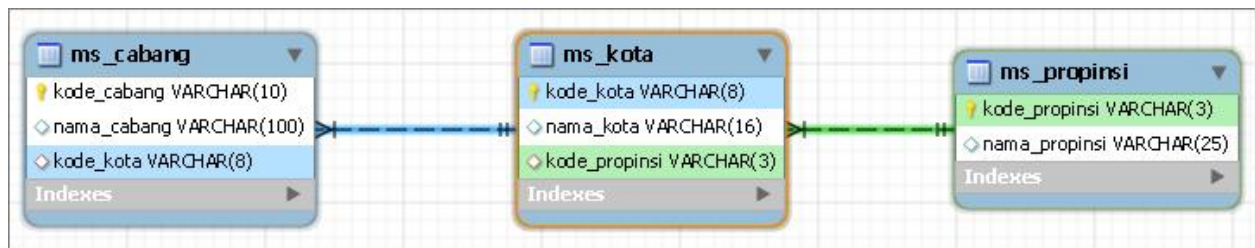


Tipe-tipe JOIN yang akan dibahas adalah :

- INNER JOIN
- CROSS JOIN
- OUTER JOIN
- STRAIGHT JOIN

Tiga table yang akan kita gunakan dari database tersebut adalah **ms_cabang**, **ms_kota** dan **ms_propinsi** dengan relasi terlihat seperti pada gambar berikut.



Keterangan : Relasi antar table ms_cabang, ms_kota dan ms_propinsi

INNER JOIN

INNER JOIN adalah tipe join yang akan kita bahas pertama. Tipe join ini akan mengambil semua row dari table asal dan table tujuan dengan kondisi nilai key yang terkait saja - jika ada, dan jika tidak maka row tersebut tidak akan muncul.

Kalau tidak terdapat kondisi key terkait antar table, maka semua row dari kedua table dikombinasikan.

Syntax dari INNER JOIN adalah sebagai berikut :

```
table_reference [INNER] JOIN table_factor [join_condition]
```

Terlihat bahwa keyword INNER boleh digunakan secara eksplisit atau tidak. Jika tidak digunakan maka konstruksi JOIN tanpa keyword lain dianggap sebagai INNER JOIN.

INNER JOIN Antar Table dengan Kondisi ("ms_cabang" dan "ms_kota" dengan key "kode_kota")

```
SELECT * FROM ms_cabang INNER JOIN ms_kota ON ms_cabang.kode_kota =
ms_kota.kode_kota
```

Hasil terlihat seperti gambar berikut ini. Disini table sumber adalah ms_cabang (left) mencari referensi row lain dari table ms_kota (right) dengan kondisi nilai **kode_kota** diantara kedua table tersebut sama. Kondisi ini menggunakan keyword **ON**.

Dengan panduan gambar, kita lihat bahwa tiap row dari ms_cabang akan dicari padanan row-nya di ms_kota :

- untuk row pertama kita memiliki kode_kota dengan nilai "KOTA-003", ini akan dicari referensinya ke table ms_kota untuk nilai yang sama dan kita dapatkan row dengan nilai nama_kota "Lhokseumawe" adalah padanannya.
- untuk row kedua kita memiliki kode_kota dengan nilai "KOTA-083", ini akan dicari referensinya ke table ms_kota untuk nilai yang sama dan kita dapatkan row dengan nilai nama_kota "Bau-bau" adalah padanannya.
- demikian seterusnya.

```

1 SELECT * FROM ms_cabang
2 INNER JOIN ms_kota
3 ON ms_cabang.kode_kota = ms_kota.kode_kota

```

kode_cabang	nama_cabang	kode_kota	kode_kota	nama_kota	kode_propinsi
CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-003	Lhokseumawe	P33
CABANG-002	PHI Mini Market - Bau-Bau 01	KOTA-083	KOTA-083	Bau-Bau	P25
CABANG-003	PHI Mini Market - Bogor 01	KOTA-039	KOTA-039	Bogor	P08
CABANG-004	PHI Mini Market - Medan 01	KOTA-007	KOTA-007	Medan	P30
CABANG-005	PHI Mini Market - Bogor 02	KOTA-039	KOTA-039	Bogor	P08
CABANG-006	PHI Mini Market - Sukabumi 01	KOTA-043	KOTA-043	Sukabumi	P08
CABANG-007	PHI Mini Market - Tegal 01	KOTA-055	KOTA-055	Tegal	P09
CABANG-008	PHI Mini Market - Jakarta Timur 01	KOTA-048	KOTA-048	Jakarta Timur	P32
CABANG-009	PHI Mini Market - Pasuruan 01	KOTA-062	KOTA-062	Pasuruan	P10
CABANG-010	PHI Mini Market - Langsa 01	KOTA-002	KOTA-002	Langsa	P33
CABANG-011	PHI Mini Market - Tarakan 01	KOTA-078	KOTA-078	Tarakan	P13
CABANG-012	PHI Mini Market - Tebing Tinggi 01	KOTA-012	KOTA-012	Tebing Tinggi	P30
CABANG-013	PHI Mini Market - Bekasi 01	KOTA-038	KOTA-038	Bekasi	P08
CABANG-014	PHI Mini Market - Batu 01	KOTA-056	KOTA-056	Batu	P10
CABANG-015	PHI Mini Market - Banjarmasin 01	KOTA-073	KOTA-073	Banjarmasin	P14

Sekarang mari kita coba hapus referensi untuk "Lhokseumawe" dari table ms_kota :

```
DELETE FROM ms_kota WHERE nama_kota = 'Lhokseumawe';
```

Setelah itu coba jalankan kembali perintah JOIN di atas, Anda akan mendapatkan hasil JOIN tanpa referensi row "Lhokseumawe". Jadi INNER JOIN dengan kondisi mengharuskan row dari tiap table memiliki nilai yang sama untuk column referensinya (dalam hal ini **kode_kota**).

```
1 SELECT * FROM ms_cabang JOIN ms_kota
2 ON ms_cabang.kode_kota = ms_kota.kode_kota
```

1 Result 2 Profiler 3 Messages 4 Table Data 5 Info 6 History

(Read Only) All rows Rows in a range First row: 0 1000 rows Refresh

	kode_cabang	nama_cabang	kode_kota	kode_kota	nama_kota	kode_propinsi
<input type="checkbox"/>	CABANG-002	PHI Mini Market - Bau-Bau 01	KOTA-083	KOTA-083	Bau-Bau	P25
<input type="checkbox"/>	CABANG-003	PHI Mini Market - Bogor 01	KOTA-039	KOTA-039	Bogor	P08
<input type="checkbox"/>	CABANG-004	PHI Mini Market - Medan 01	KOTA-007	KOTA-007	Medan	P30
<input type="checkbox"/>	CABANG-005	PHI Mini Market - Bogor 02	KOTA-039	KOTA-039	Bogor	P08
<input type="checkbox"/>	CABANG-006	PHI Mini Market - Sukabumi 01	KOTA-043	KOTA-043	Sukabumi	P08
<input type="checkbox"/>	CABANG-007	PHI Mini Market - Tegal 01	KOTA-055	KOTA-055	Tegal	P09
<input type="checkbox"/>	CABANG-008	PHI Mini Market - Jakarta Timur 01	KOTA-048	KOTA-048	Jakarta Timur	P32
<input type="checkbox"/>	CABANG-009	PHI Mini Market - Pasuruan 01	KOTA-062	KOTA-062	Pasuruan	P10
<input type="checkbox"/>	CABANG-010	PHI Mini Market - Langsa 01	KOTA-002	KOTA-002	Langsa	P33
<input type="checkbox"/>	CABANG-011	PHI Mini Market - Tarakan 01	KOTA-078	KOTA-078	Tarakan	P13
<input type="checkbox"/>	CABANG-012	PHI Mini Market - Tehing Tinggi 01	KOTA-012	KOTA-012	Tehing Tinggi	P30

Kembalikan lagi row referensi yang kita hapus tadi dengan perintah INSERT berikut ini :

```
INSERT INTO `ms_kota` VALUES ('KOTA-003','Lhokseumawe','P33')
```

INNER JOIN Antar Table Tanpa Kondisi ("ms_cabang" dengan "ms_kota")

```
SELECT * FROM ms_cabang INNER JOIN ms_kota
```

Hasilnya adalah untuk **tiap row** dari ms_cabang akan dikombinasikan dengan **semua row** dari ms_kota - contoh hasilnya terlihat seperti pada gambar di bawah ini.

1 `SELECT * FROM ms_cabang INNER JOIN ms_kota`

ms_cabang

ms_kota

1 Result 2 Profiler 3 Messages 4 Table Data 5 Info 6 History

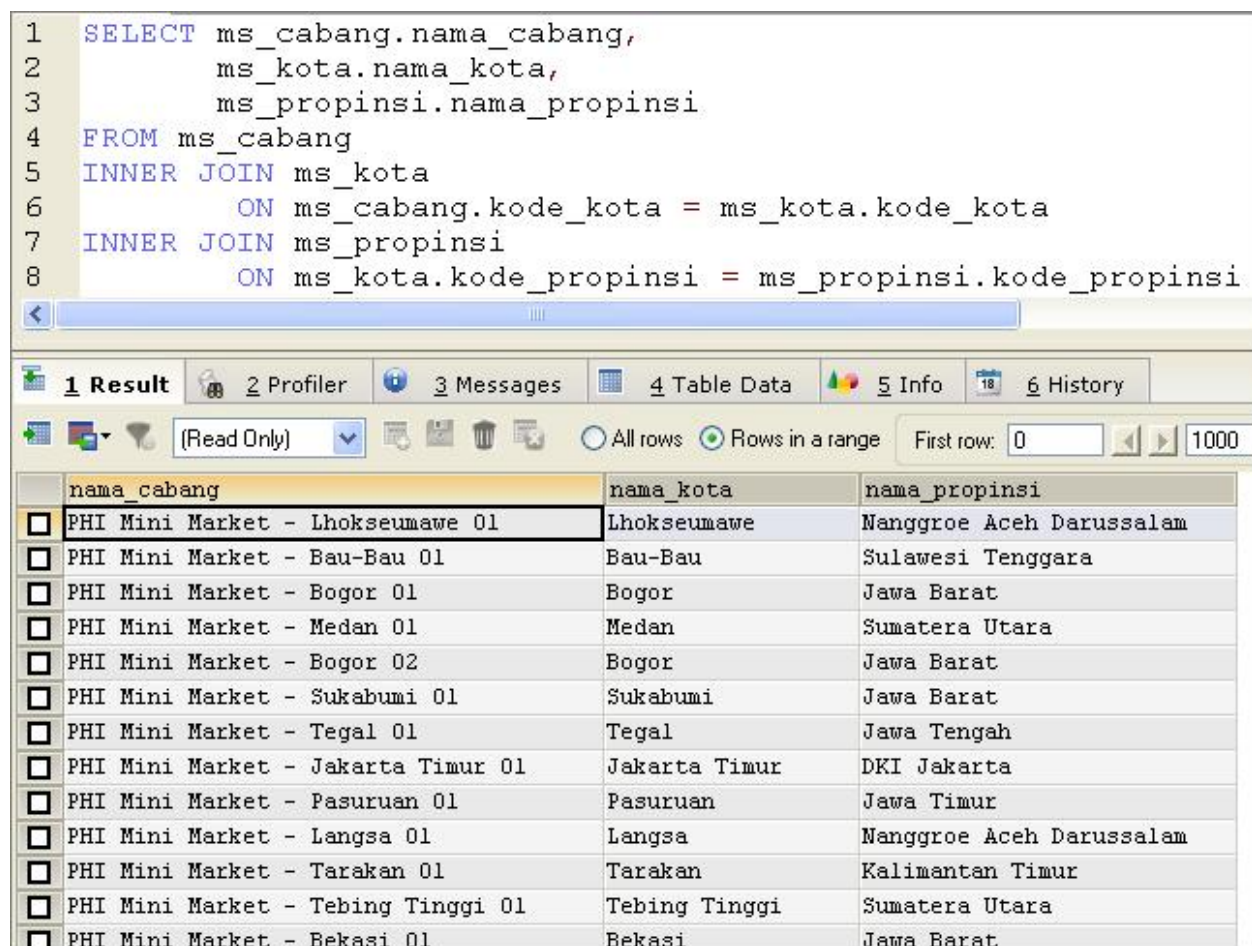
(Read Only) All rows Rows in a range First row: 0 1000 rows Refresh

kode_cabang	nama_cabang	kode_kota	kode_kota	nama_kota	kode_propinsi
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-001	Banda Aceh	P33
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-002	Langsa	P33
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-003	Lhokseumawe	P33
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-004	Sabang	P33
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-005	Subulussalam	P33
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-006	Binjai	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-007	Medan	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-008	Padang Sidempuan	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-009	Pematangsiantar	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-010	Sibolga	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-011	Tanjung Balai	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-012	Tebing Tinggi	P30
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-013	Bengkulu	P02
<input type="checkbox"/> CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	KOTA-014	Jambi	P07

INNER JOIN antar table "ms_cabang", "ms_kota" dan "ms_propinsi"

```
SELECT ms_cabang.nama_cabang,
       ms_kota.nama_kota,
       ms_propinsi.nama_propinsi
FROM ms_cabang
INNER JOIN ms_kota ON ms_cabang.kode_kota = ms_kota.kode_kota
INNER JOIN ms_propinsi ON ms_kota.kode_propinsi = ms_propinsi.kode_propinsi
```

Hasil eksekusi terlihat seperti gambar berikut. Dengan penggabungan ketiga table ini sekarang kita dapatkan setiap cabang memiliki informasi nama kota dan nama propinsi dimana cabang tersebut berada.



```
1 SELECT ms_cabang.nama_cabang,
2       ms_kota.nama_kota,
3       ms_propinsi.nama_propinsi
4 FROM ms_cabang
5 INNER JOIN ms_kota
6     ON ms_cabang.kode_kota = ms_kota.kode_kota
7 INNER JOIN ms_propinsi
8     ON ms_kota.kode_propinsi = ms_propinsi.kode_propinsi
```

nama_cabang	nama_kota	nama_propinsi
<input type="checkbox"/> PHI Mini Market - Lhokseumawe 01	Lhokseumawe	Nanggroe Aceh Darussalam
<input type="checkbox"/> PHI Mini Market - Bau-Bau 01	Bau-Bau	Sulawesi Tenggara
<input type="checkbox"/> PHI Mini Market - Bogor 01	Bogor	Jawa Barat
<input type="checkbox"/> PHI Mini Market - Medan 01	Medan	Sumatera Utara
<input type="checkbox"/> PHI Mini Market - Bogor 02	Bogor	Jawa Barat
<input type="checkbox"/> PHI Mini Market - Sukabumi 01	Sukabumi	Jawa Barat
<input type="checkbox"/> PHI Mini Market - Tegal 01	Tegal	Jawa Tengah
<input type="checkbox"/> PHI Mini Market - Jakarta Timur 01	Jakarta Timur	DKI Jakarta
<input type="checkbox"/> PHI Mini Market - Pasuruan 01	Pasuruan	Jawa Timur
<input type="checkbox"/> PHI Mini Market - Langsa 01	Langsa	Nanggroe Aceh Darussalam
<input type="checkbox"/> PHI Mini Market - Tarakan 01	Tarakan	Kalimantan Timur
<input type="checkbox"/> PHI Mini Market - Tebing Tinggi 01	Tebing Tinggi	Sumatera Utara
<input type="checkbox"/> PHI Mini Market - Bekasi 01	Bekasi	Jawa Barat

Implisit INNER JOIN dengan Koma

INNER JOIN antar table secara implisit dapat menggunakan daftar table yang dipisah dengan tanda koma (,). Pengkondisian menggunakan klausa where.

```
SELECT ms_cabang.nama_cabang,
       ms_kota.nama_kota      ,
       ms_propinsi.nama_propinsi
FROM   ms_cabang
       , ms_kota
       , ms_propinsi
WHERE  ms_cabang.kode_kota = ms_kota.kode_kota
AND    ms_kota.kode_propinsi = ms_propinsi.kode_propinsi
```

CROSS JOIN

CROSS JOIN identik dengan INNER JOIN pada MySQL 5.0. Pembahasannya sama dengan INNER JOIN sehingga tidak diulangi lagi disini.

Contoh Penggunaan :

```
SELECT ms_cabang.nama_cabang,
       ms_kota.nama_kota,
       ms_propinsi.nama_propinsi
FROM   ms_cabang
CROSS JOIN
       ms_kota ON ms_cabang.kode_kota = ms_kota.kode_kota
CROSS JOIN
       ms_propinsi ON ms_kota.kode_propinsi = ms_propinsi.kode_propinsi
```

OUTER JOIN

OUTER JOIN merupakan tipe join yang mencari referensi data dari suatu table sumber ke table lain dengan tidak menghilangkan data sumber apabila referensi tidak ditemukan.

Untuk menggunakan tipe OUTER JOIN maka perlu memperhatikan beberapa hal berikut :

- perlu dibedakan antara table sumber dan table referensi, ini ditentukan dengan cara menspesifikasikan kedudukan table sumber apakah di kiri (**LEFT**) atau di kanan (**RIGHT**).
- jika tidak ada data dari table referensi yang cocok dengan kondisi join maka hanya data dari table sumber yang ditampilkan tetapi kolom-kolom table referensi akan berisi null.

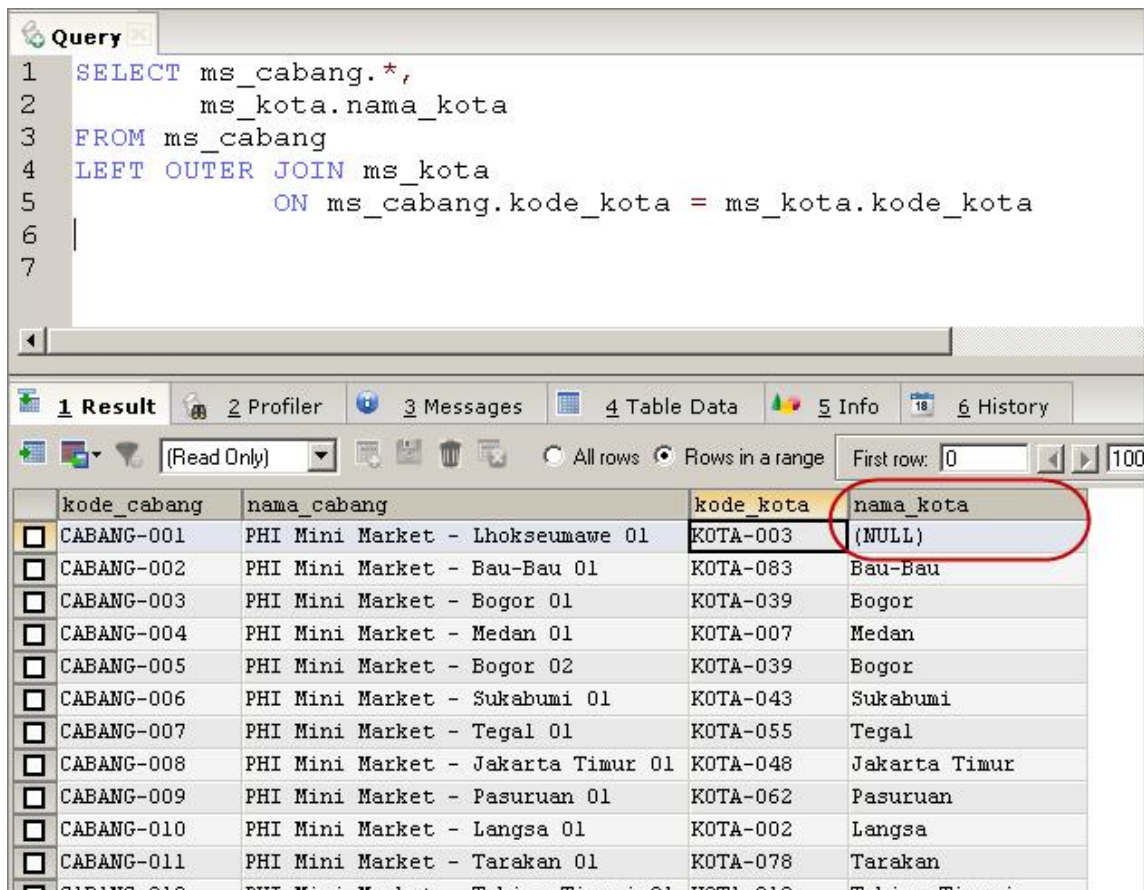
Contoh Penggunaan :

- Hapus data master "Lhokseumawe" dari table **ms_kota**

```
DELETE FROM ms_kota WHERE nama_kota = 'Lhokseumawe'
```

- Lakukan join seperti perintah berikut ini, dan perhatikan hasilnya seperti pada gambar.

```
SELECT ms_cabang.*,
       ms_kota.nama_kota
FROM ms_cabang
LEFT OUTER JOIN ms_kota
ON ms_cabang.kode_kota = ms_kota.kode_kota
```



kode_cabang	nama_cabang	kode_kota	nama_kota
CABANG-001	PHI Mini Market - Lhokseumawe 01	KOTA-003	(NULL)
CABANG-002	PHI Mini Market - Bau-Bau 01	KOTA-083	Bau-Bau
CABANG-003	PHI Mini Market - Bogor 01	KOTA-039	Bogor
CABANG-004	PHI Mini Market - Medan 01	KOTA-007	Medan
CABANG-005	PHI Mini Market - Bogor 02	KOTA-039	Bogor
CABANG-006	PHI Mini Market - Sukabumi 01	KOTA-043	Sukabumi
CABANG-007	PHI Mini Market - Tegal 01	KOTA-055	Tegal
CABANG-008	PHI Mini Market - Jakarta Timur 01	KOTA-048	Jakarta Timur
CABANG-009	PHI Mini Market - Pasuruan 01	KOTA-062	Pasuruan
CABANG-010	PHI Mini Market - Langsa 01	KOTA-002	Langsa
CABANG-011	PHI Mini Market - Tarakan 01	KOTA-078	Tarakan

- Tambahkan kembali data "Lhokseumawe" ke table **ms_kota**

```
INSERT INTO ms_kota(kode_kota, nama_kota, kode_propinsi)
VALUES ('KOTA-003', 'Lhokseumawe', 'P33');
```

- Sekarang coba ganti syntax pada query di atas dari "LEFT" menjadi "RIGHT" dan lihat hasil eksekusinya. Tentunya dari hasil tersebut Anda sudah dapat mengambil kesimpulan perbedaan dari kedua konstruksi tersebut.

STRAIGHT_JOIN

STRAIGHT_JOIN merupakan pengganti keyword **JOIN** pada MySQL yang digunakan untuk "memaksa" proses join table dari kiri (LEFT) ke kanan (RIGHT).

Contoh Penggunaan :

```
SELECT ms_cabang.* ,  
       ms_kota.nama_kota  
FROM ms_cabang  
STRAIGHT_JOIN ms_kota  
ON ms_cabang.kode_kota = ms_kota.kode_kota
```