

CodeIgniter Framework



IT507 - Rekayasa Web

CodeIgniter (CI) adalah suatu framework pengembangan aplikasi web berbasis PHP. CI menyediakan banyak library sehingga memungkinkan mengembangkan aplikasi dengan lebih cepat. Salah satu keunggulan CI dibanding framework lain adalah kesederhanaan penggunaannya dan kecepatan eksekusinya. Pada modul 2 ini akan mengenalkan pembaca kepada pengembangan aplikasi yang menggunakan framework CI, khususnya proses retrieve, insert, update, dan delete data dari database.





CodeIgniter Framework

Modul 2

Materi	Manipulasi data pada database, Database class di CI, Active Record.
Target	Mampu membuat aplikasi pengelolaan dan manipulasi data.
Pre requisite	Mengerti materi di Modul 1 Menguasai query DDL dan DML.



A. Manipulasi Data pada Database

Suatu aplikasi berbasis web, umumnya menggunakan database sebagai media penyimpanan data. CodeIgniter menyediakan banyak fungsi yang mendukung manipulasi data pada database, baik menggunakan metode tradisional maupun pola *Active Record*.

Active record adalah suatu metode untuk mengakses data pada database relasional, suatu tabel dan view dipetakan menjadi suatu class. Metode ini biasanya digunakan oleh *tools* ORM (Object Relational Mapping). Beberapa framework PHP sudah menyertakan modul ORM yang mengimplementasikan pola Active Record yang mendukung relationship dan validasi, contohnya Symfony, CakePHP dan Yii.

Pada CodeIgniter terdapat fitur untuk membangkitkan *Query* yang bernama “ActiveRecord”, tapi CI tidak menerapkan pola Active Record yang sebenarnya, hanya sebagai pembangkit query saja agar aplikasi yang dibangun dapat mendukung multi database. Untuk mengimplementasikan fungsi Active Record yang sebenarnya pada CI dapat menggunakan library tersendiri seperti CodeIgniter DataMapper atau CodeIgniter Gas ORM.

1. CodeIgniter Database Class

Agar aplikasi yang dibangun dapat terkoneksi dengan database, harus tentukan dulu parameter koneksi database pada file konfigurasi yang sudah dibahas pada Modul 1. Setelah semua parameter koneksi ditentukan, gunakan script di bawah ini untuk menggunakan Database Class CodeIgniter :

```
$this->load->database();
```

Setelah Database Class dipanggil menggunakan script di atas, maka kita dapat mengeksekusi query pada database tersebut.

Jika setiap halaman aplikasi yang dibangun membutuhkan koneksi ke database, sebaiknya Database Class dibuat *autoload* saja dengan menambahkan library “database” pada konfigurasi *autoload**. Seperti contoh di bawah ini:

```
$s $autoload['libraries'] = array('database');
```

*Cara konfigurasi file autoload dapat dilihat pada Modul 1 point B.4.b.

2. Eksekusi Query

Terdapat dua cara untuk mengeksekusi query, yaitu dengan metode “tradisional” dan dengan fitur Active Record.

a. Tradisional

Sintaks query dieksekusi oleh method **query()** pada class Database, intansiasi class Database tersebut (point A.1.) menjadi object **db**. Berikut contohnya:

```
$query = $this->db->query("SELECT name, title, email FROM my_table");
```

Sintaks query lengkap *select* data di atas dimasukkan ke method **query()**. Hasil eksekusinya dapat diolah berupa **object** atau **array**, sesuai yang kita tentukan. Lihat point A.3.

b. Active Record

Dengan menggunakan Active Record, developer dapat memanipulasi data dengan sederhana dan script yang seminimal mungkin. CodeIgniter *men-generate* sintaks query berdasar method Active Record yang developer panggil, dan juga query yang di-*generate* dipastikan didukung oleh DBMS yang digunakan. Contohnya query *select* yang menggunakan **limit** antara MySQL dan Oracle berbeda, Active Record otomatis menyesuaikan. Berikut contohnya:

```
$query = $this->db->get('table_name');
```

Dengan menggunakan method **get()** Active Record, akan menghasilkan sintaks query *select* terhadap tabel yang ditentukan pada parameter method **get()** tersebut.

Berikut beberapa method Active Record yang biasanya dipakai:

- `Select()`
- `From()`
- `Where()`
- `Get_where()`
- `Like()`
- `Limit()`
- `Order_by()`
- `Insert()`
- `Update()`
- `Delete()`

Lebih jelas mengenai cara penggunaan method-method Active Record dapat dilihat pada [user_guide](#) menu Database Class.

Hasil eksekusi query (tradisional maupun Active Record) yang pada contoh disimpan pada variabel **\$query**, **selalu bertipe object**.

Mengolah hasil query dapat dilihat pada point A.3. berikut ini.

3. Mengambil Hasil Query

Terdapat dua tipe **query result set** yang dapat diolah dari hasil eksekusi query, yaitu bertipe **object** dan bertipe **array**.

a. Result set tipe Object

Field-field hasil query *select* menjadi **properti** pada suatu **object**. Kita dapat menggunakan method **result()** pada variabel penampung hasil query, pada contoh sebelumnya **\$query**. Setiap method **result()** dipanggil maka akan mengembalikan **1 baris record data** bertipe object, dengan properti-properti yang merupakan nama field hasil query. Berikut contohnya:

```
$query = $this->db->get('tabel_buku');  
  
foreach($query->result() as $row)  
{  
    echo $row->Judul;  
    echo $row->Pengarang;  
}
```

Foreach digunakan untuk menyusuri hasil query, mulai dari record pertama sampai terakhir.

b. Result set tipe Array

Field-field hasil query *select* menjadi **index** pada suatu variabel **array**. Kita dapat menggunakan method **result_array()** pada variabel penampung hasil query, pada contoh sebelumnya **\$query**. Setiap method **result_array()** dipanggil maka akan mengembalikan **1 baris record data** bertipe array, dengan index-index yang merupakan nama field hasil query. Berikut contohnya:

```
$query = $this->db->get('tabel_buku');  
  
foreach($query->result_array() as $row)  
{  
    echo $row['Judul'];  
    echo $row['Pengarang'];  
}
```

Foreach digunakan untuk menyusuri hasil query, mulai dari record pertama sampai terakhir.

Developer bebas menentukan tipe *result set* sesuai kebutuhan. Tipe **array** biasanya cocok jika nama field dinamis tergantung kondisi tertentu yang juga dinamis, karena lebih mudah memanipulasi nama index array dibanding properti suatu objek.

B. Latihan

Latihan di Modul 2 ini akan melanjutkan studi kasus pada Modul 1, yaitu pengelolaan data Buku. Diketahui atribut data buku sebagai berikut:

Field	Tipe Data	Ukuran
ID_Buku	Integer	8, auto increment
Judul	Varchar	100
Pengarang	Varchar	150
Kategori	Varchar	10

Selanjutnya ikuti langkah-langkah berikut:

1. Buat Database

Bukalah phpmyadmin melalui browser, <http://localhost/phpmyadmin>. Kemudian buatlah database baru dengan nama **buku_NRP**. Contoh: **buku_123040024**.

2. Import struktur Tabel Buku

Download file [mst_buku.sql](#) dari modul online, kemudian import ke database yang sudah dibuat sebelumnya. Sehingga seperti contoh berikut:

	Field	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/>	ID_Buku	int(20)			No	None	auto_increment
<input type="checkbox"/>	Judul	varchar(255)	latin1_general_ci		No	None	
<input type="checkbox"/>	Pengarang	varchar(255)	latin1_general_ci		No	None	
<input type="checkbox"/>	Kategori	varchar(30)	latin1_general_ci		No	None	

Gambar B.2. Struktur Tabel mst_buku

3. Konfigurasi Koneksi Database

Masuklah ke direktori `/application/config`, lalu buka file `database.php`. Tentukan hostname, username, password, dan nama database. Contoh:

```
50 $db['default']['hostname'] = 'localhost';
51 $db['default']['username'] = 'root';
52 $db['default']['password'] = '';
53 $db['default']['database'] = 'buku_123040024';
54 $db['default']['dbdriver'] = 'mysql';
```

Gambar B.3. Konfigurasi Koneksi Database

Pada kuliah ini username yang digunakan untuk koneksi ke DBMS MySQL adalah **root**. Jika pada pembangunan aplikasi yang sebenarnya, **sangat disarankan** tidak memakai user **root** untuk koneksi suatu aplikasi. **Buatlah user spesifik yang memiliki hak akses terhadap database yang spesifik pula.**

4. Konfigurasi Autoload

Masuklah ke direktori `/application/config`, lalu buka file `autoload.php`. Tambahkan library database pada konfigurasi autoload. Seperti contoh berikut:

```
55 $autoload['libraries'] = array('database');
```

Gambar B.4. Konfigurasi Autoload Library Database

5. Buat model Buku

Buatlah sebuah file menggunakan editor PHP, ketikkan script model Buku di bawah ini:

```
1 <?php
2
3 class Buku_m extends CI_Model
4 {
5     function __construct()
6     {
7         parent::__construct();
8     }
9
10    function get_records($criteria='', $order='', $limit='', $offset=0)
11    {
12        $this->db->select('*');
13
14        $this->db->from('mst_buku');
15
16        if ($criteria != '')
17            $this->db->where($criteria);
18
19        if ($order != '')
20            $this->db->order_by($order);
21
22        if ($limit != '')
23            $this->db->limit($limit, $offset);
24
25        $query = $this->db->get();
26
27        return $query;
28    }
29
30    function insert($data)
31    {
32        $query = $this->db->insert('mst_buku', $data);
33
34        return $query;
35    }
36
37    function update_by_id($data, $id)
38    {
39        $this->db->where("ID_Buku = '$id'");
40
41        $query = $this->db->update('mst_buku', $data);
42
43        return $query;
44    }
45 }
```

Script bersambung ke halaman berikutnya.

```

45
46     function delete_by_id($id)
47     {
48         $query = $this->db->delete('mst_buku', "ID_Buku = '$id'");
49
50         return $query;
51     }
52 }
53
54 ?>

```

Simpan file tersebut di direktori kerja Anda, pada folder : **application/models/**

Beri nama file : **buku_m.php**

Penjelasan:

```
class Buku_m extends CI_Model
```

- Nama class harus diawali huruf kapital.
- Semua model yang dibuat harus turunan dari model CI (**CI_Model**)
- Karena tidak ada aturan penamaan model pada CodeIgniter, disarankan beri ciri-ciri di nama class yang menandakan bahwa class tersebut adalah model, untuk memudahkan proses *debuging*. Contoh di atas **_m**.
- Lebih detail mengenai model : user_guide/general/models.html

```
function get_records($criteria='', $order='', $limit='', $offset=0)
{
```

- Method yang akan berfungsi melakukan query *select* data.
- Terdapat 4 parameter *optional* sebagai input bagi method, yaitu
 - 1) **\$criteria**. Berfungsi sebagai kriteria di query, " where 1=1 and"
 - 2) **\$order**. Berfungsi sebagai *ordering* di query, " Judul ASC, Pengarang DESC ... "
 - 3) **\$limit**. Berfungsi sebagai angka limit (batas record) di query.
 - 4) **\$offset**. Berfungsi sebagai angka offset (record awal) di query.

```

$this->db->select('*');

$this->db->from('mst_buku');

```

- Active Record untuk mendefinisikan query *select* data.
- Script tersebut akan men-generate query:

```
select * from mst_buku
```

```

if ($criteria != '')
    $this->db->where($criteria);

```

- Jika kriteria query didefinisikan, maka gunakan Active Record untuk mendefinisikan kriteria query-nya untuk digenerate.
- Misal variabel **\$criteria** berisi "Judul like 'Laskar Pelangi'", maka akan di-generate:

```
select * from mst_buku where `Judul` like 'Laskar Pelangi'
```



```
if ($order != '')
    $this->db->order_by($order);
```

- Jika *ordering* ditentukan, maka gunakan Active Record `order_by()`.
- Misal variabel `$order` berisi "Judul" tanpa didefinisikan arah pengurutannya (ASC atau DESC), maka Active Record akan otomatis menambahkan ASC. Script yang di-generate:

```
select * from mst_buku where `Judul` like 'Laskar Pelangi%'
order by `Judul` ASC
```

```
$query = $this->db->get();
```

- Query yang sudah didefinisikan pada Active Record, dieksekusi dengan method `get()`
- Hasil eksekusi query disimpan ke variabel `$query`, untuk diolah kemudian.

6. Modifikasi controller Buku

Bukalah file controller Buku yang sudah Anda buat minggu lalu, kemudian lakukan modifikasi berikut ini:

a. Load helper URL dan model Buku_m

Pada *constructor*, tambahkan script untuk *load* helper URL dan model Buku_m.

Sehingga seperti berikut:

```
function __construct()
{
    parent::__construct();

    $this->load->helper('form');
    $this->load->helper('url');

    $this->data['opt_kategori'] = array('' => '- Pilih salah satu -',
                                      'novel' => 'Novel',
                                      'komik' => 'Komik',
                                      'kamus' => 'Kamus');

    $this->load->model('Buku_m');
}
```

Penjelasan:

- Helper URL digunakan di beberapa bagian pada controller **Buku**, contohnya function `redirect()`.
- Model **Buku_m** yang sudah dibuat harus dipanggil (instansiasi) agar dapat digunakan pada controller. Instansiasi model Buku_m, akan menjadi properti di class controller Buku yang namanya sama dengan nama model, yaitu **Buku_m**.

b. Modifikasi method index()

Pada Modul 1 method `index` berfungsi sebagai *function default* yang memanggil method `add_new()` jika pada URL tidak didefinisikan method yang dipanggil. Pada Modul 2 ini modifikasi method `index` yang dimaksudkan berfungsi mengambil data buku dari database kemudian menampilkan di view. Berikut ini script modifikasi method `index()`:

```
function index()
{
    $this->data['query'] = $this->Buku_m->get_records();

    $this->load->view('buku_v', $this->data);
}
```

Penjelasan:

```
$this->data['query'] = $this->Buku_m->get_records();
```

- Panggil method `get_records()` pada model `Buku_m`, untuk mengambil data buku di tabel.
- *Return value* dari method `get_records()` yang berupa hasil eksekusi query dimasukkan ke properti `data['query']` untuk dikirimkan ke view.

c. Buat method save()

Pada Modul 1 kita telah membuat form input data buku, dimana *action* form tersebut mengacu ke class `Buku` method `save`. Buatlah sebuah method baru bernama `save()` seperti di bawah ini:

```
function save($is_update=0)
{
    $data['Judul']      = $this->input->post('judul', true);
    $data['Pengarang']  = $this->input->post('pengarang', true);
    $data['Kategori']   = $this->input->post('kategori', true);

    if ($is_update==0)
    {
        // Jika tambah data baru
        if ($this->Buku_m->insert($data))
            redirect('buku');
    }
    else
    {
        // Jika update data
        $id = $this->input->post('id');

        if ($this->Buku_m->update_by_id($data, $id))
            redirect('buku');
    }
}
```

Penjelasan:

```
function save($is_update=0)
{
```

- Method yang akan berfungsi melakukan aksi *insert* atau *update* data, tergantung dari parameter `$is_update` yang diterimanya dari view (action dari form).
`$is_update == 0`, maka aksi **insert**.
`$is_update != 0`, maka aksi **update**.

```
$data['Judul'] = $this->input->post('judul', true);
```

- Mengambil data dari form yang dikirimkan menggunakan method `post`, dimasukkan ke variabel `$data['Judul']`. **Index** dari variabel `$data` tersebut merupakan nama field di tabel `mst_buku`, yang akan disimpan datanya.
- Parameter pertama `$this->input->post()` adalah **nama elemen** di form, parameter kedua bernilai **TRUE** mendefinisikan bahwa data harus difilter dahulu oleh class Security.
- Script tersebut sama dengan `$_POST['judul']`, namun cara konvensional seperti itu tanpa filter *security* dahulu.

```
if ($is_update==0)
{
    // Jika tambah data baru
    if ($this->Buku_m->insert($data))
        redirect('buku');
}
```

- Jika `$is_update` bernilai 0, maka tambah data baru (*insert*).
- Variabel `$data` yang merupakan penampung data dari form, dikirimkan sebagai parameter method `insert()` di model `Buku_m`.
- Jika proses *insert* data sukses, maka *redirect* browser ke controller `Buku`.

```
else
{
    // Jika update data
    $id = $this->input->post('id');

    if ($this->Buku_m->update_by_id($data, $id))
        redirect('buku');
}
```

- Jika `$is_update` tidak bernilai 0, maka update data lama.
- Ambil `ID_Buku` dari form, simpan ke variabel `$id`.
- Variabel `$data` yang merupakan penampung data dari form, dikirimkan sebagai parameter method `update_by_id()` di model `Buku_m`.
- Jika proses *update* data sukses, maka *redirect* browser ke controller `Buku`.

7. Buat view Daftar Buku

Buatlah file baru yang akan berfungsi sebagai *view* daftar buku, antarmukanya kurang lebih seperti ini:

[Tambah Buku](#)

No	Judul	Pengarang	Kategori	Aksi
1	Modul Kuliah Rekasaya Web - CodeIgniter Framework	Candra Utama	Novel	Edit Hapus

Scriptnya seperti di bawah ini:

```
<?php

echo anchor('buku/add_new', 'Tambah Buku');
echo "<br /><br />";

echo "<table border='1'>
    <tr>
        <th>No</th>
        <th>Judul</th>
        <th>Pengarang</th>
        <th>Kategori</th>
        <th>Aksi</th>
    </tr>";

$no = 0;
foreach($query->result_array() as $row)
{
    $no++;
    $kategori    = $row['Kategori'];

    $link_edit   = anchor('buku/edit/'. $row['ID_Buku'], 'Edit');
    $link_delete = anchor('buku/delete/'. $row['ID_Buku'], 'Hapus');

    echo "<tr>
        <td>".$no."</td>
        <td>".$row['Judul']."</td>
        <td>".$row['Pengarang']."</td>
        <td>$.opt_kategori[$kategori]."</td>
        <td>$.link_edit.' '. $link_delete."</td>
    </tr>";
}
echo "</table>";

?>
```

Simpan file tersebut di direktori kerja Anda, pada folder : **application/views/**

Beri nama file : **buku_v.php**

Penjelasan:

```
echo anchor('buku/add_new', 'Tambah Buku');
```

- Anchor() merupakan fungsi dari helper URL untuk men-generate *hiperlink*.
- Parameter pertama adalah alamat (url) yang akan diakses.
- Parameter kedua adalah teks dari *hiperlink* tersebut.
- Script tersebut sama dengan:

```
<a href='http://localhost/123040024/index.php/buku/add_new'>Tambah Buku</a>
```

```
foreach($query->result_array() as $row)
```

- Controller Buku mengirimkan hasil query melalui variabel `data['query']`. Oleh CI, index variabel data tersebut diubah menjadi **variabel** pada view. (Ingat kembali materi di Modul 1).
- Ulangi sebanyak **record hasil query**, resultset query di-*fetch* ke variabel `$row` dengan tipe result array. Lihat point A.3.b.
- Script tersebut sama dengan:

```
while ($row = mysql_fetch_array($query))  
{  
}
```

8. Uji Coba Insert Data

Lakukan penambahan beberapa data buku. Jika sudah sukses lanjutkan langkah 9.

9. Buat function Edit

Buka kembali file controller **Buku**, kemudian buat method baru bernama **edit**, yang akan berfungsi sebagai pengendali aksi edit data. Berikut script method **edit**:

```
function edit($id)  
{  
    $this->data['query'] = $this->Buku_m->get_records("ID_Buku = '$id'");  
    $this->data['is_update'] = 1;  
    $this->load->view('buku_form_v', $this->data);  
}
```

Penjelasan:

```
function edit($id)  
{
```

- Method yang akan berfungsi mengambil data buku berdasarkan parameter `$id` yang diterima dari view.
- Method ini dipanggil melalui *hiperlink* **Edit** pada view daftar buku.

```
$this->data['query'] = $this->Buku_m->get_records("ID_Buku = '$id'");
```

- Panggil method `get_records()` pada model `Buku_m`, untuk mengambil data buku di tabel, berdasarkan kriteria `ID_Buku = '$id'`.

```
$this->data['is_update'] = 1;
```

- Dimaksudkan untuk mengirim variabel `is_update` yang bernilai `1` ke view. Berarti aksi yang akan dilakukan adalah update data.

10.Modifikasi view Form Input Data

Untuk fitur edit data, tentunya harus ditampilkan dulu data sebelumnya yang akan diedit. Method `edit` pada controller `Buku` memberikan data yang akan diedit tersebut ke view, maka tugas view adalah menampilkannya pada kolom-kolom isian data. Buka kembali file view `buku_form_v.php`, kemudian modifikasi beberapa baris seperti berikut:

```
<?php

if (!empty($query))
{
    $row = $query->row_array();
}
else
{
    $row['ID_Buku'] = '';
    $row['Judul'] = '';
    $row['Pengarang'] = '';
    $row['Kategori'] = '';
}

echo form_open('buku/save/'. $is_update);

echo form_hidden('id', $row['ID_Buku']);

echo "Judul : ".form_input('judul', $row['Judul'], "size='50' maxlength='100'");
echo "<br /><br />";

echo "Pengarang : ".form_input('pengarang', $row['Pengarang'], "size='50' maxlength='150'");
echo "<br /><br />";

echo "Kategori : ".form_dropdown('kategori', $opt_kategori, $row['Kategori']);
echo "<br /><br />";

echo form_submit('btn_simpan', 'Simpan');

echo form_close();

?>
```

Penjelasan:

```
if (!empty($query))
{
    $row = $query->row_array();
}
```

- Jika variabel `$query` yang dikirimkan dari controller tidak kosong, maka ambil record pertama hasil eksekusi query di variabel `$query`.
- `row_array()` mengembalikan *result* satu record.

```
echo form_hidden('id', $row['ID_Buku']);
```

- Membuat elemen **hidden** bernama **id**, value diisi **ID_Buku** dari tabel **mst_buku**.
- Script tersebut sama dengan:

```
<input type="hidden" name="id" value="<?php echo $row['ID_Buku']?>" />
```

```
echo "Kategori : ".form_dropdown('kategori', $opt_kategori, $row['Kategori']);
```

- Membuat elemen combobox / dropdown list bernama **kategori**.
- Parameter pertama adalah nama elemen, kedua adalah opsi yang tersedia untuk combobox tersebut, ketiga adalah default value, keempat adalah atribut tambahan.
- Default value diambil dari data pada tabel **mst_buku**.
- Script tersebut kurang lebih sama dengan:

```
$selected_kosong = "";
if ($row['Kategori'] == '')
    $selected_kosong = "selected = 'selected'";

$selected_novel = "";
if ($row['Kategori'] == 'novel')
    $selected_novel = "selected = 'selected'";

$selected_komik = "";
if ($row['Kategori'] == 'komik')
    $selected_komik = "selected = 'selected'";

$selected_kamus = "";
if ($row['Kategori'] == 'kamus')
    $selected_kamus = "selected = 'selected'";

<select name="kategori">
    <option value="" $selected_kosong>- Pilih salah satu </option>
    <option value="novel" $selected_novel> Novel </option>
    <option value="komik" $selected_komik> Komik </option>
    <option value="kamus" $selected_kamus> Kamus </option>
</select>
```

11. Uji coba edit

Lakukan perubahan beberapa data buku. Jika sudah sukses lanjutkan langkah 12.

12. Buat function delete

Buka kembali file controller **Buku**, kemudian buat method baru bernama **delete**, yang akan berfungsi sebagai pendelasi aksi hapus data. Berikut script method **delete**:

```
function delete($id)
{
    if($this->Buku_m->delete_by_id($id))
    {
        redirect('buku');
    }
}
```

Penjelasan:

```
function delete($id)
{
```

- Method yang akan berfungsi menghapus data buku berdasarkan parameter **\$id** yang diterima dari view.
- Method ini dipanggil melalui *hiperlink* **Hapus** pada view daftar buku.

```
if($this->Buku_m->delete_by_id($id))
{
    redirect('buku');
}
```


- Hapus data buku berdasarkan **ID_Buku** yang dikirimkan dari view.
- Jika hapus sukses, maka *redirect* browser ke controller **Buku**.

13. Uji coba delete

Lakukan hapus beberapa data buku. Sebelum aksi hapus data benar-benar dilakukan program, **sangat disarankan** ada konfirmasi terlebih dahulu. Bagaimana caranya...?

Tambahkan javascript berikut pada view **buku_v.php** pada *hiperlink* **Hapus**:

```
$link_delete = anchor('buku/delete/'.$row['ID_Buku'], 'Hapus', "onclick='return confirm(\"Yakin?\")'");
```





CodeIgniter Framework
Modul 2
Selesai