

# Batching di PHP

**Terlalu banyak data yang harus ditampilkan? Gunakan *batching* untuk mempercepat dan sekaligus memperindah.**

**D**ata yang semakin bertambah tentunya dapat dimaklumi. Permasalahannya, sebagai seorang *developer*, kita dituntut untuk menampilkan data tersebut dalam bentuk yang enak dibaca dan alangkah baiknya kalau data tersebut dapat ditampilkan dalam waktu yang singkat. Tentunya akan menjadi sangat tidak indah apabila ada 100 baris yang ditampilkan sekaligus di dalam sebuah halaman laporan. Dalam konteks pemrograman web, tentunya laporan tersebut akan menjadi lebih lambat untuk ditampilkan. Solusinya? Tampilkan laporan tersebut dalam halaman-halaman yang terpisah (*batching*), dengan jumlah data tetap perhalaman.

Batching sendiri sangat umum ditemukan pada daftar posting di forum ataupun daftar e-mail di *mailbox*. Alih-alih menampilkan keseluruhan posting atau e-mail, nomor-nomor halaman yang bisa diklik ditampilkan untuk kita. Dan ketika salah satu nomor halaman tersebut diklik, maka kita pun akan disuguhi data dari halaman yang bersangkutan. Tertantang membuat sendiri menggunakan PHP?

Pertama-tama, asumsikan kita telah memiliki tabel *a* di MySQL dengan struktur seperti pada tabel 1.

Dan asumsikan pula tabel tersebut telah memiliki 10 *record*, dengan *field id* berisikan angka 1 sampai 10, dan *field* nama berisikan *bla 1* sampai *bla 10*. Record-record ini akan kita tampilkan dalam tampilan per halaman, di mana jumlah data per halaman dapat diatur sesuai keinginan.

Tutorial ini akan ditampilkan dalam langkah-langkah berikut:

**1. Melakukan koneksi ke database server**  
Langkah ini adalah langkah wajib. Untuk bekerja dengan MySQL, kita harus

terhubung dulu dengan database server. Setelah itu, kita perlu memilih database di mana tabel-tabel kita berada di dalamnya.

```
mysql_connect("localhost","nop","nop")
or die("gagal konek");
mysql_select_db("nop") or die("gagal
select db");
```

fungsi *die()* pada contoh tersebut berfungsi sebagai penanganan kesalahan.

## 2. Memberikan query

Dalam menampilkan data, tentunya kita harus mengambil data terlebih dahulu. Di MySQL, kita bisa memberikan query ke database server dengan perintah berikut ini:

```
$query = "select * from a";
$result = mysql_query($query);
```

Selanjutnya, variabel *\$result* inilah yang akan menampung data hasil query.

## 3. Penentuan variabel-variabel batching

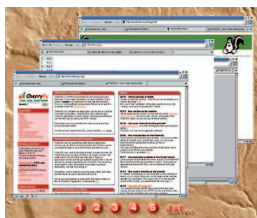
Berapa jumlah data yang didapatkan? Berapakah jumlah data per halaman yang ingin ditampilkan? Berapa jumlah halaman yang kita miliki? Semua hal tersebut akan kita atur di langkah ini.

```
$count = mysql_affected_rows();
$batch = 4;
$pages = ceil(($count / $batch));
if (!$page) $page = 1;
```

Dalam konteks ini, fungsi *mysql\_affected\_rows()* berguna untuk mendapatkan jumlah baris yang berhasil didapatkan. Jumlah baris tersebut adalah patokan bagi kita untuk menentukan jumlah halaman. Jumlah baris tersebut kita masukkan ke dalam variabel *\$count*.

Table 1. Struktur Tabel a

Field	type	Null	Key	Default	Extra
id	int(11)		PRI	NULL	auto_increment
nama	varchar(50)				



Selanjutnya, kita perlu menentukan jumlah data per halaman. Dalam hal ini, kita menentukan 4 data per halaman yang kita masukkan ke dalam variabel *\$batch*. Dan, jumlah halaman, yang kita masukkan dalam variabel *\$pages* adalah hasil bagi antara jumlah data dan jumlah *batch* (jumlah data per halaman). Jangan lupa untuk menggunakan fungsi *ceil()* untuk pembulatan ke atas.

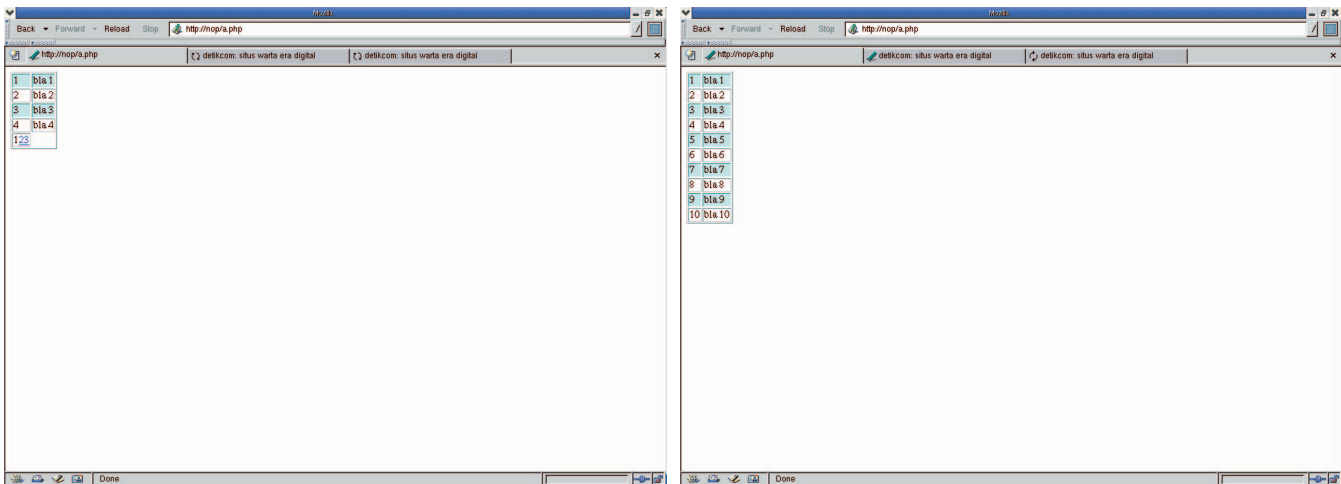
Variabel *\$page* adalah variabel yang dilewatkan bersama URL, yang berfungsi sebagai patokan untuk menampilkan halaman yang bersesuaian. Artinya, apabila *\$page* bernilai 1, maka halaman yang akan ditampilkan adalah halaman 1, yang mana akan berisi data nomor 1 sampai nomor 4. Apabila *\$page* bernilai 2, maka halaman yang akan ditampilkan adalah halaman 2, yang mana akan berisi data nomor 5 sampai nomor 8. Dan seterusnya. Karena pada pemanggilan pertama skrip mungkin tidak melewati variabel *\$page*, sehingga variabel ini tidak terdefinisi, maka kita melakukan tindakan antisipatif dengan mengasumsikan variabel *\$page* bernilai 1 apabila variabel *\$page* tidak terdefinisi.

## 4. Menentukan data pertama dan data perhalaman

Setelah variabel-variabel batching umum selesai diinisialisasi, maka tiba saatnya bagi kita untuk menentukan data pertama dan data per halaman, yang nantinya siap untuk dikirimkan di *web browser*.

```
$start = ($page-1) * $batch;
$query = "select * from a LIMIT
$start,$batch";
$result = mysql_query($query);
if (!mysql_affected_rows()) die("None");
```

Data pertama per halaman, yang diasosiasikan dengan variabel *\$start* adalah hasil kali antara isi variabel *\$page* dikurangi satu dan isi variabel



▲ Lengkap dengan nomor-nomor halaman

▲ Tanpa nomor halaman untuk data 1 halaman

\$batch. Pengurangan dengan satu diperlukan karena urutan diawali dengan nol, dan bukannya 1, termasuk nomor halaman dan data yang didapatkan dari MySQL.

Dengan demikian, apabila halaman yang diberikan adalah halaman 2, maka data pertama dihalaman 2 tersebut adalah data nomor (2-1) \* 4, yaitu data nomor 4. Maka pada *query* berikutnya, kita harus mencari data dari data nomor 4. Harap diperhatikan bahwa kita meminta *query* ke database server sesuai dengan data yang ingin ditampilkan, sehingga daripada mengambil semua data pada satu waktu dan memisahkannya kemudian per halaman, kita memilih mengambil data per halaman. Hal ini berfungsi untuk mengantisipasi apabila jumlah data terlalu besar.

Setelah menentukan awal data per halaman, barulah kita memberikan *query* ke database server yang berisikan data awal dan jumlah data yang ingin diminta. Penggunaan kunci LIMIT sangat membantu dalam hal ini. LIMIT sendiri diberikan bersama padanan awal data dan jumlah data. Jumlah data dalam hal ini adalah isi dari variabel \$batch. Sekali lagi, apa yang didapatkan dari MySQL dimasukkan ke dalam variabel \$result.

Karena variabel \$page dilewatkan bersama URL, maka ada kemungkinan user yang nakal akan melewati nilai-nilai yang tidak masuk akal, misalnya

\$page = -1, \$page = 1000000000 ataupun nilai-nilai lain yang tidak berarti. Untuk mengantisipasi, kita akan menguji nilai yang dikembalikan oleh fungsi mysql\_affected\_rows(). Apabila tidak ada nilai yang dikembalikan (dan kondisi ini hanya terjadi apabila user yang nakal memasukkan nilai yang tidak masuk akal), maka kita akan mencetak tulisan *None* dan menghentikan proses lainnya.

### 5. Menampilkan data

Kemudian, saatnya bagi kita untuk menampilkan data per halaman. Data yang akan kita tampilkan akan sangat bergantung kepada nomor halaman yang diberikan. Untuk memperindah, kita akan menampilkan dalam tabel.

```
echo("< table border = 1 >");
$_item = 0;
while ( $line = mysql_fetch_array($result) )
{
    if ( $_item % 2 ) $bgcolor = "#FFFFFF";
    else $bgcolor = "#AFFFFF";
    $_item ++;
    echo("< tr bgcolor = $bgcolor >");
    echo("< td > $line[0] < /td >");
    echo("< td > $line[1] < /td >");
    echo("< /tr >");
    $start ++;
}
```

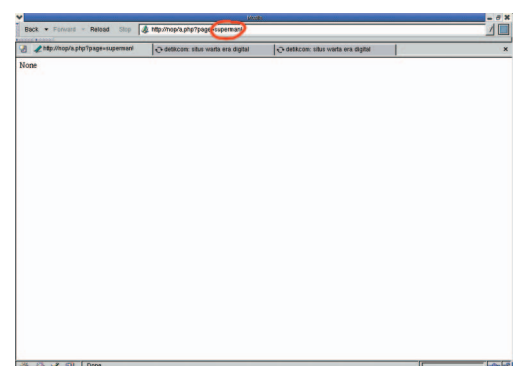
Kita menggunakan fungsi mysql\_fetch\_array() untuk mengambil data di variabel \$result. Variabel \$\_item diperbantukan untuk mengetahui apakah baris yang sedang diproses adalah baris

ganjil atau genap karena kita akan memberikan warna baris yang berbeda untuk baris ganjil dan genap. Hal tersebut semata-mata agar lebih enak dibaca oleh mata.

Fungsi mysql\_fetch\_array() akan mengembalikan *array*. Anda dapat mengambil isi dari *array* tersebut dengan mengakses indeksnya.

### 6. Membuat nomor halaman

Sampai saat ini, kita telah berhasil menampilkan data untuk halaman 1. Anda bisa menambahkan page=x di URL untuk berpindah ke halaman x. Sehingga apabila skrip PHP Anda memiliki nama a.php, maka pemberian url a.php?page=3 akan membawa Anda ke halaman 3. Namun, hal ini tentunya tidak diinginkan oleh semua orang karena seperti yang telah ditulis di awal tulisan, kita harus menampilkan nomor halaman yang tersedia sehingga user hanya perlu mengklik pada nomor halaman tertentu



▲ Antisipasi untuk page yang tidak masuk akal

untuk membawanya ke halaman tersebut.

```
if ($pages > 1)
{
    echo("<tr>");
    echo("<td>");
    for ($i=1;$i<=$pages;$i++)
    {
        if ($i != ceil($start/$batch))
        {
            echo("<a href=a.php?page=
            $i>$i</a>");
        }else
        {
            echo("$i");
        }
    }
    echo("</td>");
    echo("</tr>");
}
echo("</table>");
```

Pengecekan kondisi if (\$pages > 1) dimaksudkan untuk menampilkan halaman yang tersedia hanya apabila jumlah halaman lebih dari 1. Tentunya sangat aneh apabila kita menampilkan

nomor halaman 1 sementara data yang kita miliki hanya perlu ditampilkan dalam 1 halaman.

Untuk setiap halaman yang tersedia, kita akan menampilkan nomor halamannya dengan pengecualian nomor halaman yang aktif tidak dapat di klik. Hal ini juga dimaksudkan untuk memperindah tampilan. Perulangan *for* pada blok kode tersebut dimaksudkan untuk menampilkan nomor halaman, sementara kondisi *if* di dalam blok *for* tersebut dimaksudkan untuk mengecek halaman aktif. Kini Anda dapat menikmati halaman web yang lebih indah.

#### Source code selengkapnya:

```
<?
mysql_connect("localhost","nop","nop")
or die ("gagal konek");
mysql_select_db("nop") or die("gagal
select db");

$query = "select * from a";
$result = mysql_query($query);

$count = mysql_affected_rows();
```

```
$batch = 4;
$pages = ceil(($count / $batch));
if (!$page) $page = 1;

$start = ($page-1) * $batch;
$query = "select * from a LIMIT
$start,$batch";
$result = mysql_query($query);

if (!mysql_affected_rows()) die("None");

echo("<table border=1>");
$item = 0;
while ($line = mysql_fetch_array($result))
{
    if ($item % 2) $bgcolor =
    "#FFFFFF"; else $bgcolor =
    "#AFFFFF";
    $item++;
    echo("<tr bgcolor=$bgcolor>");
    echo("<td>$line[0]</td>");
    echo("<td>$line[1]</td>");
    echo("</tr>");
    $start++;
}
```

```
if ($pages > 1)
{
    echo("<tr>");
    echo("<td>");
    for ($i=1;$i<=$pages;$i++)
    {
        if ($i != ceil($start/$batch))
        {
            echo("<a href=a.php?
            page=$i>$i</a>");
        }else
        {
            echo("$i");
        }
    }
    echo("</td>");
    echo("</tr>");
}
echo("</table>");
?>
```

Penggunaan batching adalah mutlak apabila data yang harus ditampilkan sangatlah banyak. Paling tidak, selain lebih enak dilihat, kita pun bisa membantu untuk mengurangi waktu tunggu *user*.<sup>1</sup>

Noprianto ([noprianto@infolinux.co.id](mailto:noprianto@infolinux.co.id))

Tinggalkan Gaya Komunikasi Konvensional Selama Ini,  
Tingkatkan Image Branding Perusahaan Anda Dengan Identitas Baru  
Dan Komunikasi Memanfaatkan Teknologi Internet !

**UNLIMITED**  
WEB HOSTING

**Rp. 1.800.000** \*per tahun

Kapasitas Tak Terbatas  
POP/IMAP Account Tak Terbatas  
Mail Forward, Alias, Mailing List Manager  
RedHat Linux, MySQL Database, Qmail Mail Server  
Plesk 5.02 Control Panel  
CGI Perl, PHP 4, Server Side Include  
FTP, FrontPage Access  
Webmail, File Manager, phpMyAdmin  
Dll.

**namadomain.com**  
Telp.: 021-56 999 172 Fax: 021-56 999 167  
Email: [info@namadomain.com](mailto:info@namadomain.com)