**Praktikum #9**

**Pengolahan Multi Tabel**

|  |
| --- |
| **Standar Kompetensi**  Mahasiswa dapat melakukan pengolahan data dengan menggunakan multi tabel |

|  |  |
| --- | --- |
| **Kompetensi Dasar** | **Indikator** |
| 1. Mampu menghubungkan dua tabel untuk menampilkan informasi | 1. Mampu menggunakan INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN untuk menghubungkan dua tabel. 2. Mampu membedakan cara penggunaan INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN |
| 1. Mampu menghubungkan tiga tabel untuk menampilkan informasi | Mampu memperluas penggunaan INNER JOIN (yang secara optional digunakan oleh MySQL untuk JOIN), untuk pengolahan dengan menggunakan tiga tabel masukan. |

1. Pengolahan dengan dua tabel atau lebih

Untuk memenuhi kebutuhan informasi yang lengkap, seringkali data tidak dapat diperoleh hanya dari satu tabel. Data yang diperoleh dari isian berbagai field harus diambil dari dua tabel atau lebih.

Misal dengan menggunakan tabel **ruang**, **barang**, dan **isiruang** yang sudah dihasilkan dari praktikum sebelumnya, diinginkan informasi tentang barang-barang apa saja dan jumlahnya berapa yang ada pada suatu ruang, maka keinginan tersebut akan dapat dipenuhi dengan menghubungkan tabel **isiruang** dengan tabel **barang**. Pertanyaannya adalah kenapa?, karena data tentang nomor ruang, nomor barang dan jumlah barang ada di tabel **isiruang**, sedangkan data tentang nama barang ada di tabel **barang**.

MySQL dilengkapi dengan fasilitas untuk menghubungkan dua tabel atau lebih melalui suatu proses yang disebut JOIN. Untuk dapat dihubungkan, dua tabel harus mempunyai kolom atau field bersama, yaitu kolom yang mempunyai domain yang sama. Isi kolom bersama dari kedua tabel isinya akan dibandingkan atau dicocokkan untuk menentukan baris atau baris-baris dari dua tabel yang mempunyai hubungan.

Untuk proses JOINT yang diperlukan adalah:

* Atribut-atribut yang akan ditampilkan. Untuk memperjelas atribut mana yang dimaksud, maka penulisan nama atribut diawali dengan nama tabel dengan penghubung berupa tanda titik.

Contoh: isiruang.kodebarang, untuk atribut kodebarang dari tabel **isiruang**

* Tabel-tabel yang digunakan sebagai sumber data
* Kondisi yang berisi kondisi atau kondisi-kondisi yang harus dipenuhi

Dengan proses JOINT tabel-tabel yang digunakan sebagai sumber data tidak akan mengalami pengubahan isi, jadi sebatas hanya digunakan sebagai input.

Macam hasil penggabungan untuk dua tabel ialah:

1. Hanya menampilkan data yang ditemukan pasangannya di semua tabel sumber.
2. Menampilkan data dengan acuan tabel sumber pertama, jika di tabel sumber kedua data tidak ditemukan, diisi dengan NULL.
3. Menampilkan data dengan acuan tabel sumber kedua, jika di tabel sumber pertama data tidak ditemukan, diisi dengan NULL.

Secara optional untuk proses JONT digunakan macam yang pertama. Untuk data masukan yang terdiri lebih dari dua tabel, digunakan pengembangan dari proses dengan dua tabel.

1. Pengolahan dengan menggunakan dua tabel.

Untuk mempermudah pemberian contoh, ambil kembali tabel-tabel **ruang**, **barang**, dan **isiruang**.

1. Perintah INNER JOIN

Bentuk umum:

**SELECT field1, field2, …..**

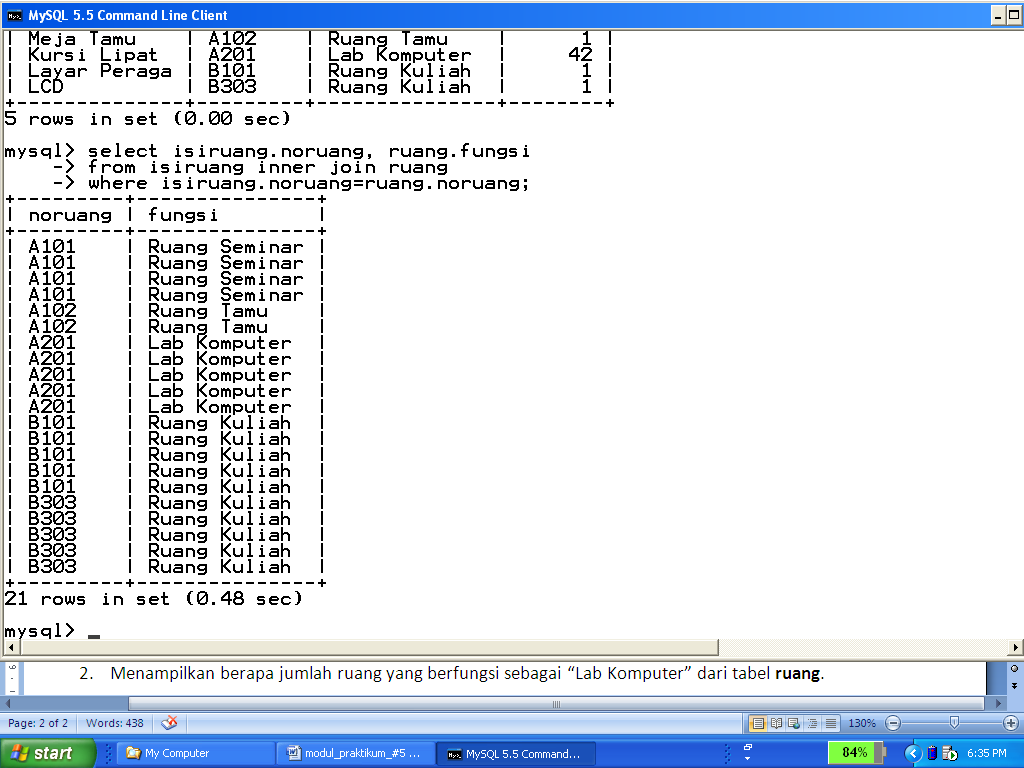
**FROM tabel1 INNER JOIN tabel2**

**ON kondisi**

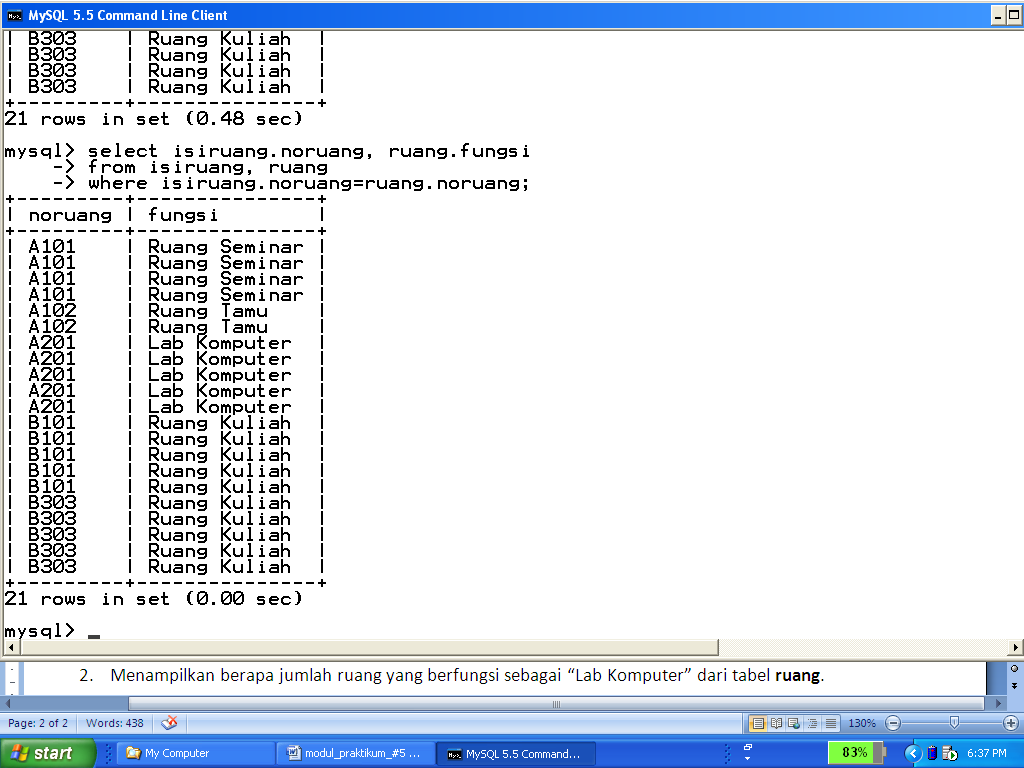
Penulisan nama field yang lengkap ialah dengan menambahkan nama tabel yang terkait, dengan diberi tanda penghubung berupa tanda titik (.). Contoh: isiruang.noruang, yang artinya field noruang dari tabel **isiruang**.

Dengan INNER JOIN akan dihasilkan baris-baris yang mempunyai hubungan diantara tabel1 dan tabel2, untuk yang tidak mempunyai hubungan tidak akan ditampilkan.

Jika diinginkan ditampilkan nomor ruang (dari tabel **isiruang**), dan fungsi ruang (dari tabel **ruang**), maka perintah dan hasilnya adalah sebagai berikut:



Atau karena INNER JOIN adalah optional, maka untuk keperluan di atas dapat juga digunakan perintah dengan tidak menyertakan INNER JOIN sebagai berikut:



Hasilnya ternyata sama.

1. Perintah STRAIGHT JOIN

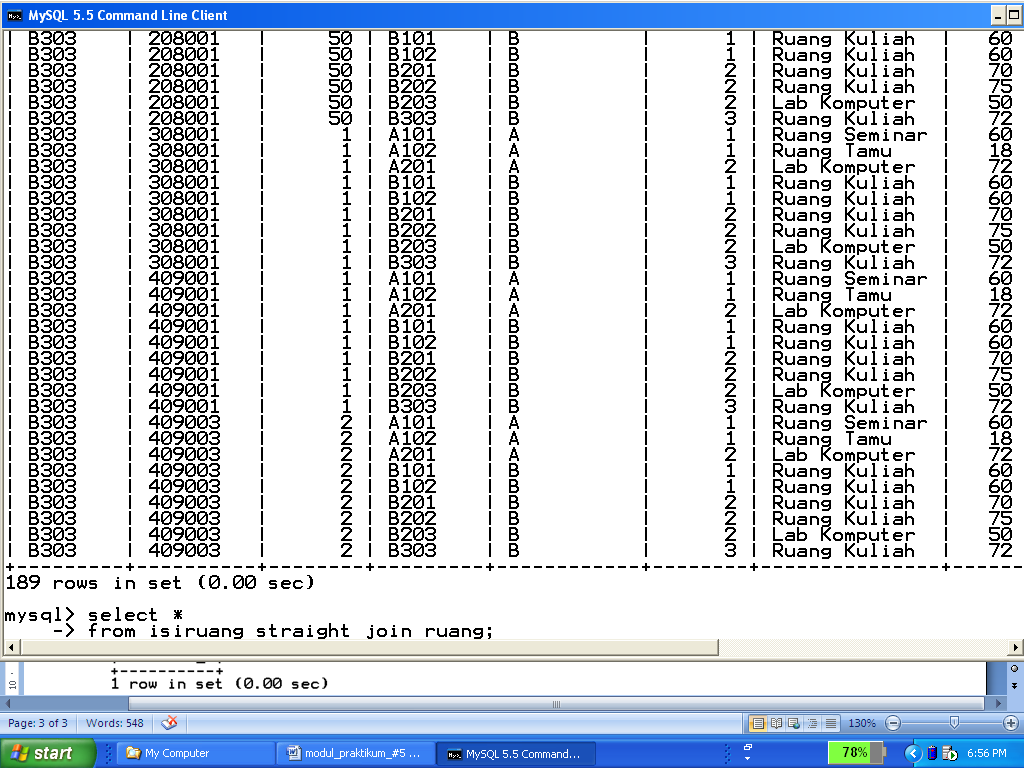
Bentuk umum:

**SELECT field1, field2, …..**

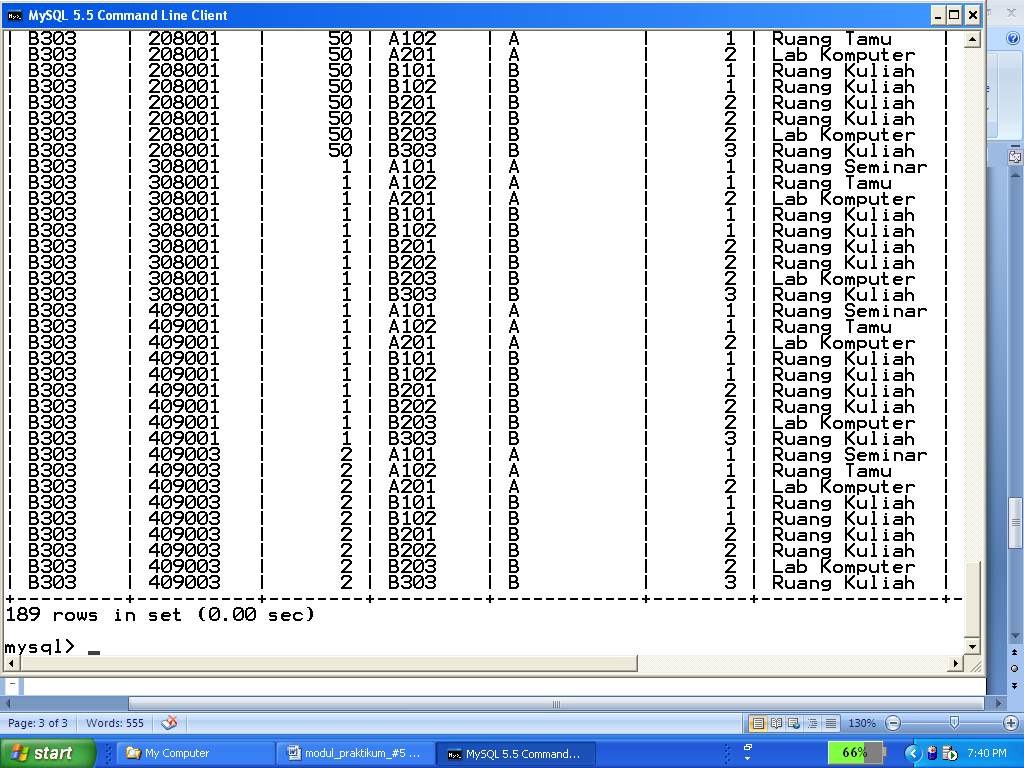
**FROM tabel1 STRAIGHT JOIN tabel2**

Perhatikan untuk STRAIGHT JOIN tidak mengenal adanya klause where. Hasil operasi dari STRAIGHT JOIN ialah munculnya pasangan data yang ada pada kedua tabel. Sehingga jumlah record yang dihasilkan merupakan hasil perkalian dari jumlah record data tabel masukan pertama dengan jumlah record data tabel masukan kedua.

Misal diberikan perintah sebagai berikut.



Hasilnya adalah setiap data dari tabel **isiruang** dipasangkan dengan setiap data dari tabel **ruang**. Tampilan lengkap cukup panjang (189 baris), dan bagian akhir dari tampilan baris-baris yang dihasilkan adalah sebagai berikut



Perhatikan kembali jumlah baris yang ditampilkan, yaitu 189.

Dari mana angka tersebut dihasilkan?

1. Perintah RIGHT JOIN

Bentuk umum:

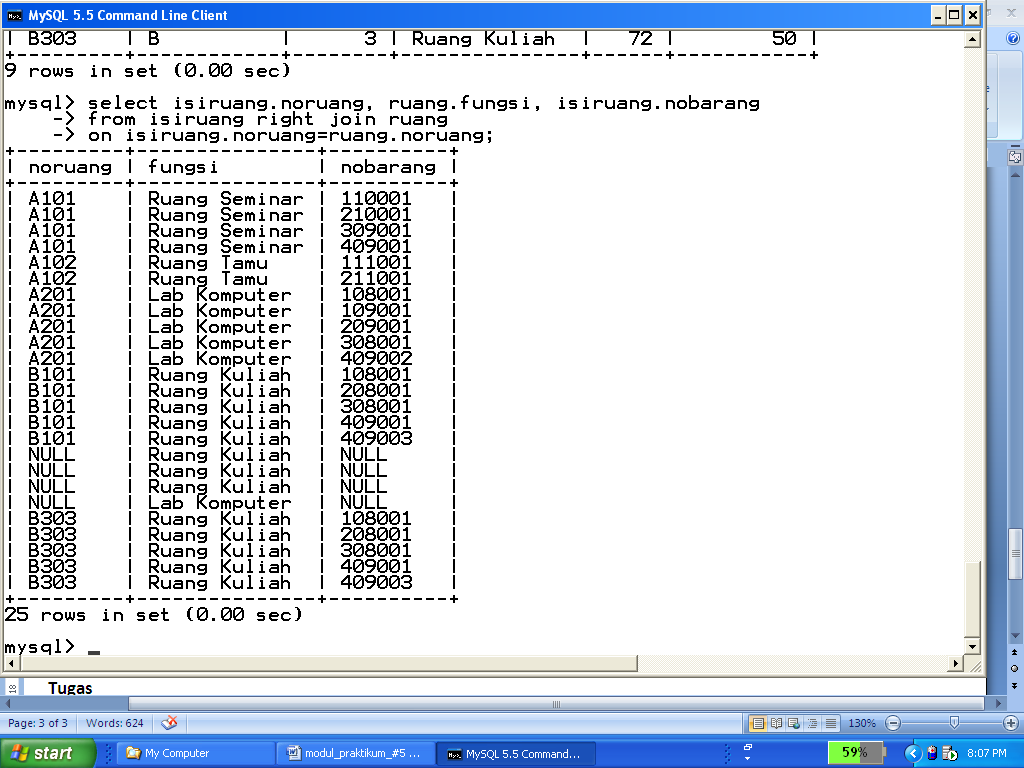
**SELECT field1, field2, …..**

**FROM tabel1 RIGHT JOIN tabel2**

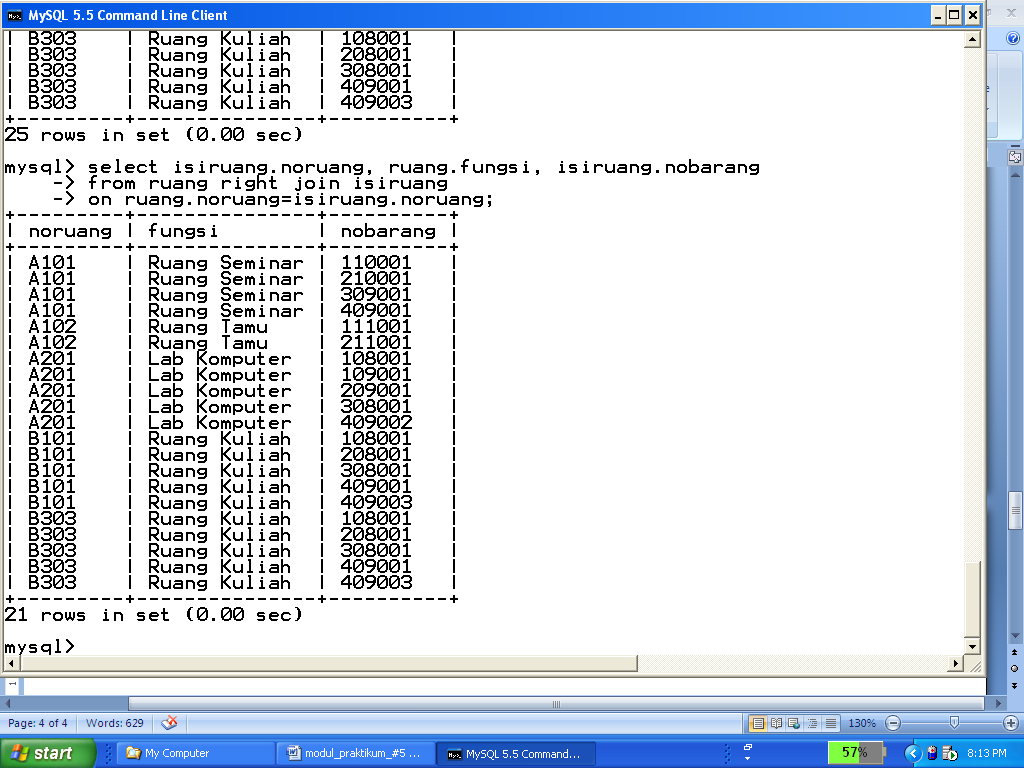
**ON kondisi**

Dengan RIGHT JOIN maka yang dipakai sebagai dasar di dalam menampilkan hasil proses JOIN adalah tabel yang ditulis di sebelah kanan perintah JOIN, yaitu tabel2. Dengan demikian semua record dari tabel2 akan ditampilkan, dan untuk yang tidak mempunyai hubungan di tabel1 isi field yang berasal dari tabel1 diberi NULL.

Misal diminta menampilkan nomor ruang (dari tabel **isiruang**), fungsi ruang (dari tabel **ruang**), dan nomor barang (dari tabel **isiruang**), maka perintah dan hasil tampilannya adalah sebagai berikut.



Bagaimana dengan perintah dan hasil yang berikut ini?



Tampilan yang terakhir diperoleh dengan melakukan saling tukar tabel yang digunakan sebagai tabel1 dan tabel2 pada pengolahan sebelumnya. Amati hasilnya, dan bandingkan dengan hasil pengolahan sebelumnya. Seperti apa kesimpulan hasil pembandingan Saudara?

1. Perintah LEFT JOIN

Bentuk umum:

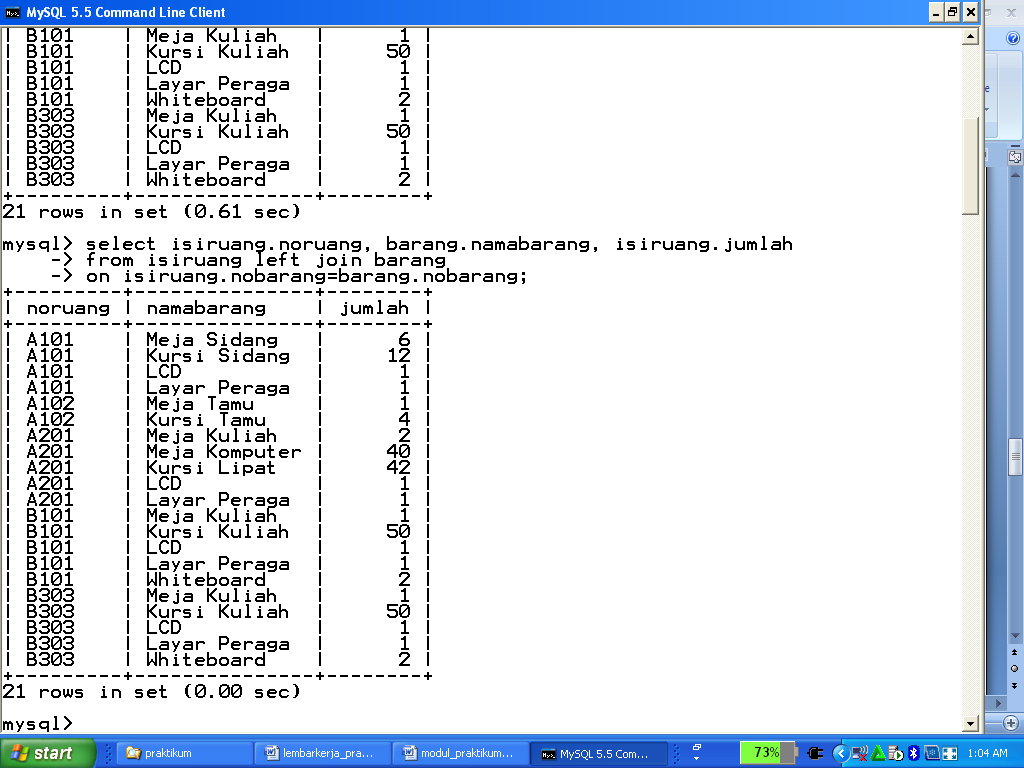
**SELECT field1, field2, …..**

**FROM tabel1 LEFT JOIN tabel2**

**ON kondisi**

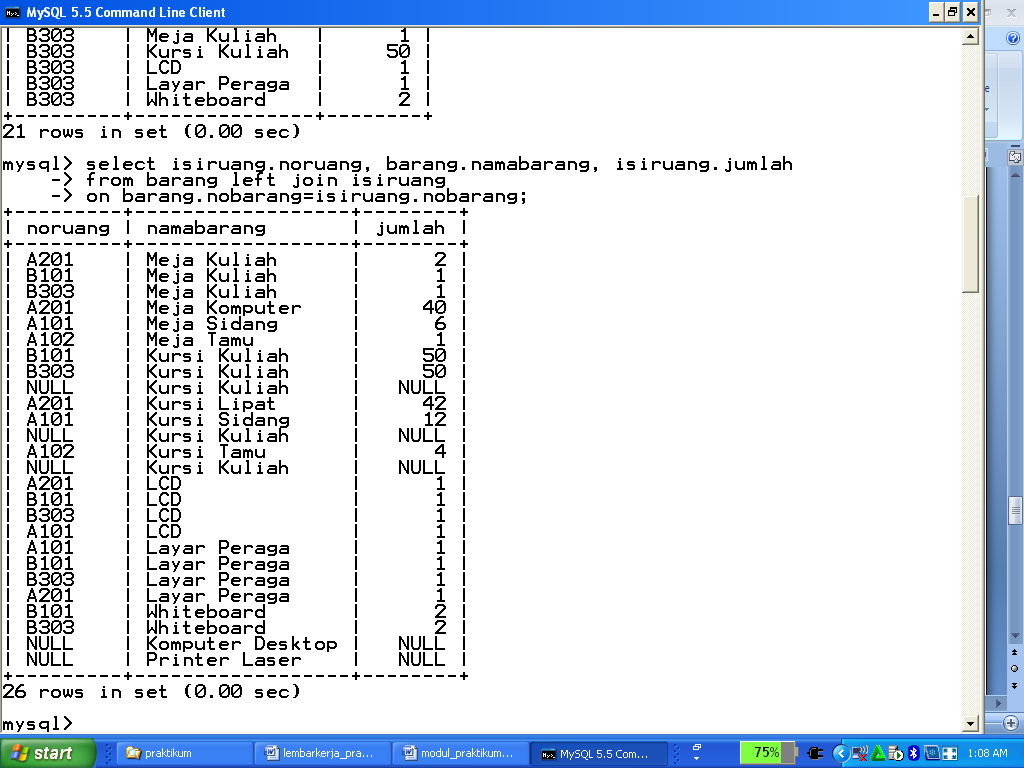
Dengan LEFT JOIN aturan yang dipakai ialah kebalikan dari RIGHT JOIN.

Misal diminta menampilkan nomor ruang (dari tabel **isiruang**), nama barang (dari tabel **barang**), dan jumlah barang (dari tabel **isiruang**), maka perintah dan hasilnya adalah sebagai berikut:



Bandingkan hasilnya dengan jika penempatan tabel yang digunakan sebagai sumber dibalik, yaitu tabel **barang** sebagai tabel1, dan tabel **isibarang** sebagai tabel2.

Perintah dan hasilnya ialah sebagai berikut.



Apa yang berbeda dengan hasil sebelumnya?

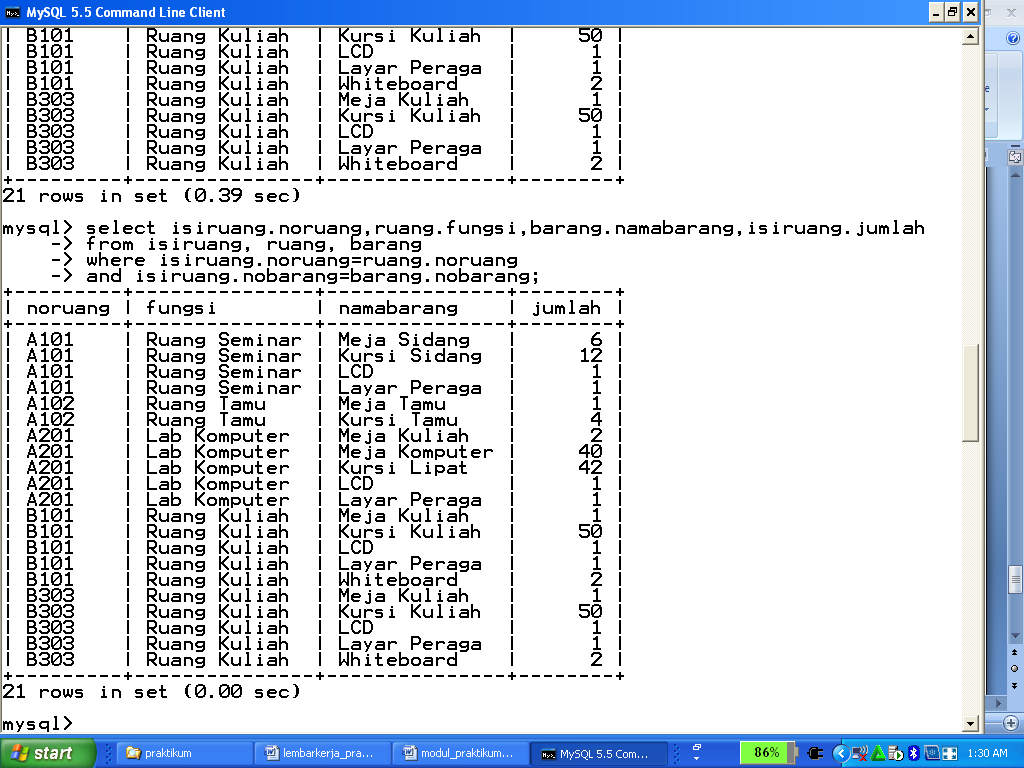
1. Pengolahan dengan menggunakan tiga tabel.

Sebagai perluasan dari proses JOIN dengan menggunakan dua tabel, ialah proses JOIN dengan menggunakan tiga tabel.

Ambil kembali tabel-tabel **ruang**, **barang**, dan **isiruang**. Misal diminta ditampilkan sekaligus field-field sebagai berikut:

* nomor ruang dan jumlah barang : dari tabel **isiruang**
* fungsi ruang : dari tabel **ruang**, dan
* nama barang : dari tabel **barang**.

Perintahyang diberikan dan tampilan yang dihasilkan adalah sebagai berikut.

****

Dengan perintah di atas akan dhasilkan tampilan sebagaimana jika digunakan INNER JOIN, karena secara optional yang digunakan untuk proses JOIN adalah INNER JOIN.

**Tugas A**

Dengan menggunakan tabel-tabel **ruang**, **barang**, dan **isiruang** yang telah dihasilkan pada praktikum-praktikum sebelumnya.

Susun perintah untuk pengolahan-pengolahan sebagai berikut:

1. Menampilkan nomor ruang, fungsi ruang, dan jumlah barang untuk ruang-ruang yang terdapat LCD di dalamnya.
2. Menampilkan nomor ruang, fungsi ruang, jumlah barang untuk ruang kuliah yang mempunyai kursi dengan jumlah minimum 50.
3. Menampilkan nomor barang, nama barang, nomor ruang, dan fungsi ruang untuk ruang kuliah yang ada di lantai 2.
4. Menampilkan nomor barang, nama barang, jumlah barang, nomor ruang, dan fungsi ruang untuk barang-barang yang ada di Lab Komputer.
5. Menampilkan nomor barang, nama barang, jumlah barang, nomor ruang, dan fungsi ruang untuk barang-barang yang berupa alat peraga/whiteboard.

**Tugas B :** Menggunakan DB Organisasi, buatlah masing-masing 1 contoh join (INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN) dari dua atau tiga tabel kemudian jelaskan query yang anda lakukan !

**-oOo-**