LAPORAN PRAKTIKUM PENGENALAN APLIKASI BERBASIS INTERNET

MERINGKAS LAPORAN PRAKTIKUM WEEK 6 SESI 2 DAN 3

Agus Pranata Marpaung

13323033

DIII TEKNOLOGI KOMPUTER

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI

Judul Praktikum

Minggu/Sesi	:	VI/2
Kode Mata Kuliah	:	1331204
Nama Mata Kuliah	:	PENGENALAN APLIKASI BERBASIS INTERNET
Setoran	:	*Softcopy*
Batas Waktu Setoran	:	*4 Maret 2024 jam 12:00*
Tujuan	:	*1. Mahasiswa dapat menggunakan dan mengakses database MySQL, PHP menyediakan 3 cara koneksi: menggunakan PDO (PHP Data Objects), mysqli extension dan mysql extension.*

Ringkasan Praktikum PHP MySQL Part 1: Jenis Koneksi PHP – MySQL: PDO, mysqli, dan mysql extension

A. Perkembangan Cara Koneksi PHP dan MySQL

PHP terus mengalami pembaruan dan perkembangan seiring dengan kemajuan teknologi. Saat ini, paradigma pemrograman berorientasi objek menjadi tren dalam pengembangan PHP, yang juga mempengaruhi pendekatan dalam mengakses database MySQL.

Terdapat tiga metode umum untuk mengakses MySQL dari PHP, yaitu melalui PDO (PHP Data Objects), ekstensi mysqli, dan ekstensi mysql. PDO menggunakan pendekatan berbasis objek, mysqli tersedia dalam mode objek dan prosedural, sementara mysql menggunakan pendekatan prosedural sepenuhnya.

Berikut beberapa pengertian dan perbedaan ketiga metode ini:

1. **Koneksi MySQL dengan mysql extension**
   Untuk mengakses MySQL dari PHP, kita dapat menggunakan fungsi-fungsi seperti *mysql_connect (), mysql_query(),* dan *mysql_fetch-array()*. Fungsi-fungsi tersebut tergabung dalam **mysql extension** yang mana pada saat itu **PDO** dan **mysql_extension** masih jarang digunakan.

   Namun ketika adanya PHP versi 5.5.0, PHP memutuskan untuk membuat **mysql extension** yang berstatus *deprecated* yang artinya pengaksesan database MySQL menggunakan fungsi *mysql extension* tidak disarankan lagi.

2. **Koneksi MySQL dengan mysqli extension**
   PHP menyediakan mysqli extension sebagai pengganti mysql extension. Mysqli extension merupakan perbaikan dari mysql extension dan dikembangkan untuk mendukung fitur-fitur terbaru MySQL 4.1 ke atas. Syntax mysqli mirip dengan mysql extension, memudahkan pengguna yang telah terbiasa dengan mysql extension untuk beralih. Selain itu, PHP juga mendukung pengaksesan database MySQL melalui PHP Data Objects (PDO).

3. **Koneksi MySQL dengan PDO (PHP Data Objects)**
   PDO (PHP Data Objects) adalah ekstensi PHP yang bertindak sebagai antarmuka universal untuk mengakses berbagai jenis database, tidak hanya MySQL. Menggunakan PDO dalam kode pemrograman memungkinkan fleksibilitas saat berpindah dari satu jenis database ke yang lain tanpa perlu mengubah seluruh kode program. Oleh karena itu, metode yang disarankan untuk koneksi PHP dengan MySQL adalah menggunakan mysqli atau PDO.

B. Perbandingan antara mysqli, PDO dan mysql extension

Berikut beberapa perbandingan antara mysli, PDO, dan mysql extension dari halaman resmi php:

Comparison of MySQL API options for PHP

	PHP's mysqli Extension	PDO (Using PDO MySQL Driver and MySQL Native Driver)
PHP version introduced	5.0	5.0
MySQL development status	Active development	Active development
API supports Charsets	Yes	Yes
API supports server-side Prepared Statements	Yes	Yes
API supports client-side Prepared Statements	No	Yes
API supports Stored Procedures	Yes	Yes
API supports Multiple Statements	Yes	Most
Supports all MySQL 4.1+ functionality	Yes	Most

C. Cara Penulisan mysql extension, mysqli extension, dan PDO (PHP Data Objects)

Berikut contoh kode PHP untuk mengakses database MySQL menggunakan mysql extension, mysqli extension, dan PDO:

```php
<?php
// cara mengakses MySQL menggunakan mysql extension:
$link = mysql_connect("localhost", "root", "qwerty"); mysql_select_db("universitas");
$result = mysql_query("SELECT * FROM mahasiswa");
$row = mysql_fetch_assoc($result);
// cara mengakses MySQL menggunakan mysqli extension:
$mysqli = new mysqli("localhost", "root", "qwerty", "universitas");
$result = $mysqli->query("SELECT * FROM mahasiswa");
$row = $result->fetch_assoc();
// cara mengakses MySQL menggunakan PDO:
$pdo = new PDO('mysql:host=localhost; dbname=universitas', 'root', 'qwerty');
$statement = $pdo->query("SELECT * FROM mahasiswa");
$row = $statement->fetch(PDO::FETCH_ASSOC);
?>

```

Dalam contoh di atas, pengguna MySQL diasumsikan sebagai root dengan kata sandi 'qwerty', dan basis data bernama 'universitas'. Kode program tidak perlu dipahami saat ini karena akan dibahas secara lengkap dalam Praktikum belajar PHP MySQL di dunia ilkom ini.

Tahap awal akan menggunakan metode ekstensi mysql yang berbasis fungsi terlebih dahulu, meskipun tidak disarankan lagi, karena metode ini paling mudah dipelajari dan telah dikenal luas. Setelahnya, kita akan beralih ke ekstensi mysqli yang dapat ditulis dengan gaya prosedural (menggunakan fungsi-fungsi) atau gaya objek.

Praktikum PHP MySQL Part 2: Perbedaan mysql dan mysqli extension PHP

A. Perbedaan Antara mysql extension Dengan mysqli extension

Selama praktikum PHP MySQL di duniailkom dari Part 4 sampai 11, kita menggunakan extension mysql dari PHP untuk mengakses MySQL. Meskipun fungsi-fungsi seperti mysql_connect, mysql_query, dan mysql_fetch_array tidak disarankan lagi, kita tetap mempelajarinya sebagai dasar sebelum beralih ke extension yang lebih baru: mysqli dan PDO.

Mulai dari PHP versi 5.5, extensi mysql tidak lagi disarankan dan berstatus deprecated, yang berarti mungkin akan dihapus pada versi PHP berikutnya. Kita disarankan untuk beralih menggunakan mysqli atau PDO.

Mari kita bahas mysqli terlebih dahulu. Mysqli merupakan kependekan dari MySQL Improved Extension. Ini adalah versi perbaikan dan penambahan dari extension mysql sebelumnya yang umum digunakan. Mysqli dibuat untuk mendukung fitur-fitur terbaru dari MySQL Server versi 4.1 ke atas.

Secara umum, tidak ada perbedaan besar antara mysql extension dengan mysqli extension. Nama-nama fungsi dalam mysqli sebagian besar mirip dengan mysql extension. Misalnya, mysql_connect() menjadi mysqli_connect(), dan mysql_query() menjadi mysqli_query().

Selain menambahkan huruf "i" dalam nama fungsi, argumen-argumen yang dibutuhkan juga hampir mirip. Satu-satunya perbedaan signifikan adalah penempatan argumen resources, yang biasanya di akhir fungsi dalam mysql extension, sekarang diletakkan di awal dalam mysqli.

Contoh di dalam mysql, kita menulis:
mysql_query("SELECT * FROM mahasiswa_ilkom", $link)\

sedangkan di mysqli, penulisannya menjadi:
mysqli_query($link, "SELECT * FROM mahasiswa_ilkom")

B. Mengenal 2 jenis Mysqli Style: Procedural dan Object Oriented

Untuk memudahkan proses "migrasi" dari mysql ke mysqli, PHP menyediakan dua alternatif cara penulisan mysqli.

1. Procedural style: Cara ini mirip dengan extension mysql, di mana kita menggunakan fungsi-fungsi untuk mengakses database MySQL.

2. Object-oriented style: Dengan cara ini, kita menggunakan aturan penulisan pemrograman objek untuk berkomunikasi dengan MySQL.

Kedua jenis gaya pemrograman mysqli ini menggunakan nama fungsi dan metode yang kurang lebih sama. Sebagai contoh, dalam procedural style mysqli, terdapat fungsi

mysqli_query(), sementara dalam OOP style mysqli, kita menggunakan metode $mysqli->query().

Praktikum PHP MySQL Part 3: Cara Menampilkan Data dengan mysqli PHP (Procedural Style)

A. Perbedaan Antara mysql extension Dengan mysqli extension

Untuk membuat koneksi antara PHP dengan MySQL Server, kita menggunakan fungsi mysqli_connect(). Fungsi ini memerlukan argumen yang sama dengan fungsi mysql_connect(), yaitu: alamat host, nama pengguna, dan kata sandi. Sebagai contoh, untuk terhubung ke MySQL di localhost menggunakan pengguna root dan kata sandi "qwerty", kita akan menggunakan kode program berikut:

```php
<?php
$link = mysqli_connect("localhost", "root", "qwerty");
?>

```

Sebagai tambahan, fungsi mysqli_connect() memiliki argumen keempat yang opsional. Kita dapat menambahkan nama database yang ingin digunakan saat proses koneksi dengan MySQL. Misalnya, jika kita ingin langsung memilih database "universitas" saat koneksi, contoh penulisannya adalah sebagai berikut:

B. Cara Menutup Koneksi Dengan Fungsi mysqli_close()

Walaupun PHP secara otomatis akan menutup koneksi dengan MySQL setelah halaman selesai diproses, kita juga dapat menutupnya secara manual menggunakan fungsi mysqli_close(). Fungsi ini membutuhkan satu argumen, yaitu variabel resources hasil pemanggilan fungsi mysqli_connect(). Berikut adalah contoh penggunaan fungsi mysqli_close():

```php
<?php
// buat koneksi dengan MySQL 4
$link = mysqli_connect("localhost", "root", "qwerty","universitas"); 7
//.... proses PHP
//.... proses PHP
//.... proses PHP
// tutup koneksi
mysqli_close($link);
?>

```

C. Cara Menjalankan Query MySQL dengan fungsi mysqli_query()

Untuk menjalankan query MySQL dengan menggunakan mysqli extension, kita menggunakan fungsi mysqli_query(). Perbedaannya dengan fungsi mysql_query() terletak pada urutan argumen. Di mysqli_query(), variabel resources hasil mysqli_connect() harus ditulis sebagai argumen pertama dan wajib ada.

Berikut contoh dengan menggunakan *mysql_extension*:

```php
<?php
    //buat koneksi dengan MySQL
    $link=mysql_connect('localhost','root','');

    //gunakan database universitas
    $result=mysql_query('USE universitas',$link);

    //tampilkan tabel mahasiswa_ilkom
    $result=mysql_query('SELECT * FROM mahasiswa_ilkom',$link);
?>
```

Untuk menggunakan *mysqli extension*, berikut perubahan cara penggunaannya:

```php
<?php
    // buat koneksi dengan MySQL, gunakan database:
    $link = mysqli_connect("localhost", "root", "",
    "universitas");

    // jalankan query
    $result = mysqli_query($link, "SELECT * FROM mahasiswa_ilkom");
?>
```

D. Cara Menampilkan Data MySQL Dengan mysqli

Untuk menampilkan hasil query MySQL dengan mysqli extension, terdapat tiga fungsi yang sering digunakan: mysqli_fetch_row(), mysqli_fetch_array(), dan mysqli_fetch_object(). Fungsi-fungsi ini memiliki cara penggunaan yang hampir sama dengan versi mysql extension, hanya dengan menambahkan huruf "i" di depan. Sebagai contoh, kita dapat menggunakan tabel mahasiswa_ilkom di dalam database universitas.

Untuk menampilkan hasil query MySQL menggunakan mysqli extension, terdapat tiga fungsi yang umum digunakan: mysqli_fetch_row(), mysqli_fetch_array(), dan mysqli_fetch_object(). Cara penggunaannya hampir sama dengan fungsi-fungsi serupa pada mysql extension, hanya perlu menambahkan huruf "i". Sebagai contoh, kita dapat menggunakan tabel mahasiswa_ilkom di database universitas.

Berikut cara menampilkan data MySQL dengan fungsi **mysqli_fetch_row():**

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = mysqli_connect("localhost", "root", "",
    "universitas");

    // jalankan query
    $result = mysqli_query($link, "SELECT * FROM mahasiswa_ilkom");

    // tampilkan query
    while ($row=mysqli_fetch_row($result))
    {
        echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
        echo "<br />";
    }
?>
```

Berikut cara menampilkan data MySQL dengan fungsi **mysqli_fetch_array():**

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = mysqli_connect("localhost", "root", "","universitas");

    // jalankan query
    $result = mysqli_query($link, "SELECT * FROM mahasiswa_ilkom");

    // tampilkan query
    while ($row=mysqli_fetch_array($result,MYSQLI_ASSOC))
    {
        echo $row['nim']." ".$row['nama']." ".$row['umur']."";
        echo $row['tempat_lahir']." ".$row['IPK'];
        echo "<br />";
    }
?>
```

Perbedaan utama antara fungsi mysqli_fetch_array() dan mysql_fetch_array() adalah letak string penentu metode array. Dalam mysqli_fetch_array(), string penentu metode array berada pada argumen kedua, sedangkan dalam mysql_fetch_array(), string ini berada pada argumen pertama. Untuk mengakses array dengan nama, kita menggunakan string

MYSQLI_ASSOC. Sedangkan string MYSQL_NUM dan MYSQL_BOTH juga tersedia untuk penggunaan lainnya.

Berikut cara mengakses hasil query sebagai objek:

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect("localhost", "root", "",
"universitas");
// jalankan query
$result = mysqli_query($link, "SELECT * FROM mahasiswa_ilkom");
// tampilkan query
while ($row=mysqli_fetch_object($result))
{
echo $row->nim." ".$row->nama." ".$row->umur." ";
echo $row->tempat_lahir." ".$row->IPK;
echo "<br />";
}
?>
```

Praktikum PHP MySQL Part 4: Cara Menampilkan Data dengan mysqli PHP (Object Style)

A. Cara Penulisan Object Style mysqli

mysqli menyediakan dua gaya pengkodean: procedural style dan object-oriented style. Saat ini, penggunaan pemrograman berbasis objek lebih populer karena kelebihannya dalam organisasi dan pengembangan kode yang lebih terstruktur, terutama dalam proyek besar. Prinsip-prinsip OOP seperti inheritance, encapsulation, dan polymorphism membantu menjaga kebersihan dan keterbacaan kode.

B. Cara Membuat Koneksi dengan MySQL (mysqli constructor)

Untuk menghubungkan PHP dengan MySQL menggunakan mysqli object style, kita menggunakan mysqli constructor. Constructor mysqli adalah fungsi yang digunakan untuk membuat objek baru dari class mysqli. Argumen dalam constructor mysqli sama dengan argumen fungsi mysqli_connect(), yaitu host, nama pengguna, dan kata sandi pengguna. Berikut contohnya:

```php
<?php
$link = new mysqli("localhost", "root", "qwerty");
?>

```

Constructor mysqli memiliki argumen tambahan yang bersifat opsional, yaitu nama database yang ingin digunakan. Jika kita ingin secara langsung mengakses database "universitas", maka kode programnya menjadi:

```php
<?php
$link = new mysqli("localhost", "root", "qwerty","universitas");

```

C. Cara Menutup Koneksi MySQL Dengan Method mysqli::close()

Meskipun PHP akan secara otomatis menutup koneksi ke MySQL setelah halaman selesai diproses, kita juga dapat menutupnya secara manual menggunakan metode close() dari objek mysqli, yang biasanya ditulis sebagai mysqli::close(). Berikut contoh penggunaannya:

```php
<?php
// buat koneksi dengan MySQL
$link = new mysqli("localhost", "root", "qwerty","universitas");
    //.... Proses PHP
    //.... Proses PHP
    //.... Proses PHP

    // tutup koneksi
    $link->close();
?>

```

D. Cara Menjalankan Query MySQL Dengan Method mysqli::query()

Jika dalam penulisan procedural kita menggunakan fungsi mysqli_query() untuk menjalankan query MySQL, maka dalam gaya objek, kita mengaksesnya menggunakan metode mysqli::query().

Berikut kode programnya:

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = new mysqli("localhost", "root", "qwerty","universitas");

    // jalankan query
    $result = $link->query("SELECT * FROM mahasiswa_ilkom");
?>

```

E. Cara Menampilkan Data MySQL Dengan mysqli object style

Untuk menampilkan hasil query MySQL, mysqli object memiliki banyak method. Tiga cara yang paling sering digunakan adalah menggunakan method fetch_row(), fetch_array(), dan fetch_object(). Penggunaannya hampir sama dengan padanan fungsinya di procedural style, yaitu dengan fungsi mysqli_fetch_row(), mysqli_fetch_array(), dan mysqli_fetch_object().

Berikut kodingan untuk menampilkan data mysqli dengan method **fetch_row().**

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = new mysqli("localhost", "root", "qwerty","universitas");

    // jalankan query
    $result = $link->query("SELECT * FROM mahasiswa_ilkom");

    // tampilkan query
    while ($row= $result->fetch_row())
    {
        echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
        echo "<br />";
    }
?>

```

Cara kedua untuk menampilkan data mysqli dengan method **fetch_array().**

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = new mysqli("localhost", "root", "qwerty",
    "universitas");

    // jalankan query
    $result = $link->query("SELECT * FROM mahasiswa_ilkom");

    // tampilkan query
    while ($row=$result->fetch_array(MYSQLI_ASSOC))
    {
        echo $row['nim']." ".$row['nama']." ".$row['umur']."";
        echo $row['tempat_lahir']." ".$row['IPK'];
        echo "<br />";
    }
?>
```

Cara ketiga untuk menampilkan data mysqli dengan method **fetch_object()**.

```php
<?php
    // buat koneksi dengan MySQL, gunakan database: universitas
    $link = new mysqli("localhost", "root", "qwerty","universitas");

    // jalankan query
    $result = $link->query("SELECT * FROM mahasiswa_ilkom");

    // tampilkan query
    while ($row=$result->fetch_object())
    {
        echo $row->nim." ".$row->nama." ".$row->umur." ";
        echo $row->tempat_lahir." ".$row->IPK;
        echo "<br />";
    }
?>
```

Praktikum PHP MySQL Part 5: Cara Menampilkan Pesan Kesalahan (Error) mysqli Extension

A. Cara Menampilkan Pesan Kesalahan (Error) Procedural Style mysqli

Dalam pemrograman mysqli dengan gaya procedural, untuk menampilkan pesan kesalahan, kita menggunakan beberapa fungsi seperti mysqli_connect_errno(), mysqli_connect_error(), mysqli_errno(), dan mysqli_error(). Berikut adalah contoh penggunaannya.

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
 $link = mysqli_connect('localhost', 'root',
'','universitas');

 // cek koneksi
 if (!$link) {
  die('Koneksi Error : '.mysqli_connect_errno() .' -
'.mysqli_connect_error());
 }

 // koneksi berhasil
 echo 'Koneksi Berhasil :'.mysqli_get_host_info($link)."<br
/>";

 // tutup koneksi
 mysqli_close($link);
?>
```

Setelah melakukan koneksi dengan mysqli_connect(), saya menggunakan kondisi if(!$link) untuk memeriksa apakah koneksi berhasil. Jika koneksi gagal, nilai FALSE akan dikembalikan oleh fungsi mysqli_connect(). Dengan menambahkan tanda !, saya membalikkan nilai FALSE menjadi TRUE sehingga kondisi if(!$link) akan dieksekusi ketika terjadi kesalahan koneksi.

Jika kondisi if(!$link) terpenuhi (artinya terjadi kesalahan), fungsi die() akan menghentikan proses PHP. Kemudian, saya menampilkan pesan kesalahan menggunakan fungsi mysqli_connect_errno() untuk menampilkan nomor kode error dan mysqli_connect_error() untuk menampilkan pesan error.

Jika koneksi berhasil, kita dapat menggunakan fungsi mysqli_get_host_info() untuk menampilkan informasi tentang jenis koneksi yang sedang digunakan, apakah dari localhost atau alamat IP.

Kita bisa menggunakan fungsi **mysqli_errno()** dan **mysqli_error()** untuk menampilkan pesan error yang terjadi ketika query dijalankan.

```php
<?php
 // buat koneksi dengan MySQL, gunakan database:
universitas
 $link = mysqli_connect('localhost', 'root', '','universitas');

 // cek koneksi
 if (!$link) {
  die('Koneksi Error : '.mysqli_connect_errno().' -
'.mysqli_connect_error());
 }

 // koneksi berhasil
 echo 'Koneksi Berhasil :'.mysqli_get_host_info($link)."<br
/>";

 // jalankan query
 $result = mysqli_query($link,"SELECT * FROM
mahasiswa_ilkom");

 // cek hasil query
 if (!$result) {
  die('Query Error : '.mysqli_errno($link). ' -
'.mysqli_error($link));
 }
// tampilkan query
while ($row=mysqli_fetch_row($result)) {
echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
echo "<br />";
}
// tutup koneksi
mysqli_close($link);
?>
```

Fungsi mysqli_errno() digunakan untuk menampilkan nomor kode error, dan fungsi mysqli_error() digunakan untuk menampilkan pesan error yang terjadi.

B. Cara Menampilkan Pesan Kesalahan (Error) Object Style mysqli

Dalam object style mysqli, untuk menampilkan pesan kesalahan, kita tidak lagi menggunakan fungsi, melainkan dengan memeriksa properti error dari objek mysqli. Properti ini memiliki nama yang mirip dengan fungsi yang digunakan pada procedural style mysqli.

Berikut contoh kode program untuk menampilkan kesalahan MySQL menggunakan mysqli dengan object style:

```php
<?php
 // buat koneksi dengan MySQL, gunakan database: universitas
 $mysqli = new mysqli("localhost", "root", "","universitas");

 // cek koneksi
 if ($mysqli->connect_errno) {
  die('Koneksi Error: '.$mysqli->connect_errno.' - '.
  $mysqli->connect_error);
 }

 // koneksi berhasil
 echo 'Koneksi Berhasil : '.$mysqli->host_info."<br />";

 // jalankan query
 $result = $mysqli->query("SELECT * FROM mahasiswa_ilkom");

 // cek hasil query
 if ($mysqli->errno) {
  die('Query Error : '.$mysqli->errno.'-'.$mysqli->error);
 }

 // tampilkan query
 while ($row= $result->fetch_row()) {
  echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
  echo "<br />";
 }

 // tutup koneksi
 $mysqli->close();
?>
```

Dalam pengecekan kesuksesan koneksi dan query dalam mysqli, kita dapat memeriksa isi dari property $mysqli->connect_errno dan $mysqli->errno. Kedua property ini akan tetap kosong jika tidak ada kesalahan, dan akan berisi nilai jika terdapat error pada koneksi atau query MySQL. Oleh karena itu, untuk memeriksa kesalahan pada koneksi MySQL, kita dapat menggunakan kondisi if ($mysqli->connect_errno), sedangkan untuk mengecek kesalahan pada query, kita dapat menggunakan if ($mysqli->errno).

Praktikum PHP MySQL Part 6: Pengertian dan Cara Penggunaan Prepared Statements mysqli

A. Pengertian Prepared Statements MySQL

Prepared statements adalah fitur MySQL yang memungkinkan pemisahan antara query inti dan data query, meningkatkan keamanan dan kinerja query yang akan digunakan berulang kali.

B. Proses Pembuatan Prepared Statement MySQL

Pembuatan prepared statements melibatkan tiga langkah utama:

1. Prepare melibatkan penyusunan query tanpa data aktual, tetapi dengan penggunaan tanda tanya (?) sebagai tempat penampungan data.
   Contoh:
   "SELECT * FROM mahasiswa_ilkom WHERE nama=?" atau
   "INSERT INTO mahasiswa_ilkom VALUES (?, ?, ?, ?, ?)".
   Query ini dikirimkan dari PHP ke MySQL Server dan disimpan sementara untuk tahap berikutnya: Bind.

2. Bind melibatkan pengiriman data yang sesuai dengan tanda tanya yang telah ditetapkan pada tahap Prepare. Jumlah data yang dikirimkan harus sesuai dengan jumlah tanda tanya dalam query yang disiapkan sebelumnya.

3. Setelah tahap Prepare dan Bind selesai, langkah terakhir adalah menjalankan prepared statement (Execute).

C. Kenapa Harus Menggunakan Prepared Statements?

Penggunaan prepared statements memberikan keuntungan besar dalam hal keamanan. Biasanya, data dalam sebuah query berasal dari inputan pengguna. Dengan menggunakan prepared statements, perintah query dipisahkan dari data, sehingga mencegah kemungkinan penyisipan perintah SQL dari input pengguna, yang dikenal sebagai SQL Injection.

D. Cara Penggunaan Prepared Statements mysqli

Sebagai gambaran, berikut adalah contoh kode program untuk menampilkan data mahasiswa dari tabel mahasiswa_ilkom menggunakan prepared statement mysqli dalam gaya procedural.

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root', '','universitas');

// cek koneksi
if (!$link) {
 die('Koneksi Error : '.mysqli_connect_errno().' -
'.mysqli_connect_error());
}

// buat prepared statements
$stmt = mysqli_prepare($link, "SELECT * FROM mahasiswa_ilkom
WHERE nama=?");

// siapkan "data" query
$nama_mhs="Neil Situmorang";

// hubungkan "data" dengan prepared statements: bind
mysqli_stmt_bind_param($stmt, "s", $nama_mhs);

// jalankan query: execute
mysqli_stmt_execute($stmt);

// cek hasil query
if (!$stmt) { die('Query Error : '.mysqli_errno($link).' -
'.mysqli_error($link));
}

// ambil hasil query
$result=mysqli_stmt_get_result($stmt);

// tampilkan hasil query
while ($row= mysqli_fetch_row($result)) {
 echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
echo "<br />";
}
// tutup statements
mysqli_stmt_close($stmt);
// tutup koneksi
mysqli_close($link);
?>
```

Praktikum PHP MySQL Part 7: Cara Menampilkan Data dengan mysqli Prepared Statements

A. Cara Menampilkan Data MySQL Menggunakan Prepared Statements mysqli

1. Langkah pertama: *Prepared*

Untuk proses prepared, mysqli PHP menyediakan fungsi mysqli_prepare(). Fungsi ini memerlukan 2 argumen, yaitu variabel hasil pemanggilan fungsi mysqli_connect() dan prepared query yang akan dijalankan. Berikut adalah contoh penulisannya:

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root', '','universitas');
// buat prepared statements
$stmt = mysqli_prepare($link, "SELECT * FROM
mahasiswa_ilkom WHERE nama=?");
?>
```

2. Langkah kedua: *Bind*

Pada proses bind, kita mengirimkan data kepada MySQL untuk menggantikan tanda "?" yang sebelumnya dibuat pada proses prepared. Dalam mysqli PHP, proses bind dilakukan menggunakan fungsi mysqli_stmt_bind_param(). Fungsi ini membutuhkan setidaknya 3 argumen, seperti contoh berikut:

```php
<?php
// siapkan "data" query
$nama_mhs="Neil Situmorang";
// hubungkan "data" dengan prepared statements
mysqli_stmt_bind_param($stmt, "s", $nama_mhs);
?>
```

Argumen pertama dari fungsi mysqli_stmt_bind_param() adalah variabel hasil pemanggilan fungsi mysqli_prepare(), yang dalam contoh ini adalah variabel $stmt.

Argumen kedua adalah string yang menunjukkan jenis tipe data dari argumen ketiga, yang merupakan data yang akan diinput ke dalam query.

Argumen ketiga adalah data yang akan menggantikan tanda "?" dari query, contohnya adalah "Neil Situmorang". Namun, karena fungsi mysqli_stmt_bind_param() membutuhkan data dalam bentuk variabel, data tersebut harus disimpan terlebih dahulu dalam variabel $nama_mhs.

Argumen kedua dari fungsi mysqli_stmt_bind_param() mengharuskan penjelasan terpisah. Argumen ini berisi string yang menunjukkan tipe data dari argumen ketiga. PHP menyediakan empat jenis tipe data:

- i = integer
- d = double

- s = string
- b = blob (binary)

3. **Langkah ketiga: *Execute***

Setelah proses bind selesai, langkah selanjutnya adalah menjalankan query dengan menggunakan fungsi mysqli_stmt_execute(). Fungsi ini membutuhkan satu argumen, yaitu variabel hasil pemanggilan fungsi mysqli_prepare().

```php
<?php
mysqli_stmt_execute($stmt);
?>
```

Fungsi mysqli_stmt_execute() memerintahkan MySQL untuk menjalankan perintah prepared statement yang sudah dibuat.

B. Menampilkan data hasil query

Fungsi mysqli_stmt_get_result() digunakan untuk mengambil hasil query dari MySQL. Argumennya adalah variabel hasil fungsi mysqli_prepare(). Dan Fungsi ini juga dapat mengembalikan nilai bertipe *resources*. Berikut codenya:

```php
<?php
// ambil hasil query
$result=mysqli_stmt_get_result($stmt);
?>
```

Berikut untuk menampilkan data dengan menggunakan **mysql_fetch_row()** atau **mysql_fetch_array().**

```php
<?php
// tampilkan hasil query
while ($row= mysqli_fetch_row($result)) {
 echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
 echo "<br />";
 }
?>
```

Fungsi mysqli_stmt_close() digunakan untuk menutup proses prepared statement secara manual, namun tidak wajib. Jika tidak dipanggil, PHP akan secara otomatis menutup koneksi ke MySQL saat halaman selesai diproses.

Berikut kode programnya dengan lengkap:

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root',
'','universitas');

// cek koneksi
if (!$link) {
 die('Koneksi Error : '.mysqli_connect_errno().' -
'.mysqli_connect_error());
}

// buat prepared statements
$stmt = mysqli_prepare($link, "SELECT * FROM mahasiswa_ilkom WHERE nama=?");

// cek query
if (!$stmt) {
 die('Query Error : '.mysqli_errno($link).' -
'.mysqli_error($link));
}

// siapkan "data" query
$nama_mhs="Neil Situmorang";

// hubungkan "data" dengan prepared statements
mysqli_stmt_bind_param($stmt, "s", $nama_mhs);

// jalankan query
mysqli_stmt_execute($stmt);

// ambil hasil query
$result=mysqli_stmt_get_result($stmt);

// tampilkan hasil query
while ($row= mysqli_fetch_row($result)) {
 echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
 echo "<br />";
}

// tutup statements
mysqli_stmt_close($stmt);

// tutup koneksi
mysqli_close($link);
?>
```

C. Prepared Statement mysqli Object Style

Sebagai alternatif, berikut cara penulisan **prepared statement** yang menggunakan *object style mysqli*.

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$mysqli = new mysqli("localhost", "root","","universitas");

// cek koneksi
if ($mysqli->connect_errno) {
die('Koneksi gagal: ' .$mysqli->connect_errno.' -
'.$mysqli->connect_error);
}

// buat prepared statements
$stmt = $mysqli->prepare("SELECT * FROM mahasiswa_ilkom WHERE nama=?");

// cek query
if (!$stmt) {
die('Query Error : '.$mysqli->errno.' - '.$mysqli->error);
}

// siapkan "data" query
$nama_mhs="Neil Situmorang";

// tutup koneksi
$mysqli->close();

// hubungkan "data" dengan prepared statements
$stmt->bind_param("s", $nama_mhs);

// jalankan query
$stmt->execute();

// hubungkan hasil query
$result = $stmt->get_result();

// tampilkan query
while ($row= $result->fetch_row()) {
echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
echo "<br />";
}

// tutup statements
$stmt->close();
?>
```

Praktikum PHP MySQL Part 8: Cara Menginput Data dengan mysqli Prepared Statements

A. Cara Menginput Data dengan myqli Prepared Statements

Perbedaan utama antara penggunaan prepared statements untuk input dan output data terletak pada jenis query yang digunakan. Saat menampilkan data, kita menggunakan perintah SELECT, sedangkan untuk memasukkan data, kita menggunakan perintah INSERT. Prepared statements memberikan keamanan tambahan terutama untuk perintah yang memodifikasi isi tabel seperti INSERT atau UPDATE, dan juga dapat meningkatkan kecepatan eksekusi terutama untuk input data yang berulang.

1. Prepared

Untuk menginput data kedalam tabel, kita menggunakan query INSERT…VALUES. Berikut code untuk fungsi **mysqli_prepare()** untuk proses penambahan data:

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root', '','universitas');
// buat prepared statements
$stmt = mysqli_prepare($link, "INSERT INTO mahasiswa_ilkom VALUES (?, ?, ?, ?, ?)");
?>
```

2. Bind

Proses pengiriman data menggunakan fungsi mysqli_stmt_bind_param(). Karena query INSERT membutuhkan 5 variabel, fungsi mysqli_stmt_bind_param() harus mencakup kelima variabel ini untuk menggantikan karakter (?, ?, ?, ?, ?). Sebelumnya, masing-masing nilai harus disimpan dalam variabel terpisah sebelum diinput ke fungsi mysqli_stmt_bind_param().

```php
<?php
// hubungkan "data" dengan prepared statements
mysqli_stmt_bind_param($stmt, "ssisd", $nim_mhs, $nama_mhs, $umur_mhs , $tempat_lahir_mhs, $ipk_mhs);
// siapkan "data" query
$nim_mhs="089023020";
$nama_mhs="Naira Alika";
$umur_mhs=20;
$tempat_lahir_mhs="Padang";
$ipk_mhs=3.9;
?>
```

3. Execute

Setelah query dan data selesai diinput, jalankan query dengan fungsi **mysqli_stmt_execute()**:

```php
<?php
// jalankan query: execute
mysqli_stmt_execute($stmt);
?>
```

B. Memeriksa Hasil Query

Untuk memeriksa keberhasilan query INSERT, kita dapat menggunakan variabel mysqli_stmt_affected_rows().

```php
<?php
// cek hasil query
if (!$stmt) {
    die('Query Error :'.mysqli_errno($link).' '.mysqli_error($link));
}
else {
    echo "Penambahan ".mysqli_stmt_affected_rows($stmt)."data berhasil<br />";
}
?>
```

Ketika query gagal, kondisi if (!$stmt) akan menjadi TRUE, sehingga fungsi die() akan dijalankan untuk menghentikan proses dan menampilkan error yang terjadi. Namun, jika query berhasil, maka bagian else akan dieksekusi.

Fungsi mysqli_stmt_affected_rows() berfungsi serupa dengan fungsi mysql_affected_rows() yang pernah dibahas sebelumnya, dan akan menampilkan jumlah baris yang terpengaruh oleh query.

Berikut contoh kode program PHP dengan lengkap mengenai cara menginput data MySQL dengan menggunakan ***mysqli prepared statement.***

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root', '','universitas');

// buat prepared statements
$stmt = mysqli_prepare($link, "INSERT INTO mahasiswa_ilkom
VALUES (?, ?, ?, ?, ?)");

// hubungkan "data" dengan prepared statements
mysqli_stmt_bind_param($stmt, "ssisd",$nim_mhs, $nama_mhs,
$umur_mhs , $tempat_lahir_mhs, $ipk_mhs);

// siapkan "data" query
$nim_mhs="089023020";
$nama_mhs="Naira Alika";
$umur_mhs=20;
$tempat_lahir_mhs="Padang";
$ipk_mhs=3.9;

// jalankan query
mysqli_stmt_execute($stmt);

// cek hasil query
if (!$stmt) { die('Query Error : '.mysqli_errno($link).'
'.mysqli_error($link));
}else {
 echo "Penambahan ".mysqli_stmt_affected_rows($stmt)." data
berhasil<br />";
}

// jalankan query untuk memeriksa hasil inputan
$result = mysqli_query($link, "SELECT * FROM mahasiswa_ilkom");

// tampilkan query
while ($row=mysqli_fetch_row($result)) {
 echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
 echo "<br />";
}

// tutup statements
mysqli_stmt_close($stmt);

// tutup koneksi
mysqli_close($link);
?>
```

C. Cara Menginput Multiple Data dengan myqli Prepared Statements

Berikut sedikit kode program yang dimodifikasi untuk menampilakn cara menginput banyak data (*multiple data*):

```php
<?php
// buat koneksi dengan MySQL, gunakan database: universitas
$link = mysqli_connect('localhost', 'root', '','universitas');

// buat prepared statements
$stmt = mysqli_prepare($link, "INSERT INTO mahasiswa_ilkom VALUES (?, ?, ?, ?, ?)");

// hubungkan "data" dengan prepared statements
mysqli_stmt_bind_param($stmt, "ssisd",$nim_mhs, $nama_mhs, $umur_mhs , $tempat_lahir_mhs,$ipk_mhs);

// siapkan "data" query 1
$nim_mhs="089023023";
$nama_mhs="Alika Shanum";
$umur_mhs=21;
$tempat_lahir_mhs="Medan";
$ipk_mhs=3.8;

// jalankan query 1
mysqli_stmt_execute($stmt);

// cek hasil query 1
if (!$stmt) {
 die('Query Error : '.mysqli_errno($link).' '.mysqli_error($link));
}else {
 echo "Penambahan ".mysqli_stmt_affected_rows($stmt)."data berhasil<br />";
}

// siapkan "data" query 2
$nim_mhs="089023026";
$nama_mhs="Rina Melita";
$umur_mhs=22;
$tempat_lahir_mhs="Lampung";
$ipk_mhs=3.5;

// jalankan query 2
mysqli_stmt_execute($stmt);

// cek hasil query 2
if (!$stmt) {
 die('Query Error : ' .mysqli_errno($link).'
'.mysqli_error($link));
}else {
 echo "Penambahan
".mysqli_stmt_affected_rows($stmt)."data berhasil<br />";
}
```

```php

// siapkan "data" query 3
$nim_mhs="089023031";
$nama_mhs="Joni Halim";
$umur_mhs=21;
$tempat_lahir_mhs="Palembang";
$ipk_mhs=3.6;

// jalankan query 3
mysqli_stmt_execute($stmt);

// cek hasil query 3
if (!$stmt) {
 die('Query Error : '.mysqli_errno($link).'
'.mysqli_error($link));
}else {
 echo "Penambahan
".mysqli_stmt_affected_rows($stmt)."data berhasil<br />";
}

// jalankan query untuk memeriksa hasil inputan
$result = mysqli_query($link, "SELECT * FROM
mahasiswa_ilkom");

// tampilkan query
while ($row=mysqli_fetch_row($result)) {
 echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
 echo "<br />";
}
// tutup statements
mysqli_stmt_close($stmt);
// tutup koneksi
mysqli_close($link);
?>
```

D. Cara Menginput Multiple Data dengan myqli Prepared Statements (Object Style)

Berikut beberapa modifikasi kode program dengan menggunakan *object style mysqli*:

```php
<?php
 // buat koneksi dengan MySQL, gunakan database: universitas
 $mysqli = new mysqli("localhost", "root", "","universitas");

 // cek koneksi
 if ($mysqli->connect_errno) {
  die('Koneksi gagal: ' .$mysqli->connect_errno.' - '.$mysqli->connect_error);
 }

 // buat prepared statements
 $stmt = $mysqli->prepare("INSERT INTO mahasiswa_ilkom
VALUES (?, ?, ?, ?, ?)");

 // hubungkan "data" dengan prepared statements
 $stmt->bind_param("ssisd",$nim_mhs, $nama_mhs, $umur_mhs ,
$tempat_lahir_mhs,$ipk_mhs);

 // siapkan "data" query
 $nim_mhs="089023020";
 $nama_mhs="Naira Alika";
 $umur_mhs=20;
 $tempat_lahir_mhs="Padang";
 $ipk_mhs=3.9;

 // jalankan query
 $stmt->execute();

 // cek query
 if (!$stmt) {
  die('Query Error : '.$mysqli->errno.' - '.$mysqli ->error);
 }else {
  echo "Penambahan ".$stmt->affected_rows. " data berhasil<br />";
 }

 // jalankan query untuk memeriksa hasil inputan
 $result = $mysqli->query("SELECT * FROM mahasiswa_ilkom");
```

```
38    // tampilkan query
39    while ($row= $result->fetch_row()) {
40    echo "$row[0] $row[1] $row[2] $row[3] $row[4]";
41    echo "<br />";
42    }
43    // tutup statements
44    $stmt->close();
45    // tutup koneksi
46    $mysqli->close();
47    ?>
```