pada tabel yang menggunakan field dengan opsi AUTO_INCREMENT, kita bisa mendapatkan ID dari baris yang terakhir diinputkan. Caranya menggunakan fungsi `mysqli_insert_id($conn)`; nilai kembalian dari fungsi ini akan memberikan id dari record atau baris terakhir yang diinputkan.

Untuk menambahkan record dengan jumlah yang banyak, kita bisa menggunakan fungsi `mysqli_multi_query($conn, $sql)`, berikut adalah contoh program untuk menambahkan record dengan jumlah yang banyak:

```

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Membuat Koneksi
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Memeriksa Koneksi
if (!$conn){

```

Gambar 1.8 Memasukan Data

```

        die("Connection failed: " . mysqli_connect_error());
    }

    $sql = "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('John', 'Doe', 'john@example.com');";
    $sql .= "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('Mary', 'Moe', 'mary@example.com');";
    $sql .= "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('Julie', 'Dooley', 'julie@example.com')";

    if (mysqli_multi_query($conn, $sql)) {
        echo "New records created successfully";
    } else {
        echo "Error: " . $sql . "<br>" . mysqli_error($conn);
    }

    mysqli_close($conn);
?>

```

Gambar 1.9 Memasukan Data

Mengambil Data dari

Database Untuk mengambil data, statement SQL yang digunakan adalah SELECT nama_kolom() FROM nama_tabel, atau kita bisa menggunakan karakter * untuk memilih semua kolom yang ada pada tabel. Pengambilan data dengan kriteria tertentu bisa dilakukan dengan menggunakan statement WHERE setelah nama_tabel. Adapun contoh kode program untuk mengambil data dari database adalah sebagai berikut:

```

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Membuat koneksi
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Memeriksa Koneksi
if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}

```

Gambar 1.10 Mengambil Data

```

$sql = "SELECT id, firstname, lastname FROM
MyGuests"; $result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
    // Menampilkan data pada setiap baris
    while($row = mysqli_fetch_assoc($result)) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " .
$row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}

mysqli_close($conn);
?>

```

Gambar 1.11 Mengambil Data

Menghapus Data

dari Database Statement DELETE digunakan untuk menghapus baris data atau records dari tabel. Clause WHERE digunakan untuk menspesifikasikan baris yang akan dihapus. Jika statement DELETE digunakan tanpa menggunakan clausa WHERE, maka semua record yang ada pada tabel akan dihapus.

Jika kita memiliki data seperti tabel “MyGuest” seperti terlihat diatas, kemudian kita ingin menghapus data dengan nama depan Julie, maka kode program untuk menghapus data tersebut adalah sebagai berikut:


```

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Membuat Koneksi
$conn = mysqli_connect($servername, $username, $password, $dbname);

```

Gambar 1.12 Menghapus

```

// Memeriksa koneksi
if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}

// sql untuk menghapus record
$sql = "DELETE FROM MyGuests WHERE id=3";

if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}

mysqli_close($conn);
?>

```

Gambar 1.13 Menghapus

Update Data dalam Database

Untuk melakukan perubahan data di dalam database, statement UPDATE digunakan, yaitu sebagai berikut:

UPDATE nama_tabel

SET kolom1=nilai1, kolom2=nilai2, ...

WHERE kolom_penentu=nilai_penentu ...

Jika statement ini digunakan untuk update data ke dua pada tabel “MyGuest” diatas untuk mengganti nama belakang dari “Moe” menjadi “Doe”, maka kode program untuk update tersebut adalah sebagai berikut:

```

<?php
    $servername = "localhost";
    $username = "username";
    $password = "password";
    $dbname = "myDB";

    // Membuat Koneksi
    $conn = mysqli_connect($servername, $username, $password, $dbname);
    // Memeriksa Koneksi
    if (!$conn){

```

Gambar 1.4 Update Data

```

        die("Connection failed: " . mysqli_connect_error());
    }

    $sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

    if (mysqli_query($conn, $sql)) {
        echo "Record updated successfully";
    } else {
        echo "Error updating record: " . mysqli_error($conn);
    }

    mysqli_close($conn);
?>

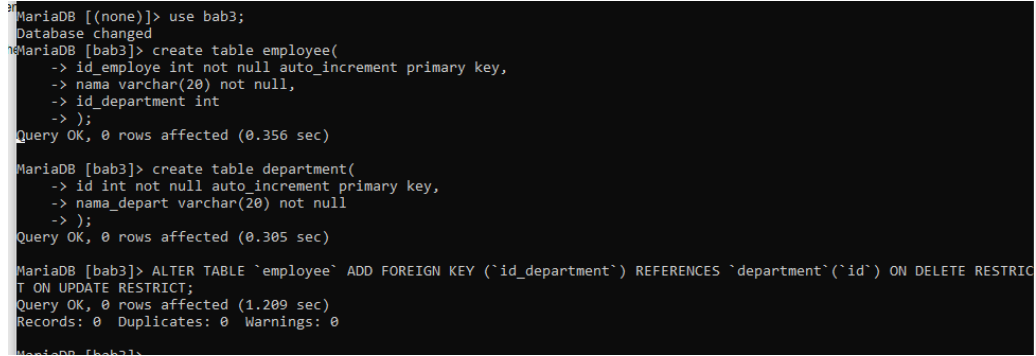
```

Gambar 1.5 Update Data

BAB II

PEMBAHASAN

2.1 Membuat Database Pada MYSQL Data Pegawai Beserta Relasinya



```
MariaDB [(none)]> use bab3;
Database changed
MariaDB [bab3]> create table employee(
-> id_employe int not null auto_increment primary key,
-> nama varchar(20) not null,
-> id_department int
-> );
Query OK, 0 rows affected (0.356 sec)

MariaDB [bab3]> create table department(
-> id int not null auto_increment primary key,
-> nama_depart varchar(20) not null
-> );
Query OK, 0 rows affected (0.305 sec)

MariaDB [bab3]> ALTER TABLE `employee` ADD FOREIGN KEY (`id_department`) REFERENCES `department`(`id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
Query OK, 0 rows affected (1.209 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bab3]>
```

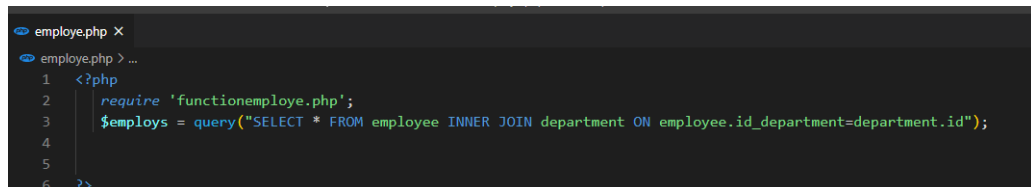
Gambar 2.1 Database

Pada tugas pertama ini membuat database pegawai yang berelasi, dimana dapat di lihat pada gambar 2.1 di atas. Create database bab3 merupakan cara untuk membuat nama database tersebut dan di lanjutkan dengan use bab3 untuk menggunakan database yang bernama bab3,Setelah itu membuat table yang bernama employee dimana mempunyai atribut pertama id_employe bertipe data int,not null,auto_increment dan primary key , yang kedua atribut nama bertipekan varchar(20) dan not null dan yang terakhir atribut id_department yang bertipe data int. Pada atribut id_department merupakan foreign key dari table department.

Selanjutnya membuat table department yang dimana mempunyai dua atribut yaitu atribut id yang mempunyai tipe data int, not null, auto_increment primary key dan atribut nama_depart yang bertipe data varchar(20).

2.2 Membuat program Menyimpan,Mengubah dan Menghapus Data Pegawai

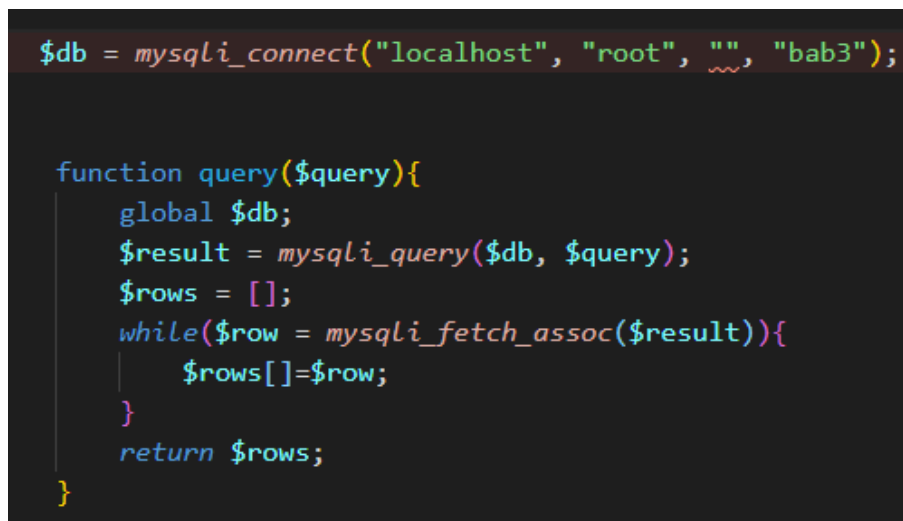
Pada tugas kedua ini membuat program menyimpan , mengubah dan menghapus data pegawai. Pada gambar 2.2 di bawah ini merupakan tahap awal untuk memulai membuat program menampilkan data.



```
employee.php X
employee.php > ...
1  <?php
2      require 'functionemployee.php';
3      $employs = query("SELECT * FROM employee INNER JOIN department ON employee.id_department=department.id");
4
5
6  ?>
```

Gambar 2.2 Menampilkan Data

Require 'functionemployee.php' merupakan untuk menghubungkan file employee.php ke functionemployee.php yang dimana pada file functionemployee.php mempunyai function yang bernama query, Dapat dilihat pada gambar 2.3.



```
$db = mysqli_connect("localhost", "root", "", "bab3");

function query($query){
    global $db;
    $result = mysqli_query($db, $query);
    $rows = [];
    while($row = mysqli_fetch_assoc($result)){
        $rows[]=$row;
    }
    return $rows;
}
```

Gambar 2.3 Function Query

Dapat di lihat gambar 2.3 di atas, function query terdapat mysqli_connect yang dimana fungsi tersebut adalah untuk menghubungkan

ke database dan di berikan variable \$db. Function Query berisikan cara untuk menampilkan data pegawai ke website yang dimana global \$db; untuk membuat variable \$db secara global. Selanjutnya adalah mysqli_query, fungsi ini di gunakan untuk menjalankan intruksi atau argument ke mysql yang mana di beri nama variable \$result. Selanjutnya adalah mysqli_fetch_assoc adalah untuk mengembalikan array dan memberikan nilai balik return. Setelah selesai membuat function query kembali lagi pada gambar 2.1 di atas, terlihat untuk memanggil function tersebut hanya dengan mengetikan query yang beirisikan perintah untuk menampilkan data employee dan di berikan variable employs. Setelah itu untuk menampilkan data employe dapat di lihat pada gambar 2.4 di bawah ini.

```
</a>
<table class="table table-striped"> Add a description to this table.
  <thead>
    <tr>
      <th scope="col">No</th>
      <th scope="col">Nama</th>
      <th scope="col">Department</th>
      <th scope="col">Action</th>
    </tr>
  </thead>

  <tbody>
    <?php $i=1; ?>
    <?php foreach($employs as $employee) : ?>
      <td><?php echo $i; ?></td>
      <td><?php echo $employee["nama"]; ?></td>
      <td><?php echo $employee["nama_depart"]; ?></td>
      <td>
        <a href="employee_ubah.php?id=<?php echo $employee["id_employee"]; ?>">ubah</a> |
        <a href="employee_delete.php?id=<?php echo $employee["id_employee"]; ?>" onclick="return confirm('yakin?');">hapus</a>
      </td>
    </tr>
  </tbody>
</table>
<?php $i++; ?>
<?php endforeach; ?>
```

Gambar 2.4 Foreach

Terlihat pada gambar 2.4 di atas adalah cara untuk menampilkan data employee(data pegawai) yang dimana menggunakan foreach untuk perulangan data yang disimpan di database. Dan hasilnya sebagai berikut

Tambah Data		Ke Halaman Department	
N0	Nama	Department	Action
1	Alex	upr	ubah hapus
2	Udin	upr	ubah hapus

Gambar 2.5 Hasil Menampilkan Data

```

<?php
require 'functionemploye.php';

$departs = query("SELECT * FROM department");

// cek tombol submit sdh ditekan atau belum
if(isset($_POST["submit"])) {
    // cek apakah data berhasil di tambahkan atau tidak
    if (tambah($_POST)>0){
        echo "
        <script>
            alert('data berhasil ditambahkan!');
            document.location.href = 'employe_tambah.php';
        </script>
        ";
    } else {
        echo "
        <script>
            alert('data gagal ditambahkan!');
            document.location.href = 'employe_tambah.php';
        </script>
        ";
    }
}
?>

```

Gambar 2.6 Menambahkan Data

Selanjutnya pada gambar 2.6 di atas merupakan cara untuk menambahkan data yang dimana sama seperti cara untuk menampilkan data, require 'functionemploye.php' merupakan untuk menghubungkan file employe_tambah.php ke functionemploye.php yang dimana pada file functionemploye.php mempunyai function yang bernama tambah. Function tambah dapat di lihat pada gambar 2.7 di bawah ini.

```

function tambah($data){
    global $db;
    // ambil data dari tiap elemen dalam form
    $nama = htmlspecialchars($data["nama"]);
    $id_department = htmlspecialchars($data["id_department"]);

    $query = "INSERT INTO employee
              VALUES
              ('', '$nama', '$id_department')
              ";
    // query insert data
    mysqli_query($db, $query);

    return mysqli_affected_rows($db);
}

```

Gambar 2.7 Function Tambah

Function tambah mempunyai parameter \$data dan berisikan cara untuk menambahkan data employee. global \$db; untuk membuat variable \$db secara global dan selanjutnya htmlspecialchars adalah sebuah fungsi atau sebuah perintah atau syntax yang di miliki oleh PHP yang berguna untuk menonaktifkan seluruh perintah – perintah html mengubah special pada sebuah halaman web. Setelah itu memberikan perintah query untuk menambahkan data.

Kembali lagi pada gambar 2.6 di atas terdapat function query untuk menampilkan data department dan selanjutnya adalah perkondisian apabila data employee berhasil di tambahkan maka terdapat pop up dengan kalimat data berhasil di tambahkan. Sebelum melihat pop up berhasil di tambahkan dapat di lihat pada gambar 2.8 di bawah ini, merupakan sebuah form untuk menginput data employee yang dimana mempunyai method post. Method post merupakan fungsi untuk mengirimkan data.

```

<form action="" method="post">
<ul>
  <li class="mb-2">
    <label for="nama">Nama : </label>
    <input type="text" name="nama" id="nama" required>
  </li>

  <li class="mb-2">
    <div class="form-group">
      <label for="id_department">id_department</label>
      <select name="id_department" class="form-control" id="id_department">
        <option selected>Choose One</option>
        <?php foreach($depart as $depart) : ?>
          <option value="<?= $depart["id"] ?>"><?= $depart["nama_depart"] ?></option>
        <?php endforeach; ?>
      </select>
    </div>
  </li>
</ul>

```

Gambar 2.8 Form Tambah

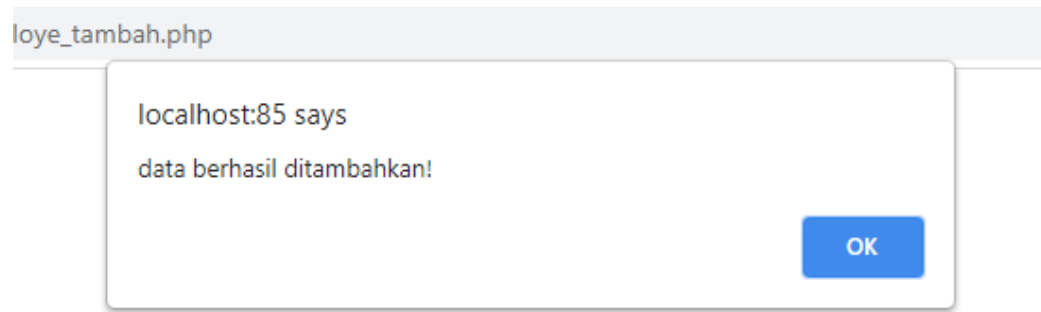
Untuk halaman form tambah dapat di lihat pada gambar 2.9 dan 2.10 di bawah ini yang dimana sekaligus menambahkan data employe dan pop up berhasil.

Tambah Data Employe

[kembali](#)

- Nama :
- id_department
-

Gambar 2.9 Input Data



Gambar 2.10 Pop Up Berhasil

Tambah Data		Ke Halaman Department	
N0	Nama	Department	Action
1	Alex	upr	ubah hapus
2	Udin	upr	ubah hapus
3	akil	ump	ubah hapus

Gambar 2.11 Hasil Menambahkan Data

Setelah berhasil menambahkan data, tugas selanjutnya adalah mengubah data employee. Pada ubah data employee hampir sama seperti nambah data employee yang membedakan nya hanya nama function dan query yang bernama function ubah dan query update. Function Ubah dapat di lihat pada gambar 2.12 di bawah ini.

```

function ubah($data){
    global $db;
    $id_employe = $data["id_employe"];
    // ambil data dari tiap elemen dalam form
    $nama = htmlspecialchars($data["nama"]);
    $id_department = htmlspecialchars($data["id_department"]);
    $query = "UPDATE employee SET
        nama = '$nama',
        id_department = '$id_department'
        WHERE id_employe = $id_employe
    ";
    // query insert data
    mysqli_query($db, $query);

    return mysqli_affected_rows($db);
}

```

Gambar 2.12 Function Ubah

Pada function ubah mempunyai parameter \$data dan berisikan cara untuk mengubah data employee. global \$db; untuk membuat variable \$db secara global dan selanjutnya htmlspecialchars adalah sebuah fungsi atau sebuah perintah atau syntax yang di miliki oleh PHP yang berguna untuk menontaktifkan seluruh perintah – perintah html mengubah specialpada sebuah halaman web. Setelah itu memberikan perintah query untuk update data.

Dapat di lihat gambar 2.13 di bawah ini terdapat function query untuk m data employe dan department. Selanjutnya adalah perkondisian apabila data employe berhasil di ubah maka terdapat pop up dengan kalimat data berhasil di ubah. Sebelum melihat pop up berhasil di ubah dapat di lihat pada gambar 2.14 di bawah ini, merupakan sebuah form untuk mengubah data employee yang dimana mempunyai method post dan value. Method post merupakan fungsi untuk mengirimkan data.

```

<?php

require 'functionemployee.php';

// ambil data di URL
$id = $_GET["id"];

// query data mahasiswa berdasarkan ID
$employee = query("SELECT * FROM employee INNER JOIN department ON employee.id_department=department.id WHERE employee.id_employe=$id");
$departs = query("SELECT * FROM department");

// cek tombol submit sdh ditekan atau belum
if(isset($_POST["submit"])) {
    // cek apakah data berhasil di ubah atau tidak
    if (ubah($_POST)>0){
        echo "
        <script>
            alert('data berhasil diubah!');
            document.location.href = 'employee.php';
        </script>
        ";
    } else {
        echo "
        <script>
            alert('data gagal diubah!');
            document.location.href = 'employee.php';
        </script>
        ";
    }
}

```

Gambar 2.13 Ubah Data

```

<form action="" method="post">

<ul>

<li>
    <label for="nama">Nama : </label>
    <input type="text" name="nama" id="nama" required value="<?php echo $employee["nama"]; ?>" />
</li>

<li>
    <label hidden for="nama">Nama : </label>
    <input type="text" hidden name="id_employe" id="nama" required value="<?php echo $employee["id_employe"]; ?>" />
</li>

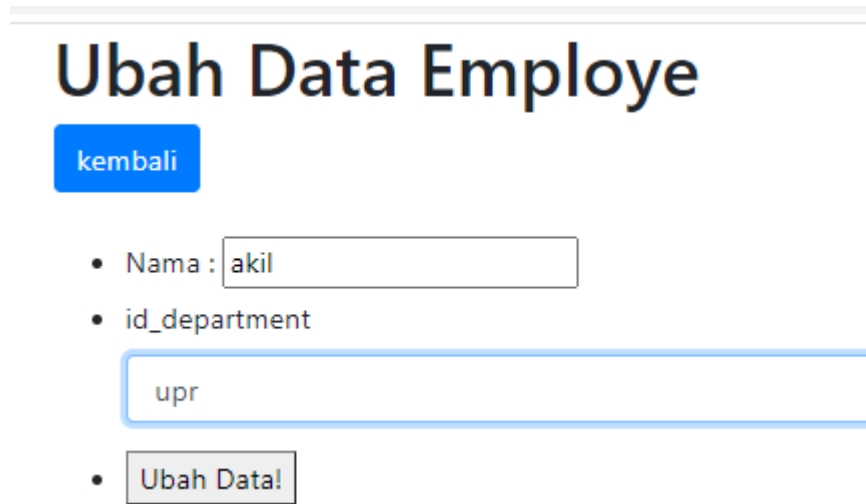
<li class="mb-2">
    <div class="form-group">
        <label for="id_department">id_department</label>
        <select name="id_department" class="form-control" id="id_department">
            <option value="<?php echo $employee["id_department"]; ?>" selected><?php echo $employee["nama_department"]; ?></option>
            <?php foreach($departs as $depart) : ?>
                <option value="<?php echo $depart["id"]; ?>"><?php echo $depart["nama_department"]; ?></option>
            <?php endforeach; ?>
        </select>
    </div>
</li>

</ul>

```

Gambar 2.13 Ubah Data

Setelah itu untuk halaman form ubah dapat di lihat pada gambar 2.14 dan 2.15 di bawah ini yang dimana sekaligus ubah data employee dan pop up berhasil.

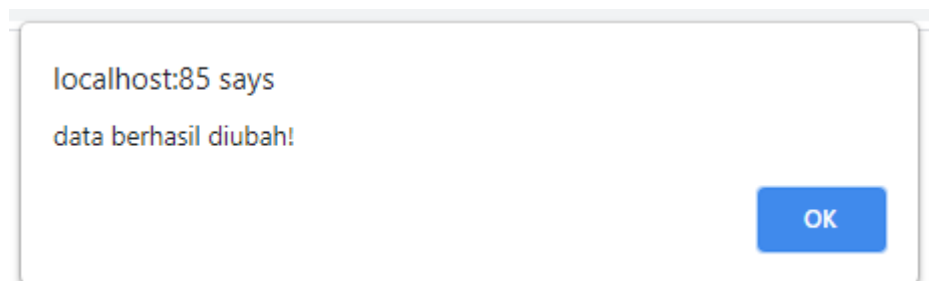


Ubah Data Employee

[kembali](#)

- Nama :
- id_department

Gambar 2.14 Form Ubah Data



localhost:85 says
data berhasil diubah!

[OK](#)

Gambar 2.15 Pop Up Ubah Data

BAB III

KESIMPULAN

CRUD merupakan operasi yang digunakan untuk memanipulasi data database pada sebuah web. CRUD sebenarnya singkatan Create, Read, Update, Delete. Create digunakan untuk membuat atau menginput data, Read untuk menampilkan data, Update untuk mengedit atau mengubah data sedangkan Delete untuk menghapus data. Dengan CRUD bisa membuat form login, register, input, update dan hapus data dan masih banyak lagi. Pada pembuatan form selain CRUD juga wajibkan menggunakan session agar nantinya web lebih aman karena dengan menggunakan session maka tidak bisa untuk langsung masuk ke halaman sesudah login karena setiap halaman akan dilakukan pengecekan session jadi apabila belum login maka halaman tidak bisa diakses

BAB IV

DAFTAR PUSTAKA

- Firman, A., Wowor, H. F., Najoan, X., Teknik, J., Fakultas, E., & Unsrat, T. (2016). Sistem Informasi Perpustakaan Online Berbasis Web. *E-Journal Teknik Elektro Dan Komputer*, 5(2), 29–36.
- Kumpulan Kode-Kode Program PHP Beserta Fungsi dan Contohnya _*. (2020). TECHFOR. <https://www.techfor.id/kumpulan-kode-kode-program-php-beserta-fungsi-dan-contohnya/>
- Praktikum, K. (2021). *MODUL PRAKTIKUM PEMROGRAMAN WEB I Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya*.

LAMPIRAN

```
MariaDB [(none)]> use bab3;
Database changed
MariaDB [bab3]> create table employee(
  -> id_employe int not null auto_increment primary key,
  -> nama varchar(20) not null,
  -> id_department int
  -> );
Query OK, 0 rows affected (0.356 sec)

MariaDB [bab3]> create table department(
  -> id int not null auto_increment primary key,
  -> nama_depart varchar(20) not null
  -> );
Query OK, 0 rows affected (0.305 sec)

MariaDB [bab3]> ALTER TABLE `employee` ADD FOREIGN KEY (`id_department`) REFERENCES `department`(`id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
Query OK, 0 rows affected (1.209 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bab3]>
```

Gambar 1 Database

```
employee.php X
employee.php > ...
1  <?php
2      require 'functionemployee.php';
3      $employs = query("SELECT * FROM employee INNER JOIN department ON employee.id_department=department.id");
4
5
6  ?>
```

Gambar 2 Menampilkan Data

```
$db = mysqli_connect("localhost", "root", "", "bab3");

function query($query){
    global $db;
    $result = mysqli_query($db, $query);
    $rows = [];
    while($row = mysqli_fetch_assoc($result)){
        $rows[]=$row;
    }
    return $rows;
}
```

Gambar 3 Function Query

```

</a>
<table class="table table-striped"> Add a description to this table.
<thead>
  <tr>
    <th scope="col">N0</th>
    <th scope="col">Nama</th>
    <th scope="col">Department</th>
    <th scope="col">Action</th>
  </tr>
</thead>
<tbody>
<?php $i=1; ?>
<?php foreach($employs as $employee) : ?>
  <td><?php echo $i; ?></td>
  <td><?php echo $employee["nama"]; ?></td>
  <td><?php echo $employee["nama_depart"]; ?></td>
  <td>
    <a href="employee_ubah.php?id=<?php echo $employee["id_employee"]; ?>">ubah</a> |
    <a href="employee_delete.php?id=<?php echo $employee["id_employee"]; ?>" onclick="return confirm('yakin?');">hapus</a>
  </td>
</tr>
</tbody>
<?php $i++; ?>
<?php endforeach; ?>

```

Gambar 4 Foreach

Terlihat pada gambar 2.4 di atas adalah cara untuk menampilkan data employee(data pegawai) yang dimana menggunakan foreach untuk perulangan data yang disimpan di database. Dan hasilnya sebagai berikut

Tambah Data		Ke Halaman Department	
N0	Nama	Department	Action
1	Alex	upr	ubah hapus
2	Udin	upr	ubah hapus

Gambar 5 Hasil Menampilkan Data


```

<?php

require 'functionemploye.php';

$departs = query("SELECT * FROM department");

// cek tombol submit sdh ditekan atau belum
if(isset($_POST["submit"])) {
    // cek apakah data berhasil di tambahkan atau tidak
    if (tambah($_POST)>0){
        echo "
        <script>
            alert('data berhasil ditambahkan!');
            document.location.href = 'employe_tambah.php';
        </script>
        ";
    } else {
        echo "
        <script>
            alert('data gagal ditambahkan!');
            document.location.href = 'employe_tambah.php';
        </script>
        ";
    }
}
?>

```

Gambar 6 Menambahkan Data

```

function tambah($data){
    global $db;
    // ambil data dari tiap elemen dalam form
    $nama = htmlspecialchars($data["nama"]);
    $id_department = htmlspecialchars($data["id_department"]);

    $query = "INSERT INTO employee
    VALUES
    ('', '$nama', '$id_department')
    ";
    // query insert data
    mysqli_query($db, $query);

    return mysqli_affected_rows($db);
}

```

Gambar 2.7 Function Tambah

```

<form action="" method="post">
<ul>
  <li class="mb-2">
    <label for="nama">Nama : </label>
    <input type="text" name="nama" id="nama" required>
  </li>

  <li class="mb-2">
    <div class="form-group">
      <label for="id_department">id_department</label>
      <select name="id_department" class="form-control" id="id_department">
        <option selected>Choose One</option>
        <?php foreach($depart as $depart) : ?>
          <option value="<?= $depart["id"] ?>"><?= $depart["nama_depart"] ?></option>
        <?php endforeach; ?>
      </select>
    </div>
  </li>
</ul>

```

Gambar 2.8 Form Tambah

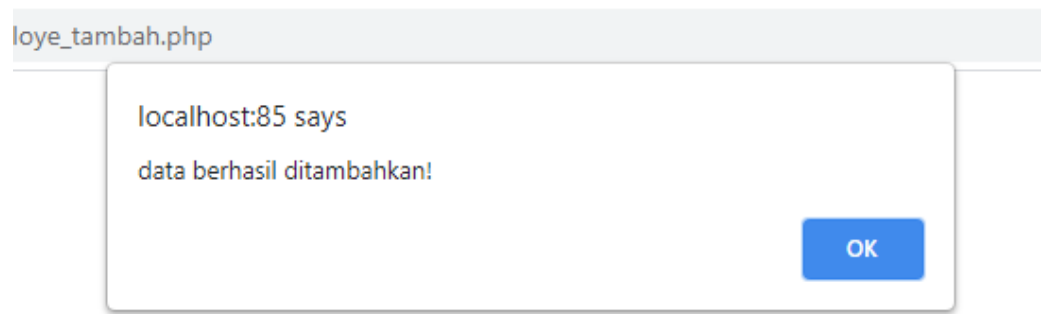
Untuk halaman form tambah dapat di lihat pada gambar 2.9 dan 2.10 di bawah ini yang dimana sekaligus menambahkan data employe dan pop up berhasil.

Tambah Data Employe

kembali

- Nama :
- id_department
-

Gambar 9 Input Data



Gambar 10 Pop Up Berhasil

Tambah Data Ke Halaman Department			
N0	Nama	Department	Action
1	Alex	upr	ubah hapus
2	Udin	upr	ubah hapus
3	akil	ump	ubah hapus

Gambar 11 Hasil Menambahkan Data

```
function ubah($data){
    global $db;
    $id_employe = $data["id_employe"];
    // ambil data dari tiap elemen dalam form
    $nama = htmlspecialchars($data["nama"]);
    $id_department = htmlspecialchars($data["id_department"]);
    $query = "UPDATE employee SET
        nama = '$nama',
        id_department = '$id_department'
        WHERE id_employe = $id_employe
    ";
    // query insert data
    mysqli_query($db, $query);

    return mysqli_affected_rows($db);
}
```

Gambar 12 Function Ubah

```

<form action="" method="post">

<ul>

<li>
<label for="nama">Nama : </label>
<input type="text" name="nama" id="nama" required value="{?php echo $employee["nama"]; ?}">
</li>

<li>
<label hidden for="nama">Nama : </label>
<input type="text" hidden name="id_employe" id="nama" required value="{?php echo $employee["id_empl

<li class="mb-2">

<div class="form-group">
<label for="id_department ">id_department</label>
<select name="id_department" class="form-control" id="id_department ">
<option value="{?php echo $employee["id_department"]; ?}" selected="{?php echo $employee["nama_d
<?php foreach($departs as $depart) : ?>

<option value="{? $depart["id"] ?}">{? $depart["nama_depart"] ?}</option>

<?php endforeach; ?>
</select>
</div>

</li>

```

Gambar 13 Ubah Data

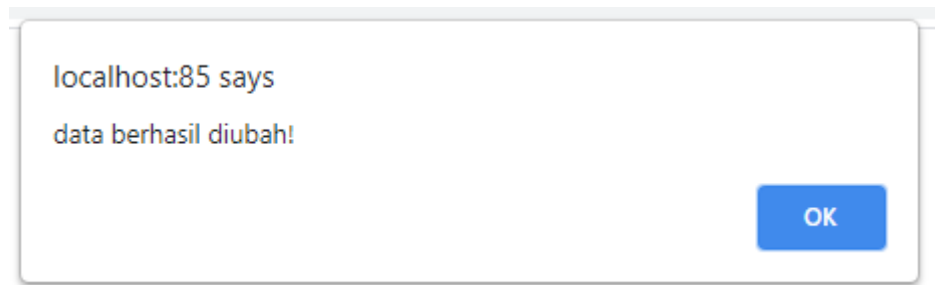
Seteleah itu untuk halaman form ubah dapat di lihat pada gambar 2.14 dan 2.15 di bawah ini yang dimana sekaligus ubah data employe dan pop up berhasil.

Ubah Data Employe

kembali

- Nama :
- id_department
-

Gambar 14 Form Ubah Data



Gambar 15 Pop Up Ubah Data