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

Untuk dapat menggunakan dan mengakses database MySQL, PHP menyediakan 3 cara koneksi: menggunakan PDO (PHP Data Objects), mysqli extension dan mysql extension. Dalam Praktikum belajar PHP kali ini kita akan membahas keunggulan masing-masing, dan perbedaan ketiganya.

Perkembangan Cara Koneksi PHP dan MySQL

PHP merupakan bahasa pemograman yang selalu di update dan berkembang mengikuti teknologi terbaru. Saat ini, **Pemrograman Berorientasi Objek** (*Object Oriented Programming*) merupakan trend pemrograman PHP, dan hal ini juga mempengaruhi cara mengakses database MySQL dari PHP.

PHP memiliki 3 cara pengaksesan MySQL, yakni melalui **PDO (PHP Data Objects)**, **mysqli extension** dan **mysql extension**. **PDO** menggunakan *pemrograman objek*, **mysqli extension** tersedia dalam bentuk *objek* dan *prosedural* (diakses melalui fungsi-fungsi) sedangkan **mysql extension** sepenuhnya menggunakan pemograman *prosedural*.

Mari kita bahas pengertian dan perbedaan ketiga metode ini:

---

1. Koneksi MySQL dengan mysql extension

Saat pertama kali mempelajari PHP MySQL sekitar tahun 2008 (atau jika anda pernah mempelajari PHP-MySQL beberapa tahun yang lalu), untuk mengakses MySQL dari PHP, kita menggunakan fungsi-fungsi seperti *mysql_connect(), mysql_query(),* dan *mysql_fetch_array().* Fungsi-fungsi ini tergabung ke dalam **mysql extension** (saat itu **PDO** dan **mysqli extension** masih jarang digunakan)

Namun sekarang (tepatnya mulai PHP versi 5.5.0) PHP memutuskan untuk membuat **mysql extension** berstatus *deprecated*. Yang artinya pengaksesan database MySQL menggunakan fungsi *mysql extension* sudah tidak disarankan lagi.

Programmer PHP diharapkan pindah ke **mysqli extension** atau **PDO** yang berbasis objek. Alasannya, MySQL versi terbaru memiliki fitur-fitur yang semakin lengkap dan kompleks, sehingga PHP memutuskan untuk membuat fungsi extension baru agar programmer PHP bisa menggunakan fitur-fitur ini.

---

2. Koneksi MySQL dengan mysqli extension

Sebagai pengganti **mysql extension**, PHP menyediakan **mysqli extension** (**mysqli** merupakan singkatan dari **MySQL Improved**). Mysqli extension ini pada dasarnya adalah perbaikan dari *mysql extension* dan dikembangkan untuk mendukung fitur-fitur terbaru untuk **MySQL 4.1** keatas.

Hampir semua fungsi yang ada pada *mysql extension* juga tersedia pada *mysqli*. Syntax (aturan penulisan) **mysqli** sangat mirip dengan **mysql** extension. Sehingga jika anda telah lama menggunakan mysql extension, akan sangat mudah untuk beralih menggunakan mysqli extension.

Selain menggunakan mysql maupun mysqli extension, cara ketiga untuk pengaksesan database MySQL dari PHP adalah dengan menggunakan PHP Data Objects (atau sering disingkat dengan **PDO**).

---

3. Koneksi MySQL dengan PDO (PHP Data Objects)

PDO (PHP Data Objects), adalah *extension* atau penambahan fitur dalam PHP yang dirancang sebagai *interface universal* untuk pengaksesan berbagai jenis database (tidak hanya MySQL). Contohnya, jika kita menggunakan **PDO** dalam menulis kode pemograman, lalu suatu saat website kita bertukar database dari **MySQL** ke **Oracle**, maka kita tidak perlu mengubah semua kode program, cukup mengubah cara pemanggilan **PDO** diawal program saja.

Dari ketiga cara koneksi PHP dengan MySQL ini, metode yang disarankan adalah menggunakan **mysqli** atau **PDO**.

Perbandingan antara mysqli, PDO dan mysql extension

Sebagai perbandingan fitur antara ketiga jenis koneksi PHP-MySQL ini, berikut adalah tabel perbandingan antara mysqli, PDO dan mysql extension yang bersumber dari manual resmi PHP:

Comparison of MySQL API options for PHP

	PHP's mysqli Extension	PDO (Using PDO MySQL Driver and MySQL Native Driver)	PHP's MySQL Extension
PHP version introduced	5.0	5.0	Prior to 3.0
Included with PHP 5.x	yes	yes	Yes
MySQL development status	Active development	Active development as of PHP 5.3	Maintenance only
Recommended by MySQL for new projects	Yes - preferred option	Yes	No
API supports Charsets	Yes	Yes	No
API supports server-side Prepared Statements	Yes	Yes	No
API supports client-side Prepared Statements	No	Yes	No
API supports Stored Procedures	Yes	Yes	No
API supports Multiple Statements	Yes	Most	No
Supports all MySQL 4.1+ functionality	Yes	Most	No

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

Untuk mengetahui secara sekilas perbedaan cara pengaksesan database MySQL menggunakan **mysql extension**, **mysqli extension**, dan **PDO**, berikut contoh kode PHPnya:

```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 diatas, dianggap bahwa user MySQL adalah **root** dengan password adalah **'qwerty'**, dan nama database **'universitas'**.

Anda tidak perlu memahami kode program diatas, karena kita akan membahasnya dengan lengkap dalam Praktikum belajar PHP MySQL di duniailkom ini.

Untuk tahap pertama, saya akan menggunakan metode **mysql extension** yang berbasis fungsi terlebih dahulu. Walaupun metode ini tidak disarankan lagi, namun metode inilah yang paling mudah dipelajari dan telah dikenal luas.

Setelah selesai membahas **mysql extension**, selanjutnya kita akan beralih ke **mysqli extension** yang bisa ditulis dengan *procedural style* (menggunakan fungsi-fungsi) ataupun dengan *object style*. Terakhir, kita akan membahas cara membuat koneksi PHP MySQL menggunakan PDO. Cara pengaksesan MySQL dengan PDO yang berbasis objek akan lebih mudah dipahami jika anda telah mempelajari pemrograman berbasis objek.

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

Seperti yang pernah kita bahas dalam Praktikum Jenis Koneksi PHP MySQL: PDO, mysqli, dan mysql extension, penggunaan **mysql extension** tidak lagi direkomendasikan (*deprecated*). Dalam Praktikum PHP MySQL kali ini, kita akan membahas perbedaan **mysql** extension yang "lama" dengan versi barunya, yakni **mysqli**. Extension **mysqli** juga hadir dengan 2 jenis "rasa": **procedural** dan **object oriented**.

Perbedaan Antara mysql extension Dengan mysqli extension

Sepanjang Praktikum PHP MySQL Part 4 sampai 11 di duniailkom ini, kita menggunakan ***extension mysql*** dari PHP untuk mengakses MySQL. Fungsi-fungsi yang kita pelajari tersebut (seperti fungsi ***mysql_connect, mysql_query,*** dan ***mysql_fetch_array***) memang sudah tidak disarankan lagi, tetapi saya tetap membahasnya sebagai dasar bagi kita untuk masuk ke extension yang lebih baru:
mysqli dan **PDO**.

Terhitung dari PHP versi 5.5, extension **mysql** tidak lagi disarankan penggunaannya dan berstatus *deprecated*, yang berarti mungkin akan dihapus pada PHP versi berikutnya. Kita disarankan untuk mulai beralih menggunakan **mysqli** atau **PDO**.
Mari kita bahas tentang **mysqli** terlebih dahulu.

Mysqli merupakan kependekan dari **MySQL Improved Extension**. Seperti yang terlihat dari namanya, extension ini merupakan versi perbaikan dan penambahan dari extension **mysql** sebelumnya yang umum digunakan. Extension ***mysqli*** dibuat untuk mendukung fitur-fitur terbaru dari MySQL Server versi 4.1 keatas.

Secara garis besar, tidak ada perbedaan mencolok antara *mysql extension* dengan *mysqli extension*. Nama-nama fungsi didalam mysqli sebagian besar mirip dengan apa yang telah kita pelajari (*extension mysql*).

Sebagai contoh, untuk membuat koneksi dengan MySQL Server, di dalam mysql kita menggunakan fungsi **mysql_connect()**, sedangkan di dalam mysqli, kita

menggunakan **mysqli_connect()**. Begitu juga dengan fungsi lain seperti **mysql_query()** menjadi **mysqli_query()**.

Selain menambah huruf "i" di dalam nama fungsi, argumen-argumen yang dibutuhkan juga hampir mirip. Perbedaannya, jika di dalam extension mysql umumnya kita meletakkan argumen *resources* di akhir fungsi, maka di dalam **mysqli**, argumen ini diteletakkan di awal.

Sebagai contoh, di dalam mysql kita menulis:

```
mysql_query("SELECT * FROM mahasiswa_ilkom", $link)
```

Sedangkan di dalam mysqli penulisannya menjadi:

```
mysqli_query($link, "SELECT * FROM mahasiswa_ilkom")
```

Namun perbedaan paling mendasar di dalam mysqli adalah: mysqli mendukung cara penulisan **object oriented programming**.

Mengenal 2 jenis Mysqli Style: Procedural dan Object Oriented

Agar proses "*migrasi*" dari **mysql** ke **mysqli** tidak terlalu menyusahkan, PHP memberikan 2 alternatif cara penulisan **mysqli**.

- Cara yang pertama adalah menggunakan **procedural style**. Cara ini mirip dengan extension mysql, dimana kita menggunakan fungsi-fungsi untuk mengakses database MySQL.
- Cara kedua adalah dengan **object oriented style**. Dengan cara ini, kita menggunakan aturan penulisan pemrograman objek untuk berkomunikasi dengan MySQL.

Kedua jenis style pemrograman **mysqli** ini menggunakan nama fungsi dan method yang kurang lebih sama. Sebagai contoh, di dalam *procedural style* **mysqli**, terdapat fungsi **mysqli_query(),** sedangkan dalam *OOP style* **mysqli**, kita menggunakan method **$mysqli->query()**. Kita akan membahas lebih dalam tentang cara penggunaan 2 style ini nantinya.

Di dalam Praktikum PHP MySQL kali ini kita telah membahas sekilas mengenai perbedaan extension mysql dengan extension mysqli yang lebih baru. Sesuai dengan instruksi dari manual MySQL, sebaiknya kita tidak menggunakan lagi extension mysql.

Dalam Praktikum selanjutnya, kita akan mulai membahas cara penggunaan mysqli extension.

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

Dalam Praktikum PHP MySQL kali ini kita akan membahas cara menampilkan data MySQL dengan menggunakan mysqli extension. Karena mysqli memiliki 2 cara penggunaan: **procedural** dan **object-oriented**, pada Praktikum ini kita akan fokus pada penggunaan ***procedural style***.

Secara garis besar, cara penggunaan **mysqli** secara procedural (menggunakan fungsi-fungsi) sangat mirip dengan *mysql extension*. Karena itu jika anda sudah memahami cara pemakaian *mysql extension* (yang telah kita bahas), tidak akan kesulitan untuk beralih ke mysqli.

Dalam Praktikum ini saya tidak akan membahas secara detail mengenai fungsi-fungsi yang ada, karena sebagian besar sangat mirip dengan fungsi yang telah kita pelajari di **mysql extension**. Jika anda merasa sedikit bingung dengan penjelasan disini, silahkan mempelajari kembali Praktikum PHP MySQL dari part 4 s/d 11.

Cara Membuat Koneksi MySQL Dengan Fungsi mysqli_connect()

Untuk membuat koneksi antara PHP dengan MySQL Server, kita menggunakan fungsi **mysqli_connect()**. Fungsi *mysqli_connect()* ini membutuhkan argumen yang sama dengan fungsi *mysql_connect()*, yakni: **alamat host, nama user,** dan **password user.**

Sebagai contoh, untuk masuk kedalam MySQL di **localhost** menggunakan user **root** dan dengan password "**qwerty**", kita menggunakan kode program sebagai berikut:

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

?>
```

Sebagai fitur tambahan, fungsi *mysqli_connect()* memiliki argumen ke-4 yang bersifat opsional. Kita bisa menambahkan **nama database** yang ingin digunakan. Misalkan kita ingin

sewaktu proses koneksi dengan MySQL, langsung memilih database "*universitas*", maka contoh penulisannya adalah sebagai berikut:

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

Cara Menutup Koneksi Dengan Fungsi mysqli_close()

Walaupun PHP akan menutup koneksi dengan MySQL pada saat halaman selesai diproses, kita bisa menutupnya secara manual menggunakan fungsi **mysqli_close()**. Fungsi ini membutuhkan 1 argumen, yakni variabel *resources* hasil pemanggilan fungsi *mysqli_connect()*.

Berikut contoh penggunaan fungsi *mysqli_close()*:

```
<?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);

?>
```

Cara Menjalankan Query MySQL Dengan Fungsi mysqli_query()

Cara menjalankan query MySQL dengan *mysqli extension* juga mirip dengan *mysql extension*. Di dalam mysqli, kita menggunakan fungsi **mysqli_query()**. Namun berbeda dengan fungsi *mysql_query()* yang meletakkan variabel *resources* hasil **mysql_connect()** sebagai argumen kedua dan bersifat opsional, di dalam fungsi *mysqli_query()*, variabel ini diletakkan sebagai argumen pertama dan harus ditulis.

Sebagai contoh, dengan menggunakan *mysql_extension*, kita biasa menulis kode PHP sebagai berikut:

```
<?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);
?>
```

Sedangkan dengan menggunakan *mysqli extension*, berikut adalah perubahan cara penggunaannya:

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

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

Perhatikan bahwa kita menggunakan format: **mysqli_query($link, "query_MySQL")** untuk menjalankan query MySQL.

Cara Menampilkan Data MySQL Dengan mysqli

Untuk menampilkan hasil query MySQL, *mysqli extension* memiliki banyak fungsi. Kali ini kita akan membahas 3 fungsi yang paling sering digunakan: **mysqli_fetch_row(), mysqli_fetch_array(),** dan **mysqli_fetch_object()**.

Ketiga fungsi ini hampir sama cara penggunaannya dengan versi *mysql extension* yang telah kita bahas, yakni: *mysql_fetch_row(), mysql_fetch_array(),* dan *mysql_fetch_object()*. Jika anda perhatikan, perbedaan namanya hanya dengan menambah huruf "**i**".

Langsung saja kita masuk ke contoh program, untuk memahami cara penggunaan ketiga fungsi ini. Di dalam contoh-contoh berikut saya menggunakan tabel **mahasiswa_ilkom** di dalam database **universitas**. Tabel ini kita buat sewaktu membahas tentang fungsi mysql_fetch_row.

Pertama, mari kita lihat cara menampilkan data MySQL dengan fungsi **mysqli_fetch_row():**

```
<?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 />";
   }
?>
```

Fungsi **mysqli_fetch_row()** sama persis cara penggunaanya dengan fungsi **mysql_fetch_row()**, oleh karena itu saya tidak akan membahasnya lagi. Tapi jika anda ragu, bisa mengunjungi Praktikumnya di Cara Menampilkan Tabel MySQL dari PHP dengan fungsi mysql_fetch_row.

Berikutnya, kita bahas cara menampilkan data MySQL dengan fungsi **mysqli_fetch_array():**

```
<?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 />";
   }
```

```
?>
```

Sedikit perbedaan antara fungsi **mysqli_fetch_array()** dengan fungsi "kembarannya" **mysql_fetch_array()**, kali ini string penentu metoda array berada pada argumen kedua. Argumen pertama adalah variabel resources hasil pemanggilan fungsi mysqli_query(). String **MYSQLI_ASSOC** digunakan agar key array dapat diakses dengan nama. String lain yang bisa kita gunakan adalah **MYSQL_NUM** dan **MYSQL_BOTH**. Perbedaan ketiganya telah kita bahas lengkap dalam Praktikum PHP MySQL: Cara Menampilkan Tabel MySQL dari PHP dengan mysql_fetch_array.

Cara ketiga yang akan kita bahas adalah dengan menggunakan fungsi **mysqli_fetch_object()**. Fungsi ini sama dengan fungsi **mysql_fetch_object()**, dimana kita mengakses hasil query sebagai objek. Berikut adalah contoh kode PHPnya:

```
<?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 />";
  }
?>
```

Jika anda butuh penjelasan mengenai cara penggunaan **mysqli_fetch_object()**, silahkan kunjungi Praktikum PHP MySQL : Cara Menampilkan Tabel dengan objek (mysql_fetch_object).

Dalam Praktikum PHP MySQL kali ini kita telah membahas tentang cara penggunaan mysqli untuk menampilkan data dari database MySQL.

Seperti yang pernah kita bahas, **mysqli** memiliki 2 cara penulisan (2 jenis style). Selain dengan cara procedural seperti dalam Praktikum ini, di dalam Praktikum selanjutnya kita akan membahas cara penggunaan mysqli dengan object oriented style.

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

Jika dalam Praktikum sebelumnya kita mempelajari mysqli dengan menggunakan fungsi-fungsi atau dikenal dengan procedural style mysqli, dalam Praktikum PHP MySQL kali ini kita akan membahas tentang object style mysqli.

Cara Penulisan Object Style mysqli

Sebagai cara koneksi yang lebih baru daripada *mysql extension*, **mysqli** memiliki 2 jenis style, yakni **procedural style** dan **object-oriented style**. Kecendrungan pemrograman saat ini lebih banyak menggunakan objek. Untuk hal inilah PHP juga menyediakan mysqli dengan "rasa" objek.

Pemrograman berbasis objek lebih banyak disukai karena cenderung rapi dan mudah dikembangkan, terutama jika kita mengerjakan proyek besar yang butuh ribuan baris program. Konsep OOP seperti ***inheritance, encapsulation*** dan ***polymorfism*** membuat program menjadi lebih tertata.

Jika anda baru berkenalan dengan PHP atau konsep pemrograman secara umum, pemrograman berbasis objek mungkin akan terasa membingungkan. OOP membutuhkan alur penulisan program yang berbeda dibandingkan metode procedural yang kita pelajari hingga saat ini.

Saya sangat menyarankan anda untuk mulai beralih menggunakan OOP. Karena dengan OOP-lah tingkat pemahaman dan pengalaman kita "naik level". Jika anda bermaksud "serius" mempelajari tentang PHP, duniailkom telah membuat Praktikum lengkap mengenai konsep OOP, di dalam Praktikum Pemrograman Berbasis Objek PHP.

Langsung saja kita lihat bagaimana cara penggunaan objek di dalam mysqli.

Didalam Praktikum ini saya berasumsi anda telah memahami sedikit tentang OOP PHP. Terutama cara pemanggilan **constructor** (cara pembuatan objek) dan cara pemanggilan **method** objek (menggunakan tanda panah ->).

Cara Membuat Koneksi dengan MySQL (mysqli constructor)

Untuk membuat koneksi MySQL dengan PHP menggunakan **mysqli object style**, caranya adalah dengan menggunakan **mysqli constructor**. Constructor mysqli adalah sejenis fungsi yang digunakan untuk membuat object baru dari ***class mysqli***.

Argumen di dalam construktor mysqli ini sama persis dengan argumen fungsi **mysqli_connect()**, yakni lokasi komputer, nama user, dan password user.

Berikut adalah cara membuat koneksi mysqli di **localhost** untuk user "**root**" dengan password "**qwerty**":

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

Pada kode diatas, saya membuat objek **$link** dari class **mysqli**. Di dalam OOP, keyword **new** digunakan untuk membuat objek baru. Dengan objek **$link** hasil *mysqli constructor* inilah proses query MySQL nantinya kita jalankan.

Sama seperti fungsi **mysqli_connect()**, *constructor mysqli* juga memiliki argumen tambahan, yakni nama database yang ingin digunakan. Jika kita ingin langsung mengakses database "**universitas**", maka kode programnya menjadi:

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

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

Walaupun PHP akan lansung menutup koneksi ke MySQL jika halaman telah selesai di proses, namun kita juga bisa menutupnya secara manual menggunakan method **close()** dari objek **mysqli,** atau biasa ditulis sebagai **mysqli::close()**. Berikut contoh penggunaannya:

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

```
    //.... Proses PHP
    //.... Proses PHP
    //.... Proses PHP


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

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

Jika di dalam penulisan procedural kita menggunakan fungsi **mysqli_query()** untuk menjalankan query MySQL, maka di dalam objek style, kita mengaksesnya menggunakan method **mysqli::query()**.

Agar lebih mudah dipahami, langsung saja kita masuk kedalam contoh kode program:

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


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

Perhatikan bahwa di dalam proses pemanggilan method **query()**, kita membutuhkan 1 argumen, yakni query MySQL yang akan dijalankan.

Cara Menampilkan Data MySQL Dengan mysqli object style

Untuk menampilkan hasil query MySQL, *mysqli object* memiliki banyak method. Kali ini kita akan membahas 3 cara yang paling sering digunakan, yakni method **fetch_row(), fetch_array()** dan **fetch_object()**. Cara penggunaan ketiga method ini hampir sama dengan padanan fungsinya di procedural style, yakni dengan fungsi **mysqli_fetch_row(), mysqli_fetch_array(),** dan **mysqli_fetch_object().**

Agar lebih jelas, langsung saja kita masuk ke dalam contoh kode program untuk menampilkan seluruh isi data dari tabel *mahasiswa_ilkom*.

Pembahasan pertama, yakni cara menampilkan data mysqli dengan method **fetch_row().** Berikut adalah contoh kode programnya:

```
<?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 />";
   }
?>
```

Dalam contoh diatas, saya mengunakan method **fetch_row()** dari objek **$result** yang merupakan objek hasil pemanggilan method **query()**. Method **fetch_row()** ini tidak memerlukan argumen apapun, sehingga kita memanggilnya dengan cara **$result->fetch_row().**

Cara kedua untuk menampilkan data mysqli adalah dengan method **fetch_array()**. Berikut contoh kode programnya:

```
<?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 />";
    }
?>
```

Sama seperti method **fetch_row()**, method **fetch_array()** juga diakses dari objek **$result** yang merupakan hasil pemanggilan query.

Method **fetch_array()** membutuhkan 1 argumen, berupa string yang berisi bagaimana cara array hasil method akan diakases. String ini bisa dipilih dari salah satu nilai: **MYSQLI_NUM, MYSQLI_ASSOC** atau **MYSQLI_BOTH**. Lebih lengkap tentang perbedaan ketiga nya telah kita bahas dalam Praktikum mengenai fungsi mysql_fetch_array().

Cara ketiga yang akan kita bahas untuk menampilkan data mysqli, adalah dengan method **fetch_object().** Berikut contoh pengunaanya:

```
<?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 />";
   }
?>
```

Cara penggunaan method **fetch_object()** hampir sama dengan method-method sebelumnya, yakni di akses dari objek **$result** hasil menjalankan query MySQL.

Dalam Praktikum PHP MySQL kali ini kita telah membahas cara penggunaan mysqli extension yang berbasis object. Praktikum mysqli ini dan juga Praktikum sebelumnya tentang mysqli procedural saya buat sesingkat mungkin tanpa menggunakan prosedur untuk penanganan kesalahan. Misalnya, bagaimana mengecek apakah koneksi dengan MySQL sudah terhubung atau belum.

Dalam Praktikum berikutnya, akan kita bahas secara mendalam mengenai beberapa fungsi dan method mysqli yang digunakan untuk penanganan kesalahan.