# Pertemuan I

# PEMODELAN BASIS DATA

1. **Tujuan**
   1. Memahami konsep pemodelan Data base, yang meliputi Entity,Atribut,Relationship, Key,dan Kardinalitas.
   2. Mampu membuat pemodelan dengan Entity Relationship Diagram(ERD).
2. **Alokasi Waktu : 2 x 60 menit**
3. **Dasar Teori**

Entity Relationship Diagram(ERD) adalah ilustrasi dari entitas-entitas dalam bisnis dan relationship antar entitas. ERD memisahkan antara informasi yang dibutuhkan dalam bisnis dari aktivitas – aktivitas yang dilakukan dalam bisnis.

Tujuan utama dari permodelan ERD adalah untuk menunjukkan struktur objek data(entity) dan hubungan (relationship) yang ada pada objek tersebut. ERD ini berguna bagi professional sistem,karena ERD memperlihatkan hubungan antara data store pada Data Flow Diagram(DFD). Terdapat enam macam konstruksi utama dari ER,yaitu entity,atribut,relationship,key,dan Kardinalitas.

Perlu diingat bahwa tata cara penulisan model ERD yang dibahas dalam modul ini bukanlah satu-satunya pilihan. Notasi ERD yang dipakai di modul ini diambil dari buku “Database System Concept 6th edition” karangan Silberschatz.

**1.1 Entity**

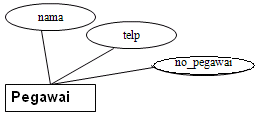
Entity atau entitas adalah konsep dasar dalam permodelan basis data berupa individu yang mewakili sesuatu yang nyata dan dapat di bedakan dari sesuatu yang lain. Entity dinotasikan dengan empat persegi panjang serta pemberian nama,biasanya dengan menggunakan kata benda,seperti terlihat pada gambar.

PRODUK

**1.2 Atribut**

Atribut adalah karakteristik suatu entitas yang mendeskripsikan suatu entitas. Atribut dapat juga disebut sebagai karakteristik atau property dari entitas tersebut. Contoh: suatu produk memilii atribut berupa identitas seperti id\_produk,nama\_produk,dan karakteristik lain yang mewakili identitas supplier. Setiap atribut bisa bersifat wajib(harus diisi,not null) dan bisa pula optional. Atribut dinotasikan dengan bentuk oval,seperti:

Contoh bentuk entitas dengan atribut:



Atribut memiliki jenis yang bermacam – macam, tergantung dari bentuk,asal,dan isi atribut itu sendiri.

**1.2.1 Atribut Sederhana dan Komposit**

Pada contoh di atas yang dikategorikan atribut sederhana adalah no\_pegawai,nama,dan telp. Sedangkan atribut komposit terdiri dari beberapa atribut sederhana (bisa dipahami secara hirarkis). Misal,atribut telp,yang bisa dibagi menjadi jalan,kota,dan kode\_pos.

**1.2.2 Atribut Bernilai Tunggal (single-valued) dan Bernilai Ganda(multivalued)**

Bernilai tunggal artinya hanya memiliki satu nilai data. Misalnya, atribut nama untuk satu nama pegawai saja(terdiri dari satu nama saja untuk satu entitas). Sedangkan atribut ganda bernilai ganda memiliki lebih dari satu nilai data.

**1.2.3 Atribut bernilai Null**

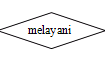
Jika sebuah baris tidak memiliki nilai (data) pada kolom tertentu, maka nilai kolom tersebut dinamakan null. Null berarti nilai yang sangat unix dalam basis data. Nilai null bisa berarti nilai tidak tersedia(unavailable),tidak ditulis(unassigned),tidak diketahui(unknown),atau tidak mungkin dimasukkan(inapplicable). Akan tetapi,null TIDAK sama dengan nol(0) atau spasi kosong. Nol(0) adalah angka,sedangkan spasi termasuk karakter. Meskipun begitu,terkadang user tidak menginginkan adanya nilai null. Untuk itu,doperlukan batasan(constraint)berupa NOT NULL atau PRIMARY KEY untuk mencegah munculnya nilai null.

**1.2.4 Atribut Turunan(derivied attribute)**

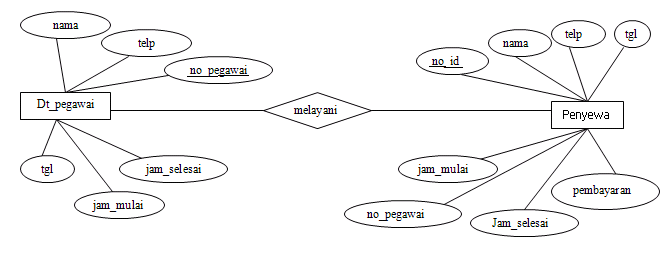
Nilai atribut ini diturunkan dari atribut yang lain. Biasanya berlaku untuk perhitungan selisih yang dibutuhkan oleh satu atribut, dan perhitungannya tersebut tergantung dari atribut lain yang berkaitan dengan atribut yang bersangkutan.

**1.3 Relationship**

Relasi menunjukkan adanya hubungan di antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Relationship tidak mempunyai keberadaaan fisik kecuali yang mewarisi dari hubungan entitas tersebut. Dalam relationship,terdapat istilah Relationship set,yaitu kumpulan relationship sejenis. Notasi yang digunakan adalah bentuk wajik dan pemberian nama biasanya menggunakan kata kerja,Contoh :



Contoh dari penggabungan entitas,atribut dengan relationship:

****

**1.4 Key**

Setiap entitiy biasanya memiliki sebuah atribut yang nilainya berbeda untuk masing – masing individu dalam entity. Oleh karena itu, dua nilai yang sama untuk atribut kunci tersebut tidak dibolehkan. Key digambarkan dengan atribut bergaris bawah pada ERD,seperti ditunjukkan pada contoh di atas. Key ini ada dua macam yaitu primary key dan foreign key.

Primary Key atau Unique Key merupakan kunci yang secara unik mengidentifikasi baris pada tabel. Primary Key atau Unique Key dapat diterapkan pada satu atau lebih field. Apabila primary key / unique key diterapkan pada satu atau lebih field maka pada field tersebut tidak boleh bernilai sama untuk seluruh baris pada tabel. Kemudian yang membedakan primary key dan unique key adalah setiap kita memberikan constraint primary key pada field maka constraint NOT NULL akan diberikan pada field tersebut secara otomatis, sedangkan pada unique key tidak.

Dengan adanya foreign key dapat terlihat adanya hubungan antara satu tabel dengan tabel lain. Foreign key mengidentifikasi satu atau lebih kolom pada satu tabel (tabel referensi) mengarah dari satu atau lebih tabel lainnya (tabel induk). Dengan adanya constraint foreign key pada satu kolom maka data pada kolom tabel referensi harus terdapat pada tabel induk. Pada mysql 5.0, foreign key hanya dapat diterapkan pada tabel dengan type InnoDB namun pada versi selanjutnya dapat diterapkan juga pada MyIsam. Kemudian field dengan tipe BLOB atau TEXT tidak dapat diberikan contraint foreign key.

Untuk mengoptimasi database kita dapat menggunakan index, karena index mampu meningkatkan kecepatan operasi pada suatu tabel (terutama operasi SELECT). Pada mysql, index dan data dari suatu tabel tersimpan secara terpisah, dengan kata lain membutuhkan ruang penyimpanan yang lebih banyak, kesalahan pemberian index pada suatu kolom dapat memperlambat operasi pada suatu tabel, karena index dibaca terlebih dahulu untuk mendapatkan data. Gunakan foreign key pada kolom yang sering digunakan untuk pencarian data.

**1.5 Kardinalitas**

Pemetaan kardinalitas (mapping cardinalities) atau rasio kardinalitas menspesifikasikan jumlah kejadian relationship dimana sebuah entitiy dapat berpartisipasi. Rasio kardinalitas sangat berguna untuk mendiskripsikan relationship yang melibatkan dua entity. Ada empat macam rasio kardinalitas,yaitu:

1. Hubungan satu ke satu (*one to one*)

Adalah **satu** *entity* dalam A dihubungkan dengan maksimum **satu** *entity* dalam B. Contoh:



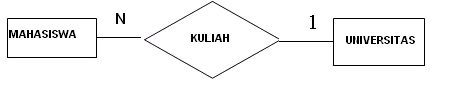
2. Hubungan satu ke banyak (*one to many*)

Adalah **satu** *entity* dalam A dihubungkan dengan **sejumlah** *entity* dalam B. Satu *entity* dalam B dihubungkan dengan maksimum satu *entity* dalam A. Contoh:



1. Hubungan banyak ke satu (*many to one*)

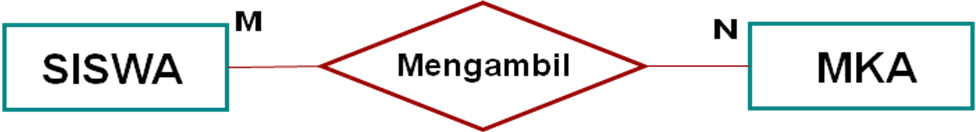
**Satu** *entity* dalam B dapat dihubungkan dengan **sejumlah** *entity* dalam A. Contoh:



4. Hubungan banyak ke banyak (*many to many*)

Adalah satu *entity* dalam A dihubungkan dengan sejumlah *entity* dalam B.  Satu *entity* dalam B dapat dihubungkan dengan sejumlah *entity* dalam A.

Contoh:



Macam-macam *entity*:

1. *Strong entity* atau *entity* kuat, adalah *entity* yang mempunyai *primary key*.
2. *Weak entity* atau *entity* lemah, adalah e*ntity* yang tidak mempunyai *primary key*.

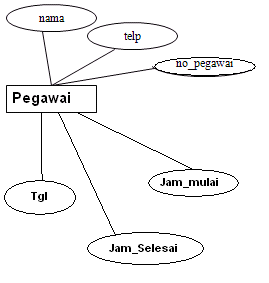
**1.6 Konversi ERD**

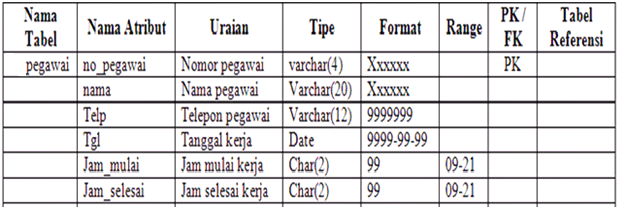
Tabel adalah representasi data dua dimensi: kolom dan baris. Dalam database, baris menunjukkan isi data sesungguhnya. Oleh karena model data hanya menggambarkan struktur data (bukan isinya), maka pembahasan tabel untuk saat ini dibatasi pada penamaan tabel dan kolom. Konversi ERD dapat dipetakan dengan mudah melalui cara berikut ini.

• Entity 🡺 tabel

• Atribut 🡺 kolom

Dari pemetaan di atas, dapat diambil suatu pengertian yaitu, Entity dalam ERD dijadikan / dikonversikan sebagai Tabel dalam Database, dan Atribut dalam ERD dijadikan / dikonversikan sebagai Kolom dalam Database. Contoh :





**Tabel Pegawai:**

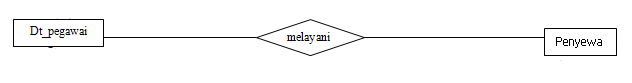
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **no\_pegawai** | **nama** | **telp** | **tgl** | **jam\_mulai** | **jam\_selesai** |
| **DS01** | **Dasha** | **883334** | **2009-02-18** | **9** | **15** |
| **DS02** | **Yudhitya** | **883335** | **2009-02-18** | **15** | **21** |
| **DS03** | **Fajarina** | **883336** | **2009-02-19** | **9** | **15** |
| **DS04** | **Dhita** | **883337** | **2009-02-19** | **15** | **21** |
| **DS05** | **Dina** | **883340** | **2009-02-20** | **9** | **15** |
| **DS06** | **Vyean** | **883349** | **2009-02-20** | **15** | **21** |

**1.7 MEMBUAT ERD**

Tahapan Pembuatan ERD

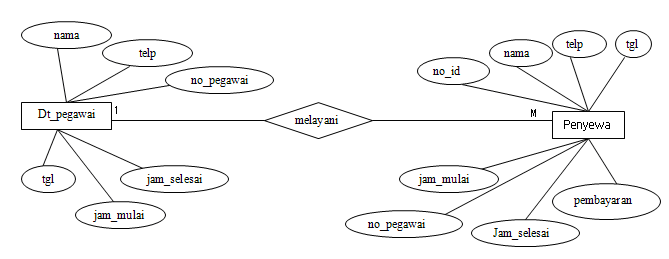
1. Tahap pembuatan ERD Awal (preliminary design)

Tujuan : memperoleh rancangan basisdata minimal yang dapat mengakomodasi kebutuhan penyimpanan data suatu sistem

Contoh:

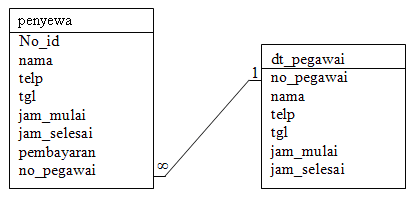
1. Tahap optimasi (final design)
2. Memperhatikan aspek efisiensi, performansi, dan fleksibiltas
3. Dilakukan dengan melakukan koreksi pada hasil tahap pertama
4. Bentuk koreksi :

* Pendekomposisian himpunan entitas
* Penggabungan himpunan entitas
* Pengubahan derajat relasi
* Penambahan relasi baru
* Pengubahan atribut-atribut untuk masing-masing himpunan entitas atau relasi

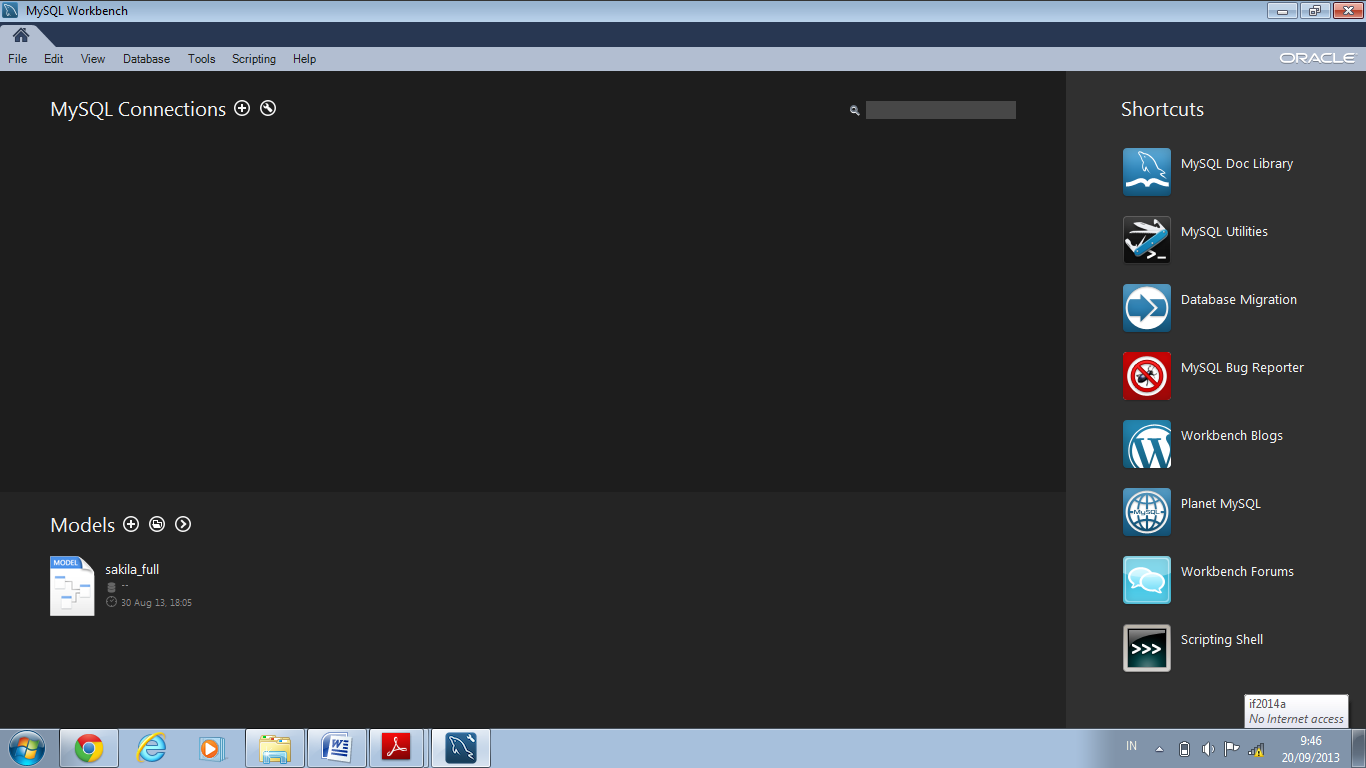
Contoh:

1. Langkah-langkah tahap pertama :
2. Mengidentifikasi dan menetapkan seluruh himpunan entitas yang akan terlibat
3. Menentukan atribut-atribut kunci dari masing-masing himpunan entitas
4. Mengidentifikasi dan menetapkan semua himpunan relasi di antara himp.entitas yang ada beserta foreign key nya
5. Menentukan kardinalitas/derajat relasi untuk setiap himpunan relasi
6. Melengkapi himpunan entitas dan himp.relasi dengan atribut-atribut deskriptif

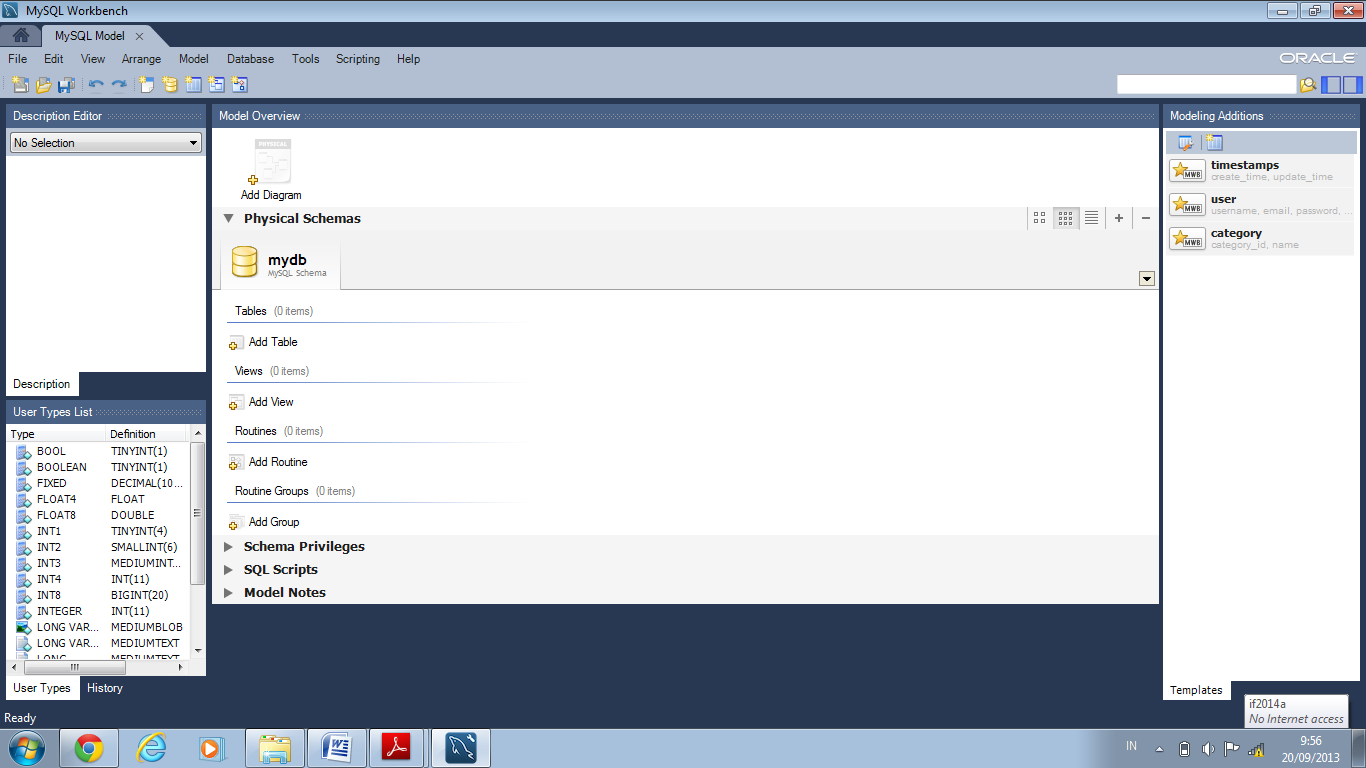
Berikut contoh ERD –nya:



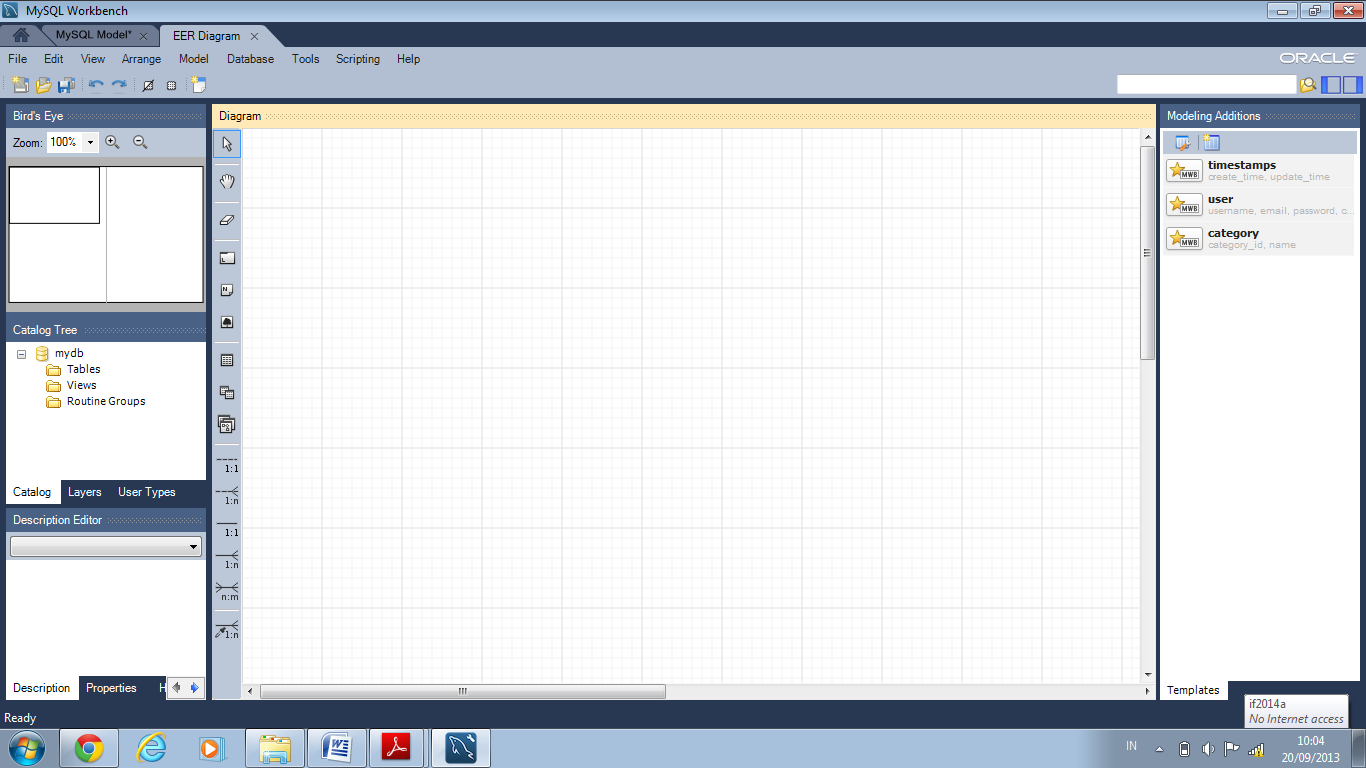
1. **Langkah-langkah Praktikum**
   * 1. **Jalankan MySQL Workbench**

****

* + 1. **Klik tanda + pada models untuk memilih Create new EER MODEL**

****

* + 1. **Pilih Add Diagram**

****

* + 1. **Buatlah ERD sederhana Praktikum Basis Data,seperti ini:**

ASISTEN DOSEN

Mengajar

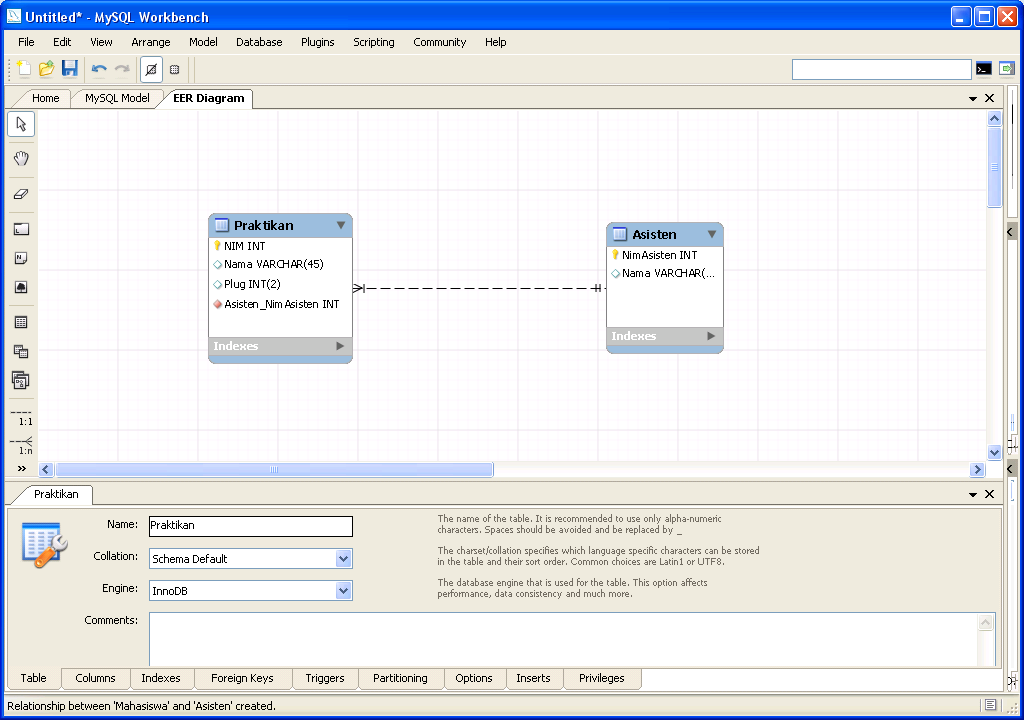
PRAKTIKAN

**Kamus Data:**

**Asisten Dosen = {NimAsisten,NamaAsisten}**

**PRAKTIKAN = {NIMPraktikan,NAMAPraktikan,PLUG}**

Mengajar = {NimAsisten,NimPraktikan}

****

1. **Tugas Praktikum Pertemuan I :**
2. Buatlah suatu permodelan(preliminary design) CBIS (untuk menginput mata kuliah) di Jurusan Teknik Informatika!
3. Buatlah final design nya!
4. Buatlah ERD nya!
5. Buatlah suatu paper mengenai Database (di dalamnya terdapat Sejarah Database, Pengertian Database, Konsep Database, Struktur fisik Database, Struktur Logikal Database, dan Administrasi User pada Database)

Pertemuan II

**Pengenalan MySQL**

1. **Tujuan**
   1. Mahasiswa memahami cara instalasi MySQL.
   2. Mahasiswa memahami cara konfigurasi MySQL
2. **Alokasi Waktu : 2 x 60 menit**
3. **Dasar Teori**

MySQL merupakan implementasi dari sistem manajemen basis data relational (RDBMS), seperti halnya ORACLE, Postgresql, MS SQL, dan sebagainya. MySQL dikembangkan sekitar tahun 1994 oleh sebuah perusahaan pengembang software dan konsultan database bernama MYSQL AB yang berada di Swedia. Waktu itu perusahaan tersebut masih bernama TcX DataKonsult AB, dan tujuan awal dikembangkannya MySQL adalah untuk mengembangkan aplikasi berbasis web pada client. MySQL menyebut produknya sebagai database open source terpopuler di dunia. Berdasarkan riset dinyatakan bahwa di platform Web, dan baik untuk kategori open source maupun umum, MySQL adalah database yang paling banyak dipakai. Menurut perusahaan pengembangnya, MySQL telah terpasang di sekitar 3 juta komputer. Puluhan hingga ratusan ribu situs mengandalkan MySQL.

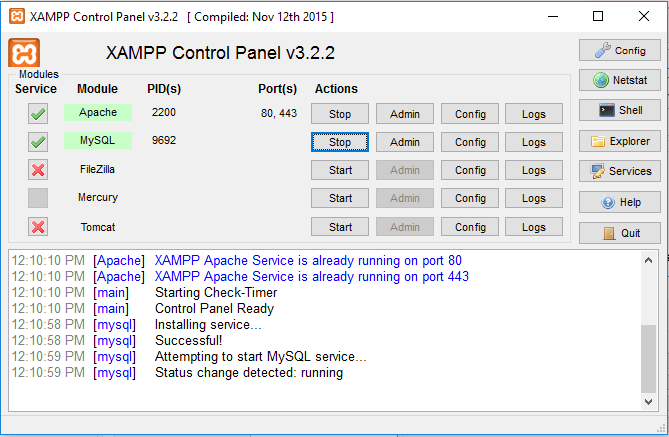
**Format Perintah**

Berikut adalah ketentuan-ketentuan memberi perintah pada MySQL:

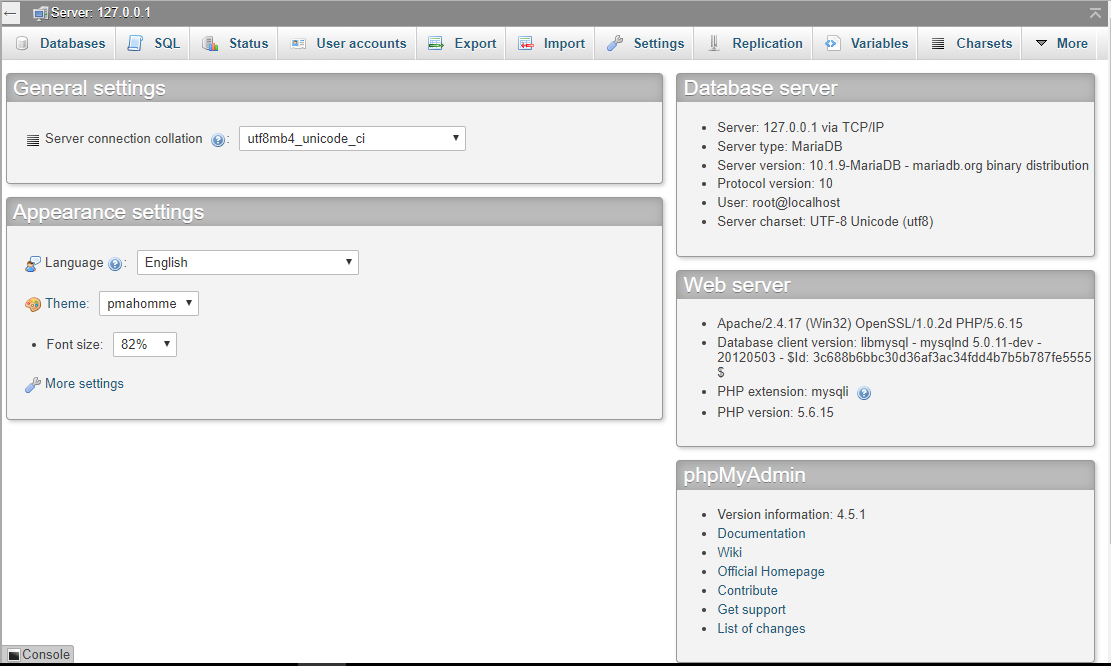
1. Setiap perintah harus diakhiri dengan tanda titik koma, kecuali untuk perintah tertentu, misal : quit
2. Setiap perintah akan disimpan dalam buffer (memori sementara) untuk menyimpan histori perintah-perintah yang pernah diberikan.Perintah dapat berupa perintah SQL atau perintah khusus MySQL.
3. Perintah-perintah dalam lingkungan MySQL tidak menerapkan aturan case sensitive, tetapi case insensitive yaitu perintah bisa dituliskan dalam huruf besar atau pun huruf kecil.
4. Aturan case sensitive diterapkan pada penamaan objek-objek dalam database seperti nama database atau nama table, namun aturan ini hanya ada dalam lingkungan Unix.

**Start dan Stop MySQL**

Untuk mulai menggunakan MySQL dengan bantuan XAMPP, pertama- tama aktifkan Apache (untuk server) dan MySQL (database), disarankan juga untuk menginstall service terlebih dahulu, agar fitur dari database dapat dipakai sepenuhnya.



Database MySQL yang telah diaktifkan dapat diakses melalui *Browser*, dengan mengetikkan alamat *localhost/phpmyadmin* pada *address bar* browser tersebut. Browser akan menampilkan laman administrator dari database, lengkap dengan berbagia tab dan tools untuk memanupulasi database.



Untuk keluar dari database MySQL ini, cukup tekan close pada browser, dan non-aktifkan MySQL dan Apache, serta uninstall service apabila dibutuhkan.

Pertemuan III

## PENGAMBILAN DATA DENGAN PERINTAH SQL SELECT

* + 1. **Tujuan :**

Setelah menyelesaikan praktikum ini, praktikan menguasai penggunaan perintah-perintah SQL SELECT untuk menampilkan data

* + 1. **Alokasi Waktu : 2 x 60 menit**
    2. **Dasar Teori**

Perintah dasar SELECT.

Sintaks dari pernyataan select adalah sebagai berikut :

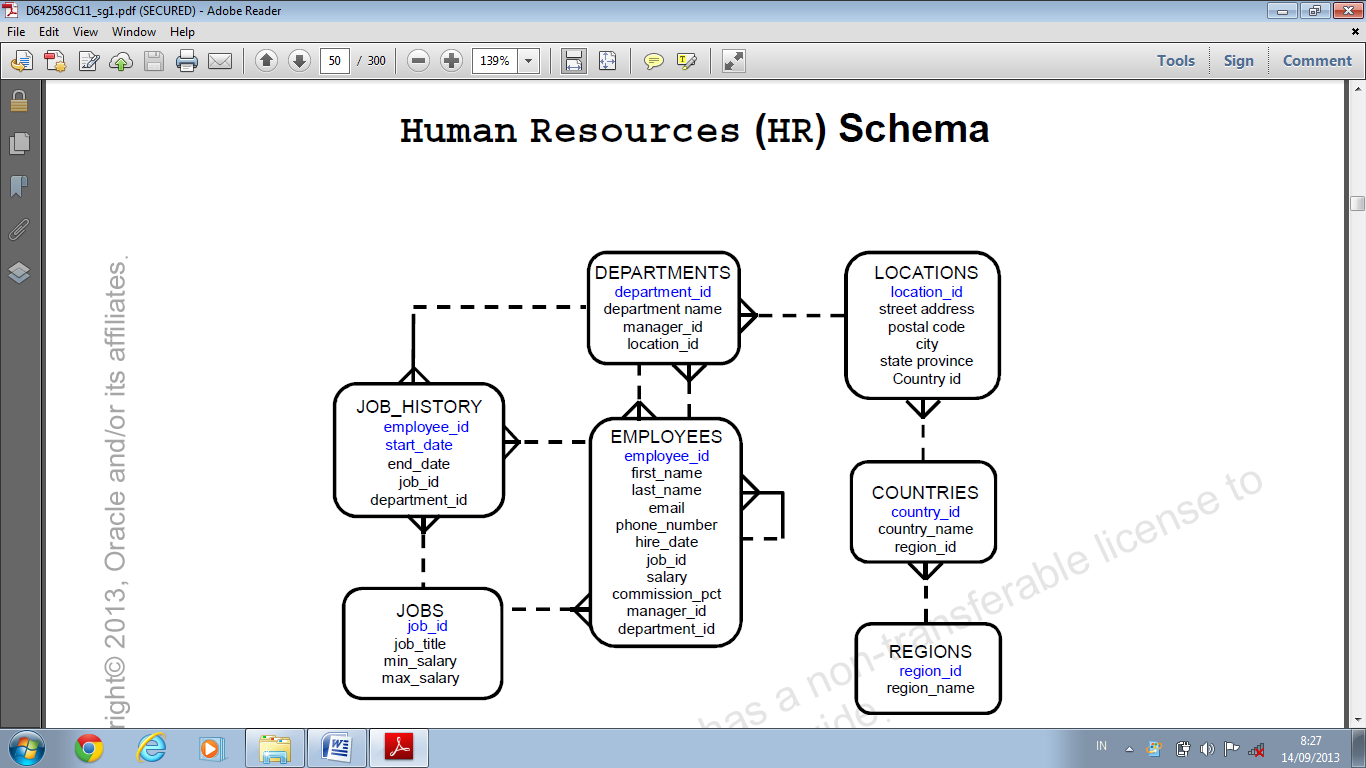
SELECT { \* | [DISTICNT] *column* | *expression* | [*alias*] .........}

FROM  *tabel ;*

* SELECT menentukan kolom yang akan ditampilkan
* FROM digunakan untuk menentukan tabel yang digunakan

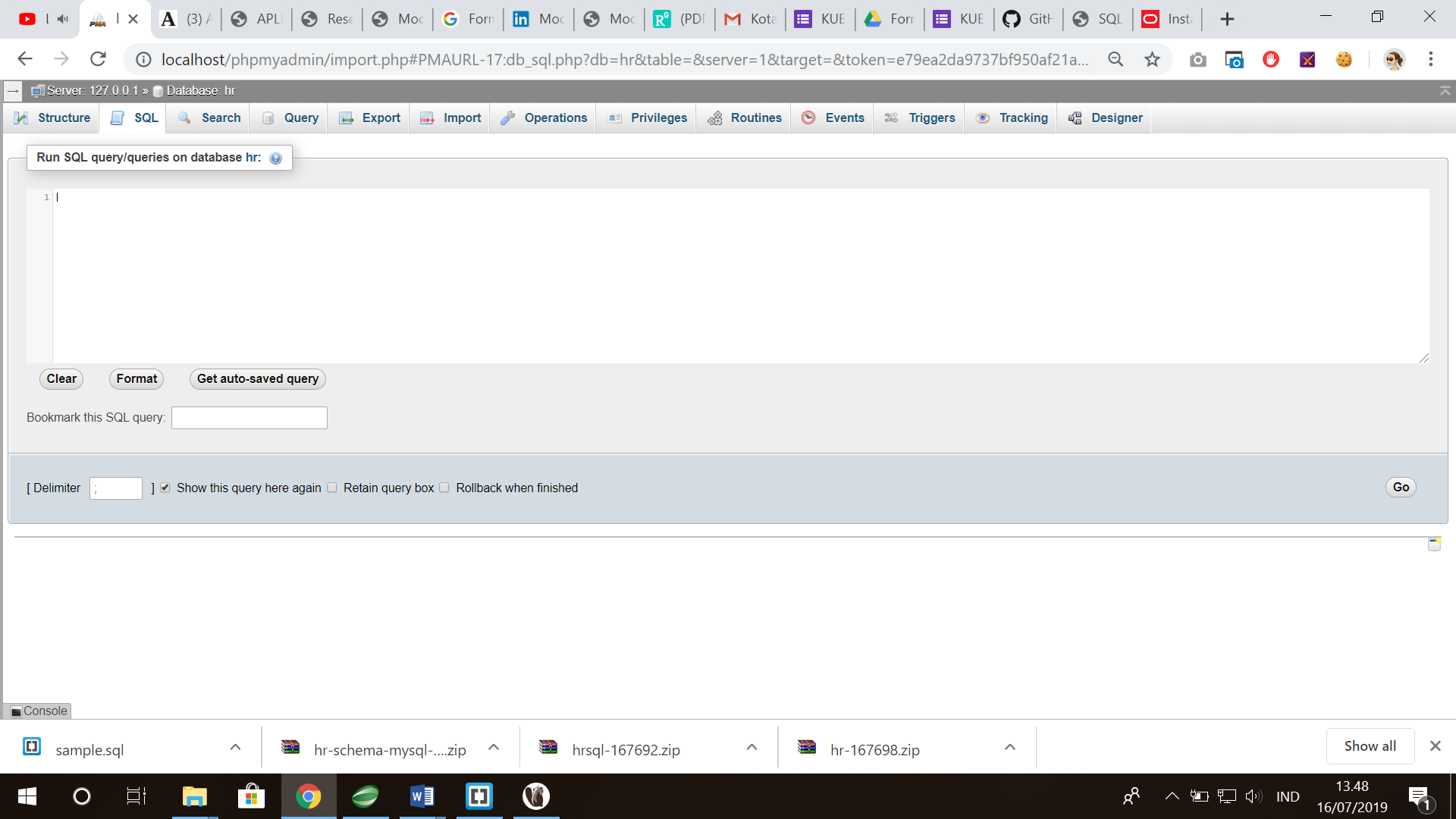
Pada pernyataan diatas, **SELECT** berfungsi untuk mengakses tabel, pernyataan all atau distinct menentukan apakah data diakses seluruhnya atau hanya yang beda saja (distinct).

Pada praktikum kali ini akan digunakan tabel-tabel HR dengan tabel-tabel beserta relasi tabelnya sebagai berikut : :



* + 1. **Langkah Praktikum**

Untuk menjalankan perintah SQL pada MySQL, pilih tab SQL sebagai berikut :



1. **Memanipulasi Nama Kolom dengan AS**
   1. menampilkan seluruh data tabel EMPLOYEES

SELECT \* FROM employees;

* 1. menampilkan last\_name dengan last\_name sebagai Nama Belakang

SELECT last\_name AS "Nama Belakang" FROM employees;

1. **Menggunakan ekspresi artimatika**

|  |  |
| --- | --- |
| Operator | Penjelasan |
| + | Penjumlahan |
| \_ | Pengurangan |
| \* | Perkalian |
| / | Pembagian |

a. menampilkan nama belakang, gaji dan gaji + 500 ;

SELECT last\_name, salary, salary + 500 FROM employees;

b. menampilkan nama belakang dan jumlah gaji selama 12 bulan

SELECT last\_name, 12\*salary AS "Gaji Setahun" FROM employees;

1. **Menghilangkan baris yang bernilai sama dengan DISTINCT**

SELECT department\_id from employees; (masih menampilkan baris

yang benilai sama )

SELECT DISTINCT department\_id from employees;

1. **Perintah DESCRIBE**

Perintah DESCRIBE untuk menampilkan stuktur dari sebuah tabel

**DESCRIBE employees ;**

DESCRIBE departments;

* + 1. **Tugas Praktikum**
    2. Tampilkan seluruh data yang terdapat pada tabel-tabel selain tabel employees pada SCHEMA HR (database)
    3. Tampilkan struktur tabel yang ada
    4. Pada tabel employees, tampilkan kolom sumbangan yang merupakan potongan 5% dari gaji
    5. Pada tabel employees, tampilkan kolom gaji baru setelah dikurangi sebesar 10 persen dari salary.

Pertemuan IV

## PERINTAH DDL UNTUK MEMBUAT TABEL

* + 1. **Tujuan**

Menguasai penggunaan perintah Data Definition Languange untuk membuat dan memodifikasi tabel

* + 1. **Alokasi Waktu : 2 x 60 menit**
    2. **Dasar Teori**

DDL (Data Definition Language), DDL merupakan kelompok perintah yang berfungsi untuk mendefinisikan atribut-atribut basis data, tabel, atribut(kolom), batasan-batasan terhadap suatu atribut, serta hubungan antar tabel. Yang termasuk dalam kelompok DDL ini adalah CREATE, ALTER, dan DROP.

1. **CREATE**

1. Membuat Database : **CREATE DATABASE namadatabase;**

Nama database tidak boleh mengandung spasi dan tidak boleh memiliki nama yang sama antar database. Syntax tambahan untuk menampilkan daftar nama database yang ada pada

mysql menggunakan perintah : **SHOW DATABASES;**

1. Membuat Tabel :

**CREATE TABLE namatabel2 ( Field1 TipeData1,Field2 TipeData2);**

Sebelum membuat suatu tabel, terlebih dahulu harus memilih salah satu database sebagai database aktif yang akan digunakan untuk menyimpan tabel-tabel. Nama tabel tidak boleh mengandung spasi (space). Field1 dan TipeData1 merupakan nama kolom pertama dan tipe data untuk kolom pertama. Jika ingin membuat tabel dengan kolom lebih dari satu, maka setelah pendefinisian tipe data sebelumnya diberikan tanda koma (,).

**Tipe Data MySQL**

Macam-macam Tipe Data pada MySQL sebenarnya mempunyai beberapa jenis, secara umum tipe-tipe data MySQL ini ada empat (4), diantaranya yaitu:

1. Tipe Data Numeric.

2. Tipe Data String.

3. Tipe Data Date.

4. Tipe Data Kelompok Himpunan.

**Tipe data Numerik**

Tipe data numerik yaitu tipe data [yang](https://www.blogger.com/null) digunakan untuk menyimpan data numerik (angka).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Jangkauan** | **Ukuran** |
| **1** | TINYINT | Menyimpan data bilangan bulat positif [dan](https://www.blogger.com/null) negatif. | -128 s/d 127 | 1 byte (8 bit). |
| **2** | SMALLINT | menyimpan data bilangan bulat positif [dan](https://www.blogger.com/null) negatif. | : -32.768 s/d 32.767 | 2 byte (16 bit). |
| **3** | MEDIUMINT | menyimpan data bilangan bulat positif [dan](https://www.blogger.com/null) negatif. | -8.388.608 s/d 8.388.607 | 3 byte (24 bit). |
| **4** | INT | menyimpan data bilangan bulat positif [dan](https://www.blogger.com/null) negative | -2.147.483.648 s/d 2.147.483.647 | 4 byte (32 bit). |
| **5** | BIGINT | menyimpan data bilangan bulat positif dan negatif. | ± 9,22 x 1018 | 8 byte (64 bit). |
| **6** | FLOAT | menyimpan data bilangan pecahan positif dan negatif presisi tunggal | -3.402823466E+38 s/d -1.175494351E-38, 0, dan  1.175494351E-38 s/d 3.402823466E+38. | 4 byte (32 bit) |
| **7** | DOUBLE | menyimpan data bilangan pecahan positif dan negatif presisi ganda. | -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308. | 8 byte (64 bit) |
| **8** | REAL | menyimpan data bilangan pecahan positif dan negatif presisi ganda. | -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308. | 8 byte (64 bit). |
| **9** | DECIMAL | menyimpan data bilangan pecahan positif dan negatif. | -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308. | 8 byte (64 bit). |
| **10** | NUMERIC | menyimpan data bilangan pecahan positif dan negatif. | -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308. | 8 byte (64 bit). |

**Tipe Data Date dan Time**

Tipe data date dan time yaitu tipe data [yang](https://www.blogger.com/null) digunakan untuk menyimpan data tanggal dan waktu.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Jangkauan** | **Ukuran** |
| **1** | DATE | menyimpan data tanggal | 1000-01-01 s/d 9999-12-31 (YYYY-MM-DD) | 3 byte. |
| **2** | TIME | menyimpan data waktu | -838:59:59 s/d +838:59:59 (HH:MM:SS) | 3 byte |
| **3** | DATETIME | menyimpan data tanggal dan waktu. | '1000-01-01 00:00:00' s/d '9999-12-31 23:59:59' | 8 byte |
| **4** | YEAR | menyimpan data tahun dari tanggal | 1900 s/d 2155 | 1 byte |

**Tipe Data BLOB (Biner)**

Tipe data blob digunakan untuk menyimpan data biner.

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Jangkauan** |
| **1** | BIT | Menyimpan data biner. | 64 digit biner |
| **2** | TINYBLOB | menyimpan data biner/ Gambar ukuran kecil | 255 byte |
| **3** | BLOB | Menyimpan data biner/ Gambar | 4 |
| **4** | MEDIUMBLOB | Menyimpan data biner/ Gambar kuran sedang | 224-1 byte |
| **5** | LONGBLOB | Menyimpan data biner/ Gambar ukuran besar | 232- 1 byte |

**Tipe Data String (Text)**

Tipe data string yaitu tipe data [yang](https://www.blogger.com/null) digunakan untuk menyimpan data string (text).

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Jangkauan** |
| **1** | CHAR | menyimpan data string ukuran tetap. | 0 s/d 255 karakter |
| **2** | VARCHAR | menyimpan data string ukuran dinamis. | 0 s/d 255 karakter (versi 4.1), 0 s/d 65.535 |
| **3** | TINYTEXT | menyimpan data text. | 0 s/d 255 karakter (versi 4.1), 0 s/d 65.535 |
| **4** | TEXT | menyimpan data text. | 0 s/d 65.535 |
| **5** | MEDIUMTEXT | menyimpan data text | 0 s/d 224 - 1 karakter |
| **6** | LONGTEXT | menyimpan data text. | 0 s/d 232 - 1 karakter |

**Tipe Data**[**yang**](https://www.blogger.com/null)**Lain**

Selain tipe data di atas, MySQL juga menyediakan tipe data yang lain, diantaranya adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Jangkauan** |
| **1** | ENUM | enumerasi (kumpulan data). | sampai dengan 65535 string. |
| **2** | SET | combination (himpunan data). | sampai dengan 255 string anggota |

1. **ALTER TABLE**

Perintah ALTER TABLE digunakan untuk :

1. Menambah kolom baru
2. Memodifikasi definisi kolom
3. Menghapus kolom

Syntaks :

**ALTER TABLE** mahasiswa **ADD** keterangan char(20);

**ALTER TABLE** nama\_tabel **MODIFY** nama\_field tipe\_data();

**ALTER TABLE** nama\_table **DROP** field\_yangdidrop;

1. **DROP TABLE**

Drop table digunakan untuk menghapus tabel dan dipindahkan ke recycle bin.

Untuk benar-benar menghapus tabel beserta isinya, harus ditambahkan PURGE ,

Syntaks :

DROP TABLE namatabel [PURGE]

**D. Langkah-langkah Praktikum**

Berikut ini adalah diagram ER Perpustakaan yang akan diterjemahkan menjadi tabel-tabel pada praktikum kali ini :



Kamus data :

Buku = { kode\_buku, judul, pengarang, jml\_buku, kode\_penerbit }

Penerbit = { Kode\_penerbit, nm\_penerbit }

Mahasiswa = { nim, nama, alamat, kota, tgl\_lhr, jenis\_kel }

Meminjam = { nim, kode\_buku, Tgl\_pinjam, Tgl\_hrs\_kembali, Tgl\_kembali }

1. Buat database baru dengan nama ***usernameanda\_*perpus**
2. Buatlah tabel mahasiswa berikut (sesuai dengan ER diagram) :

Mahasiswa :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Nim | Char | 9 |  |
| Nama | Char | 20 |  |
| Alamat | Char | 20 |  |
| Kota | Char | 15 |  |
| Tgl\_lhr | datetime |  |  |
| Jenis\_kel | Char | 1 |  |

Perintahnya adalah sebagai berikut :

**CREATE TABLE mahasiswa (**

**Nim char(9) NOT NULL PRIMARY KEY,**

**Nama char(20) default NULL,**

**Alamat char(20) default NULL,**

**Kota char(15) default NULL,**

**Tgl\_thr datetime default NULL,**

**Jenis\_kel char(1) default NULL**

**);**

1. Isikan data mahasiswa sebagai berikut :

**Mahasiswa**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nim | Nama | Alamat | Kota | TglLhr | Jenis\_kel |
| 123010001 | Ifan | Jl Babarsari | Yogya | 27-Juli-1980 | L |
| 123010002 | Adi | Jl Janti | Yogya | 28-Juni-1979 | L |
| 123010003 | Ayu | Jl Pemuda | Klaten | 10-Mei-1981 | P |
| 123010004 | Yulia | Jl Veteran | Sleman | 15-April-1970 | P |

Perintahnya adalah sebagai berikut :

**INSERT INTO** Mahasiswa

**VALUES** (‘123010001’,’Ifan’,’Jl Babarsari’,’Yogya’,’27-Juli-1980’,’L’)

Ulangi perintah diatas dengan data yang berbeda

1. Tampilkan seluruh data yang telah diisikan dengan perintah

SELECT \* FROM Mahasiswa;

**E. Tugas Praktikum**

1. Buatlah tabel table dengan struktur sebagai berikut :

Tabel Buku :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | Keterangan |
| Kode\_buku | Char | 4 |  |
| Judul | Char | 20 |  |
| Pengarang | Char | 15 |  |
| Jml\_buku | Int | - |  |
| Kode\_penerbit |  |  |  |

Tabel Penerbit :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | Keterangan |
| Kode\_penerbit | Char | 2 |  |
| Nm\_penerbit | Char | 15 |  |

Tabel Pinjam :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Tgl\_pinjam | Datetime |  |  |
| Mhs\_no\_induk | char | 9 |  |
| Buk\_kode\_buku | char | 4 |  |
| Tgl\_hrs\_kembali | Datetime |  |  |
| Tgl\_kembali | datetime |  |  |

1. Isikan tabel tabel terebut dengan data sebagai berikut :

Tabel Buku :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kode buku | Judul | Pengarang | Jml buku | Kode\_penerbit |
| S001 | DOS | M. Urip | 10 | 01 |
| BD01 | dBase III plus | Epsi B | 5 | 01 |
| BD04 | Clipper | Ahmad G | 4 | 02 |
| FI01 | Fisika | Sutrisno | 10 | 04 |

Tabel Penerbit :

|  |  |
| --- | --- |
| Kode\_penerbit | Nama Penerbit |
| 01 | Piksi ITB |
| 02 | Ganesha |
| 03 | Epsilon |
| 04 | Gramedia |

Tabel Pinjam :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tgl\_pinjam | Mhs\_no\_induk | Buk\_kode\_buku | Tgl\_hrs\_kembali | Tgl\_kembali |
| 2003/04/20 | 123010001 | S001 | 2003/04/23 | 2003/04/23 |
| 2003/04/20 | 123010002 | BD01 | 2003/04/23 | 2003/04/22 |
| 2003/04/20 | 123010003 | BD04 | 2003/04/23 | 2003/04/24 |
| 2003/04/21 | 123010003 | FI01 | 2003/04/24 | 2003/04/25 |
| 2003/04/21 | 123010004 | S001 | 2003/04/24 | 2003/04/25 |
| 2003/04/21 | 123010001 | BD01 | 2003/04/24 | 2003/04/24 |

Pertemuan V

## DEKLARASI CONSTRAINT

**A. Tujuan**

* 1. Memahami konsep dan penerapan constraint pada table
  2. Mampu mendeklarasikan dan menghapus constraint

**B. Alokasi Waktu : 2 x 60 menit**

**C. Dasar Teori**

*Constraint* adalah aturan untuk membatasi data yang dapat dimasukkan ke dalam suatu tabel. Tujuan pemberian constraint adalah untuk menjaga integritas data sehingga kekonsistenan dan kebenaran data yang disimpan terjamin. Integritas data dibagi dalam empat kategori :

* 1. Integritas Entitas

Integritas entiti dimaksudkan untuk memastikan bahwa tiap baris harus secara unik teridentifikasi oleh atribut yang disebut *primary key*. Primary key pada sebuah tabel tidak boleh NULL

* 1. Integritas Domain

Integritas domain memastikan bahwa hanya nilai yang sah diizinkan untuk disimpan pada sebuah kolom. Implementasi integritas domain dilakukan dengan memformat data, menentukan kisaran nilai atau jenis tipe data kolom yang sesuai

* 1. Integritas Referensi

Integritas referensi memastikan bahwa seluruh nilai di *foreign ke*y sesuai dengan yang ada di *primary key*. Caranya dengan menghubungkan primary key dengan foreign key sehingga data tersebut akurat, berguna dan konsisten

* 1. Integritas yang didefinisikan oleh pengguna

Berfungsi sebagai aturan untuk membuat data menjadi konsisten sesuai dengan yang diinginkan oleh pengguna, aturan ini juga dibuat oleh pengguna

#### Membuat Constraint

Sebuah constraint dapat dibuat bersamaan pada saat pembuatan tabel atau setelah pembuatan tabel. Pembuatan constraint mempunyai banyak cara , secara umum dengan sintaks sebagai berikut.

Syntax : **ALTER TABLE** *namatabel*

[ WITH CHECK | WITH NOCHECK ]

**ADD CONSTRAINT** *namaconstraint* tipe\_constraint

**Macam-macam constraint :**

1. **Primary Constraint**

Primary key constraint berguna untuk memastikan bahwa kolom primary key bersifat unik.

* 1. Dibuat pada saat pembuatan tabel

Contoh :

**CREATE TABLE** Mahasiswa

(

Nim char(9) **NOT NULL PRIMARY KEY**,

Nama char(20),

Alamat char(20),

Kota char(15),

Tgl\_thr date,

Jenis\_kel char(1)

)**;**

Dalam contoh ini akan dibuat tabel Mahasiswa dengan Nim sebagai primary key

* 1. Dibuat setelah tabel dibuat

Contoh : **ALTER TABLE** Mahasiswa

**ADD CONSTRAINT** pkMahasiswa **PRIMARY KEY** (Nim);

1. **Unique Constraint**

Unique constraint digunakan untuk membuat keunikan data di kolom yang bukan primary key. Unique constraint mirip dengan primary key constraint, bedanya didalam unique constraint diizinkan adanya nilai NULL, tapi hanya boleh satu baris saja.

1. **Foreign Key Constraint**

Foreign key constraint digunakan untuk menghilangkan ketidakkonsistenan antara dua tabel ketika data pada satu tabel bergantung pada data di tabel lainnya. Sebuah kolom foreign key ialah sebuah constaint yang didefinisikan dengan mereferensi ke primary key pada tabel lain

* 1. Dibuat pada saat pembuatan tabel

Contoh : **CREATE TABLE** Buku

(

Kode\_buku char(4) **NOT NULL PRIMARY KEY**,

Judul char(20),

Pengarang char(15),

Jml\_buku int,

Kode\_penerbit char(2),

CONSTRAINT fkKode\_penerbit FOREIGN KEY (Kode\_penerbit) REFERENCES penerbit(Kode\_penerbit)

)**;**

**CATATAN** : tabel yang akan direferensikan harus dibuat lebih dulu

Jika tabel sudah ada dan tidak memiliki foreign key, dapat dibuat sebagai berikut :

**ALTER TABLE** Buku

**ADD CONSTRAINT** fkKode\_penerbit **FOREIGN KEY**

(Kode\_penerbit) **REFERENCES** Penerbit(kode\_penerbit);

1. **Check Constraint**

Check constraint berfungsi untuk membuat integritas domain yang memaksa nilai yang dimasukkan ke tabel harus memenuhi syarat yang telah ditentukan.

1. **Default Constraint**

Default Constraint berguna untuk memasukkan sebuah nilai ke kolom, sehingga pengguna tidak perlu memasukkan nilai ke kolom tersebut

Contoh :

**CREATE TABLE** Mahasiswa

(

…..

Kota char(15) **DEFAULT** ‘Yogyakarta’

…..

) **;**

**D. Langkah Praktikum**

* + 1. Buatlah tabel mahasiswa berikut :

Mahasiswa :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Nim | Char | 9 | Primary key |
| Nama | Char | 20 |  |
| Alamat | Char | 20 |  |
| Kota | Char | 15 |  |
| Tgl\_lhr | Date |  |  |
| Jenis\_kel | Char | 1 |  |

Perintahnya adalah sebagai berikut :

**CREATE TABLE** Mahasiswa

(

Nim char(9) NOT NULL PRIMARY KEY,

Nama char(20),

Alamat char(20),

Kota char(15) DEFAULT ‘Yogyakarta’,

Tgl\_thr date,

Jenis\_kel char(1)

)**;**

**Catatan** : Jika sebelumnya telah ada tabel Mahasiswa, maka proses create tabel tidak bisa dilakukan.

* + 1. Isikan data mahasiswa sebagai berikut :

Mahasiswa

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nim | Nama | Alamat | Kota | TglLhr | Jenis\_kel |
| 123010001 | Ifan | Jl Babarsari |  | 27-Juli-1980 | L |
| 123010002 | Adi | Jl Janti |  | 28-Juni-1979 | L |
| 123010003 | Ayu | Jl Pemuda | Klaten | 10-Mei-1981 | P |
| 123010004 | Yulia | Jl Veteran | Sleman | 15-April-1970 | P |

Perintahnya adalah sebagai berikut :

**INSERT INTO Mahasiswa**

**VALUES ('123010002',' Ifan',' Jl Babarsari',' ','09-06-80','L')** ;

Ulangi perintah diatas dengan data yang berbeda

* + 1. Tampilkan seluruh data yang telah diisikan dengan perintah

**SELECT** \* **FROM** Mahasiswa

1. **Tugas Praktikum**
   1. Buatlah tabel tabel dengan struktur sebagai berikut :

Tabel Buku :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | Keterangan |
| Kode\_buku | Char | 4 | Primary key |
| Judul | Char | 20 |  |
| Pengarang | Char | 15 |  |
| Jml\_buku | Int | - |  |
| Kode\_penerbit |  |  | Foreign key |

Tabel Penerbit :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | Keterangan |
| Kode\_penerbit | Char | 2 | Primary key |
| Nm\_penerbit | Char | 15 |  |

Tabel Pinjam :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Tgl\_pinjam | date |  |  |
| Mhs\_no\_induk | char | 9 | Foreign key |
| Buk\_kode\_buku | char | 4 | Foreign key |
| Tgl\_hrs\_kembali | Date |  |  |
| Tgl\_kembali | Date |  |  |

* 1. Isikan tabel tabel tersebut dengan data sebagai berikut :

Tabel Buku :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kode buku | Judul | Pengarang | Jml buku | Kode\_penerbit |
| S001 | DOS | M. Urip | 10 | 01 |
| BD01 | dBase III plus | Epsi B | 5 | 01 |
| BD04 | Clipper | Ahmad G | 4 | 02 |
| FI01 | Fisika | Sutrisno | 10 | 04 |

Tabel Penerbit :

|  |  |
| --- | --- |
| Kode\_penerbit | Nama Penerbit |
| 01 | Piksi ITB |
| 02 | Ganesha |
| 03 | Epsilon |
| 04 | Gramedia |

Tabel Pinjam :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tgl\_pinjam | Mhs\_no\_induk | Buk\_kode\_buku | Tgl\_hrs\_kembali | Tgl\_kembali |
| 2003/04/20 | 123010001 | S001 | 2003/04/23 | 2003/04/23 |
| 2003/04/20 | 123010002 | BD01 | 2003/04/23 | 2003/04/22 |
| 2003/04/20 | 123010003 | BD04 | 2003/04/23 | 2003/04/24 |
| 2003/04/21 | 123010003 | FI01 | 2003/04/24 | 2003/04/25 |
| 2003/04/21 | 123010004 | S001 | 2003/04/24 | 2003/04/25 |
| 2003/04/21 | 123010001 | BD01 | 2003/04/24 | 2003/04/24 |

* 1. Isikan tabel Mahasiswa dengan data sebagai berikut :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 123010002 | Joko | Jl Kalasan |  | 25-Juni-1979 | L |

Amati apa yang terjadi dan mengapa bisa terjadi ?

* 1. Isikan tabel buku dengan data sebagai berikut

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FI02 | Fisika | Saroyo | 10 | 05 |

Amati apa yang terjadi mengapa bisa terjadi ?

* 1. Isikan tabel pinjam dengan data sebagai berikut

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2003/04/21 | 123010005 | S001 | 2003/04/24 | 2003/04/25 |

Amati apa yang terjadi dan mengapa bisa terjadi ?

Pertemuan VI

## PERINTAH SQL

* + 1. **Tujuan** 
       - 1. Menguasai penggunaan perintah-perintah SQL untuk memanipulasi data
         2. Menguasai penggunaan perintah-perintah SQL untuk memanipulasi data menggunakan Operator String, Operator logika, Klausa Group By, Having, dan Order By
    2. **Alokasi Waktu : 2 x 60 menit**
    3. **Dasar Teori**

Perintah yang paling sering digunakan untuk mengakses data didalam SQL adalah perintah SELECT. Secara umu sintaks dari pernyataan select adalah sebagai berikut :

**SELECT** [ ALL | DISTINCT ] *namakolom* [ **INTO** *tabel\_baru*]

**FROM** *namatabel*

[ **WHERE** klausa ]

[ **GROUP BY** klausa ]

[ **HAVING** klausa ]

[ **ORDER BY** klausa ]

Pada pernyataan diatas, **SELECT** berfungsi untuk mengakses tabel, pernyataan all atau distinct menentukan apakah data diakses seluruhnya atau hanya yang beda saja (distinct).

#### Operator String

Operator string menggunakan kata kunci **LIKE**, digunakan untuk memilih baris-baris sesuai dengan karakter yang digunakan

#### Operator Logika

Operator logika digunakan unuk pencarian data dengan beberapa kondisi. Operator logika tersebut adalah **OR** untuk menampilkan hasil ketika salah satu kondisi yang ditentukan bernilai benar, **AND** akan menampilkan hasil ketika seluruh kondisi yang ditentukan benar, dan **NOT** akan membalikkan nilai ekspresi

#### Klausa Group By

Klausa **Group By** digunakan untuk mengelompokkan data berdasarkan ekspresi group, misalnya berdasarkan kolom tertentu

#### Klausa Having

Klausa **Having** digunakan untuk mendefinisikan batasan seleksi berdasarkan GROUP BY. Biasanya dipakai pada saat penggunaan fungsi agregat

#### Klausa Order By

Klausa **Order By** digunakan untuk mengurutkan hasil sesuai dengan kolom yang ditentukan.

* + 1. **Langkah Praktikum**

1. **Memanipulasi Nama Kolom dengan AS**
   1. Menampilkan seluruh data tabel mahasiswa

**SELECT** \* **FROM** Mahasiswa;

* 1. Menampilkan nim, nama dengan kolom nim sebagai nomor induk mhs

**SELECT** nim **AS** ‘nomor induk mhs‘, nama **FROM** Mahasiswa;

1. **Memilih baris berdasarkan kriteria**

a. menampilkan seluruh data tabel mahasiswa dengan jenis kelamin L

**SELECT** \* **FROM** Mahasiswa **WHERE** jenis\_kel = ‘L’;

b. menampilkan seluruh data tabel mahasiswa yang beralamat di Jl Janti

**SELECT** \* **FROM** Mahasiswa **WHERE** alamat = ‘Jl Janti’;

c. menampilkan Kode\_buku, Judul buku, jml buku dari tabel buku dimana jumlah buku lebih besar dari 5

**SELECT** Kode\_buku, Judul, jml\_buku FROM Buku

**WHERE** jml\_buku **>** 5;

1. **Menghapus baris berdasarkan kriteria**

a. menghapus buku yang berjudul Clipper dari tabel Buku

**DELETE** Buku **WHERE** judul= ‘Clipper’

b. menghapus isi seluruh data dari tabel buku

**DELETE** Buku

1. **Operator Kisaran**

a. mengambil data buku yang jumlahnya antara 1 hingga 5 buah

**SELECT** Judul, Jml\_Buku **FROM** Buku

**WHERE** jml\_buku **BETWEEN** 1 **AND** 5;

b. mengambil data buku yang jumlahnya diluar rentang 1 hingga 5 buah

**SELECT** Judul, Jml\_Buku **FROM** Buku

**WHERE** jml\_buku **NOT BETWEEN** 1 **AND** 5;

1. **Operator List ( Operator IN )**

Mengambil data Mahasiswa yang berasal dari kota Yogya atau Klaten

**SELECT** Nim, Nama, Kota **FROM** Mahasiswa

**WHERE** Kota **IN** (‘Yogya’,’Klaten’);

1. **Operator String ( LIKE )**
   * 1. menampilkan seluruh nama mahasiswa dari tabel mahasiswa dimana nama mahasiswa dimulai dengan karakter ‘A’

**SELECT** nama **FROM** Mahasiswa

**WHERE** nama **LIKE** ‘A%’;

* + 1. menampilkan seluruh nama mahasiswa dari tabel mahasiswa dimana nama mahasiswa diakhiri dengan karakter ‘a’

**SELECT** nama **FROM** Mahasiswa

**WHERE** nama **LIKE** ‘%a’;

* + 1. menampilkan seluruh nama mahasiswa dari tabel mahasiswa dimana nama mahasiswa mengadung karakter ‘li’ ditengahnya

**SELECT** nama **FROM** Mahasiswa

**WHERE** nama **LIKE** ‘%li%’;

1. **Operator Logika**
   1. menampilkan seluruh data tabel mahasiswa yang berasal dari Yogya atau dari Klaten

**SELECT** \* **FROM** Mahasiswa

**WHERE** kota=’Yogya’ **OR** kota=’Klaten’;

* 1. menampilkan seluruh data tabel mahasiswa yang berasal dari Yogya dan berjenis kelamin L

**SELECT** \* **FROM** Mahasiswa

**WHERE** kota=’Yogya’ **AND** Jenis\_kel=’L’;

* 1. menampilkan seluruh data tabel mahasiswa yang berasal bukan berasal dari Yogya

**SELECT** \* **FROM** Mahasiswa

**WHERE** **NOT** kota=’Yogya’;

1. **Klausa GROUP BY**

Menampilkan seluruh data tabel mahasiswa dikelompokkan berdasarkan jenis kelamin

**SELECT** \* **FROM** Mahasiswa

**GROUP BY** jenis\_kel;

1. **Klausa HAVING**
   1. menampilkan banyaknya peminjaman yang dilakukan oleh mahasiswa yang lebih dari sekali pinjam

**SELECT** nim **FROM** Pinjam

**GROUP BY** nim **HAVING COUNT(\*)** > 1;

1. **Klausa ORDER BY**
2. menampilkan seluruh data dari tabel mahasiswa yang diurutkan berdasar kolom nama

**SELECT** \* **FROM** Mahasiswa

**ORDER BY** nama;

1. menampilkan seluruh data dari tabel mahasiswa yang diurutkan berdasar kolom nama secara menurun

**SELECT** \* **FROM** Mahasiswa

**ORDER BY** nama **DESC;**

1. menampilkan judul, jumlah buku dari tabel buku yang diurutkan berdasar jumlah buku

**SELECT** judul, jml\_buku **FROM** Buku

**ORDER BY** jml\_buku;

* 1. menampilkan judul, jumlah buku dari tabel buku yang diurutkan berdasar jumlah buku secara menurun

**SELECT** judul, jml\_buku **FROM** Buku

**ORDER BY** jml\_buku **DESC;**

* + 1. **Tugas Praktikum**
    2. Tampilkan kolom Nama, TglLhr sebagai tanggal lahir, Jenis\_kel sebagai Jenis Kelamin dari tabel Mahasiswa
    3. Tampilkan judul-judul buku dengan kode penerbit 01
    4. Tampilkan namamahasiswa yang berasal dari Yogya
    5. Tampilkan Judul buku yang jumlah bukunya lima
    6. Tampilkan Judul buku yang jumlah bukunya kurang dari sepuluh
    7. Tampilkan Judul buku yang jumlahnya lebih besar dari lima dan kurang dari sepuluh
    8. Tampilkan Judul-judul buku yang diterbitkan oleh kode\_penerbit 01,02
    9. Tampilkan Nama mahasiswa yang berasal dari yogya dan berjenis kelamin P
    10. Tampilkan Judul buku yang dikarang oleh Sutrisno dan jumlahnya lebih besar dari lima
    11. Tampilkan Judul buku dengan kode penerbit 01 dan jumahnya sama dengan lima

**Tugas Praktikum ( PR )**

1. Tampilkan Judul-judul buku yang diawali huruf D dari tabel buku
2. Tampilkan judul-judul buku yang diakhiri karakter ‘er’ tabel buku
3. Tampilkan Kota yang dimulai dengan huruh K
4. Tampilkan Judul buku yang jumlah bukunya kurang dari sepuluh dan kode penerbitnya 02
5. Tampilkan Judul buku yang jumlahnya lebih besar dari empat dan kurang dari sepuluh
6. Tampilkan Judul-judul buku yang tidak diterbitkan oleh kode\_penerbit 01
7. Tampilkan Nim, Nama, alamat , kota dari table mahasiswa dikelompokkan berdasarkan kota
8. Tampilkan seluruh data buku dari tabel buku dikelompokkan berdasarkan kode\_penerbit
9. Tampilkan seluruh data mahasiswa diurutkan berdasarkan nim secara menurun
10. Tampilkan seluruh data mahasiswa diurutkan berdasarkan Kota secara menurun.

Pertemuan VII

## PERINTAH SQL LANJUTAN

**A. Tujuan**

Menguasai penggunaan perintah-perintah SQL untuk memanipulasi data antar beberapa tabel

**B. Alokasi Waktu : 2 x 60 menit**

**C. Dasar Teori**

MySQL menyediakan sebuah metode untuk mengakses data yang berasal lebih dari satu tabel yang disebut dengan join. Join mengimplementasikan pernyataan SELECT berisi nama-nama dari kolom yang akan diakses dari tabel.

1. **Cross Join**

Sebuah penggabungan yang meliputi lebih dari satu tabel tanpa syarat apapun. Hasil dari cross join akan menghasilkan produk kartesian

1. **Natural / Inner Join**

Join yang menghasilkan baris-baris yang minimal ada sebuah baris di kedua tabel yang sesuai dengan kondisi join, baris-baris yang tidak sesuai dengan sebuah baris dari tabel lain.

1. **Outer Join**

Outer join akan menghasilkan semua data dari sebuah tabel dan membatasi data dari tabel lainnya. Terdapat tiga tipe outer join yaitu : left join, right join, dan full join. Semua baris dari tabel sebelah kiri diacu dengan sebuah left outer join, dan semua baris dari tabel sebelah kanan diacu dengan right outer join. Semua baris dari kedua tabel dihasilkan oleh sebuah full outer join. Kegunaan dari aouter join adalah untuk mencari record-record piatu, artinya record yang ada di sebuah tabel tetapi tidak mempunyai pasangan di tabel lain.

**D. Langkah Praktikum**

**1. Cross Join antara tabel Buku dan Penerbit**

Contoh :

**SELECT** \* **FROM** Buku **CROSS JOIN** Penerbit;

Jika tabel buku mempunyai 4 baris dan tabel Penerbit 4 baris maka hasil cross join adalah 16 baris

**2. Natural Join/Inner Join**

Contoh :

**SELECT** b.judul, b.Pengarang, p.Kode\_penerbit **FROM** Buku b

**INNER JOIN** Penerbit p **ON** b.kode\_penerbit = p.kode\_penerbit;

Perintah di atas akan membatasi kolom data yang bertumpuk, perintah diatas dapat diganti dengan memakai inner join style baru yang memakai kondisi **where :**

**SELECT** b.judul, b.Pengarang, p.Kode\_penerbit

**FROM** Buku b, Penerbit p

**WHERE** **b.kode\_penerbit = p.kode\_penerbit;**

**3. Left Outer Join**

Contoh :

**SELECT** b.judul, b.Pengarang, p.Kode\_penerbit **FROM** Buku b

**LEFT OUTER JOIN** Penerbit p **ON** b.kode\_penerbit=p.kode\_penerbit;

**4. Right Outer Join**

Contoh :

**SELECT** b.judul, b.Pengarang, p.Kode\_penerbit **FROM** Buku b

**RIGHT OUTER JOIN** Penerbit p **ON** b.kode\_penerbit=p.kode\_penerbit;

**5. Full Outer Join**

Contoh :

**SELECT** b.judul, b.Pengarang, p.Kode\_penerbit **FROM** Buku b

**FULL OUTER JOIN** Penerbit p **ON** b.kode\_penerbit=p.kode\_penerbit;

**E. Tugas Praktikum**

1. Rubahlah Nama penerbit Gramedia menjadi Elexmedia
2. Tampilkan semua Judul buku, pengarangnya beserta penerbitnya
3. Tampilkan buku buku yang diterbitkan oleh Piksi ITB
4. Tampilkan seluruh nim, nama dan judul buku yang dipinjamnya
5. Tampilkan Judul-judul buku yang dipinjam oleh mahasiswa dengan nim 123010003.

Pertemuan VIII

## FUNGSI AGREGAT, FUNGSI NUMERIK, FUNGSI STRING, FUNGSI TANGGAL

**A. Tujuan**

1. Menguasai peggunaan fungsi agregat, numerik, string dan tanggal dalam menyelesaikan query yang diinginkan
2. Mampu mengubah data dari satu tipe ke tipe lainnya

**B. Alokasi Waktu : 2 x 60 menit**

**C. Dasar Teori**

1. **Fungsi Agregat**

Fungsi agregat adalah fungsi yang bekerja terhadap suatu group, dimana fungsi-fungsi tersebut terdiri dari :

* **Count :** untuk menghitung cacah record/baris dari suatu tabel

Contoh : - untuk menampilkan banyaknya record mahasiswa :

**select** count(\*) **from** Mahasiswa;

- untuk menampilkan banyaknya mahasiswa angkatan 2006 :

**select** count(\*) **from** Mahasiswa

**where** nim **like** ‘12306%’;

* **Sum :** untuk menghitung jumlah dari sekumpulan kolom dari suatu tabel

Contoh : - untuk menampilkan total sks untuk kuliah di semester 2:

**select** sum(sks) **from** Kuliah

**where** semester = 2;

* **Avg :** untuk mendapatkan nilai rata-rata nilai kolom tertentu pada suatu tabel

Contoh : - untuk menampilkan rata-rata sks untuk semua mata kuliah

**select** avg(sks) **from** Kuliah;

* **Max :** Untuk mendapatkan nilai terbesar dari sekumplulan nilai kolom dari suatu tabel

Contoh : - untuk menampilkan nilai terbesar yang diperoleh mahasiswa

untuk mata kuliah dengan kode kuliah ‘IF-110’

**select** max(nilai) **from** nilai

**where** kode\_kul = ‘IF-110’;

* **Min :** untuk mendapatkan nilai terkecil dari sekumpulan nilai kolom dari suatu tabel

Contoh : - untuk menampilkan tanggal lahir paling tua yang ada di tabel mahasiswa

**select** min(tgl\_lahir) **from** Mahasiswa;

1. **Fungsi Numerik**

Format standar dalam penggunaan operasi aritmatika. Tabel berikut ini adalah fungsi-fungsi numerik yang sering dipakai disertai penjelasan kegunaannya :

|  |  |
| --- | --- |
| **Fungsi** | **Penjelasan** |
| ROUND | Membulatkan ekspresi numerik ke nilai presisi sesuai denganekspresi integer yang ditentukan |
| TRUNC | Memotong nilai sesuai desimal yang ditentukan |
| MOD | Menghasilkan sisa pembagian |

Contoh penggunaan :

SELECT ROUND (10.3676,2); akan menghasilkan 10.36

SELECT TRUNCATE(10.28372,1) from dual ; akan menghasilkan 10.2

SELECT MOD (11, 2); akan menghasilkan 1

1. **Fungsi Character**

**Fungsi Karakter** atau **Character Function**adalah sebuah fungsi yang digunakan untuk merubah tipe inputan yang tadinya bertipe karakter menjadi bertipe [**number**](http://risnotes.com/2012/01/number-function/)maupun bertipe [**karakter**](http://risnotes.com/2012/01/fungsi-karakter/).

Ada 2 jenis Character Function :

1. Case Manipulation Function
2. Character Manipulation Function

## Case Man****ipulation F****unction

Case Manipulation Function yaitu fungsi yang digunakan untuk memanipulasi kata. Case Manipulation Function terbagi menjadi 3 jenis yaitu :

* + - 1. **LOWER**

LOWER merupakan fungsi yang digunakan untuk merubah kata menjadi huruf kecil

* + - 1. **UPPER**

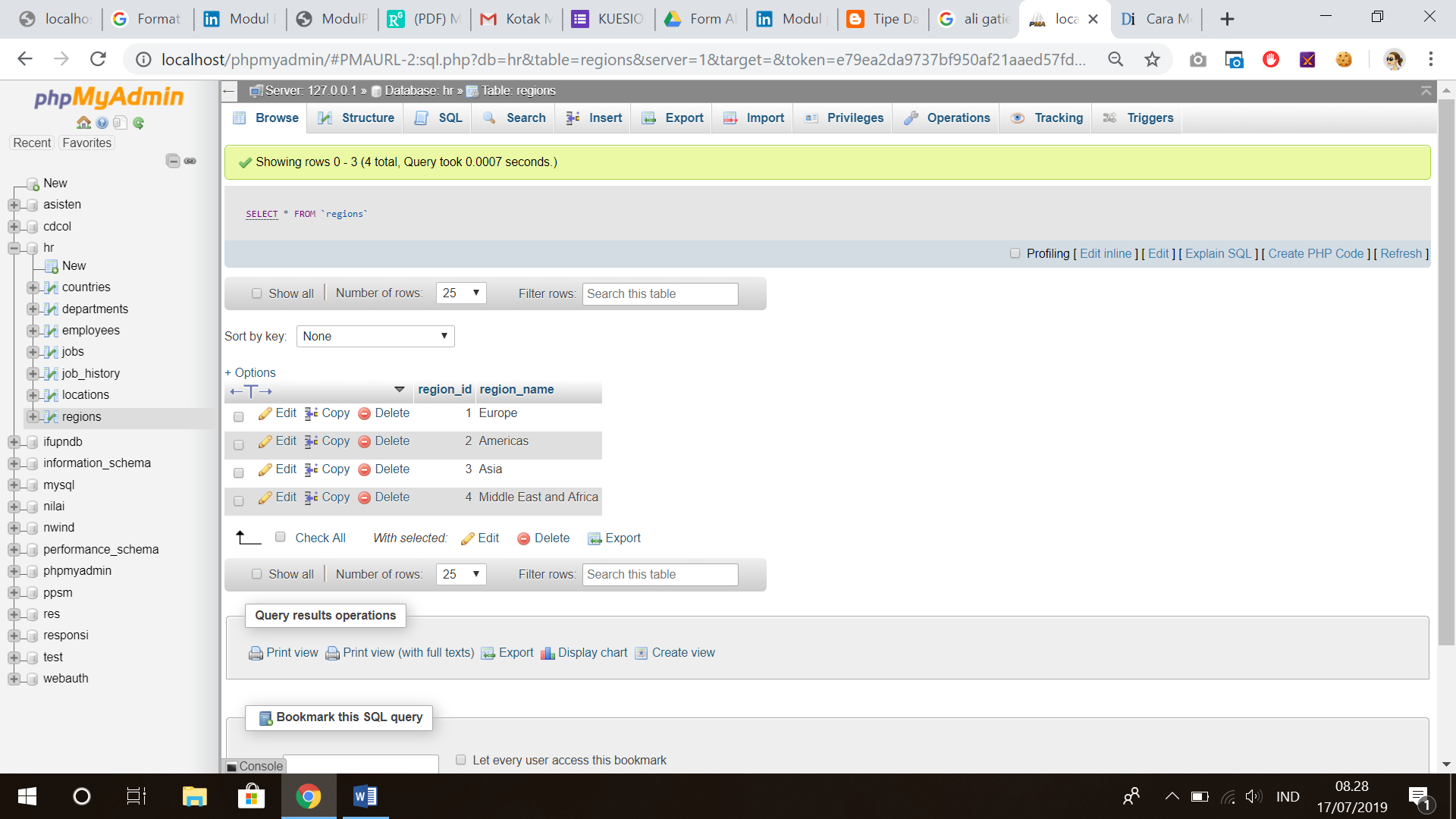
Kebalikan dari LOWER, UPPER merupakan fungsi yang digunakan untuk merubah kata menjadi huruf capital

* + - 1. **INITCAP**

INITCAP merupakan fungsi yang digunakan untuk mengubah karakter awal pada kata menjadi huruf capital

Contoh penggunaan :

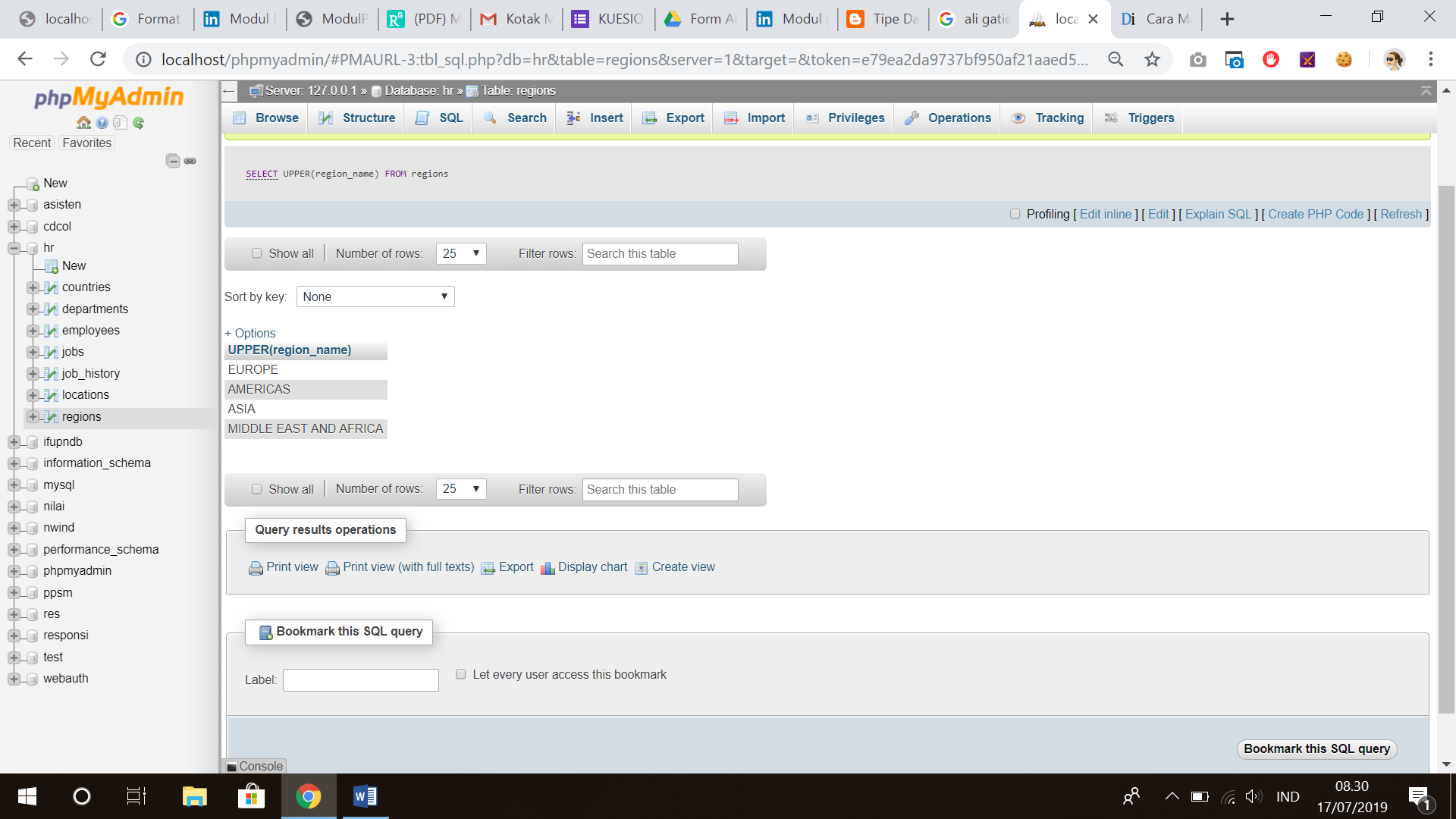
Terdapat tabel region pada database HR sebagai berikut



Kemudian diberi fungsi UPPER dengan syntax :

SELECT UPPER(region\_name) FROM region;

Maka hasilnya adalah :



## Character Manipulation Function

*Character Manipulation Function* merupakan fungsi yang digunakan untuk memanipulasi karakter. Berikut merupakan fungsi-fungsi dari karakter :

1. **CONCAT**

CONCAT atau Concatenation merupakan fungsi yang digunakan untuk menggabungkan 2 kata menjadi satu. Contoh :

SELECT CONCAT('w3resource','.','com');

1. **SUBSTR**

SUBSTR merupakan fungsi yang digunakan untuk mengambil sejumlah karakter didalam satu kata atau kalimat. Contoh:

SELECT SUBSTR('w3resource',4,3);

1. **LENGTH**

LENGTH merupakan fungsi yang digunakan untuk menghitung panjang karakter dalam satu kata atau kalimat. Contoh :

SELECT LENGTH('text');

1. **INSTR**

INSTR digunakan untuk menentukan pada posisi ke berapa karakter yang ingin kita ketahui. Contoh:

SELECT INSTR('myteststring','st');

1. **LPAD**

LPAD merupakan fungsi yang digunakan untuk menambahkan karakter sebanyak n, yang diletakkan di sebelah kiri suatu kata. Contoh :

SELECT LPAD('hi',1,'\*\*');

1. **RPAD**

RPAD merupakan kebalikan dari LPAD, menambahkan karakter sebanyak n di sebelah kanan. Contoh :

SELECT RPAD('w3resource',15,'\*');

1. **TRIM**

TRIM digunakan untuk menghapus atau memotong karakter yang kita inginkan. Contoh :

SELECT TRIM(' trim ');

SELECT TRIM(LEADING 'leading' FROM 'leadingtext' );

SELECT TRIM(BOTH 'leadtrail' FROM 'leadtrailtextleadtrail');

1. **REPLACE**

REPLACE merupakan fungsi yang digunakan untuk mengganti kata dengan kata yang kita inginkan. Contoh :

SELECT REPLACE('w3resource','ur','r');

1. **Fungsi Tanggal**

**Fungsi tanggal atau Date Function** merupakan fungsi yang digunakan untuk mengoperasikan nilai yang bertipe DATE. Untuk menampilkan tanggal dan waktu saat ini (server), menggunakan fungsi NOW(). Selain itu, dapat juga menggunakan CURDATE() yang akan menghasilkan tanggal saat ini dan CURTIME() yang akan menghasilkan waktu saat ini (tanpa tanggal). Contoh :

SELECT NOW(), CURDATE(), CURTIME();

Format default tanggal di MySQL adalah **YYYY-MM-DD**. Format tersebut harus diikuti, terutama saat menyimpan nilai kedalam field yang bertipe DATE. Seringkali dalam menyajikan data ke user, memerlukan format tanggal yang berbeda dari format tanggal default tersebut. Misalnya ingin menampilkan tanggal dalam format DD/MM/YYYY atau format panjang seperti ‘September 20th, 2010’. Fungsi **DATE\_FORMAT() dapat digunakan** untuk mengatur tampilan tanggal dan waktu. Dengan fungsi ini kita dapat menampilkan tanggal dalam berbagai format. Contoh :

SELECT DATE\_FORMAT(NOW(), '%d/%m/%Y') as tanggal1, DATE\_FORMAT(NOW(), '%M %D, %Y') as tanggal2;

**D. Tugas Praktikum**

* 1. Tampilkan Jumlah Mahasiswa yang ada dalam tabel Mahasiswa
  2. Tampilkan Jumlah Penerbit yang ada
  3. Tampilkan jumlah buku terbanyak
  4. Tampilkan Jumlah buku paling sedikit
  5. Tampilkan total buku per penerbit
  6. Tampilkan total judul buku per penerbit yang lebih besar dari 2

Pertemuan IX

## STUDI KASUS BASIS DATA PEGAWAI

* + 1. **Tujuan**

Menguasai pembuatan tabel dan mampu memanipulasi data-data sesuai dengan kebutuhan

**B. Alokasi Waktu : 2 x 60 menit**

**C. Diagram ER**

Berikut ini adalah studi kasus tentang Kepegawaian dengan ER diagram sebagai berikut.



Kamus data :

Jabatan = { kode\_jabatan, namajabatan }

Pegawai = { nip, namapegawai, alamat, kode\_kota ,kode\_jabatan, gaji }

Absensi = { bulan, nip ,jml\_masuk }

Kota = { kode\_kota , namakota }

1. **Langkah Praktikum**

1. Buatlah tabel-tabel berikut (sesuai dengan ER diagram) :

Tabel Jabatan :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Kode\_jabatan | Char | 4 | Primary key |
| Namajabatan | Char | 20 |  |

Tabel Pegawai :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Nip | Char | 4 | Primary key |
| namapegawai | Char | 25 |  |
| Alamat | Char | 30 |  |
| Kode\_kota | Char | 3 | Foreign key |
| Kode\_jabatan | Char | 4 | Foreign key |
| Gaji | Int |  |  |

Tabel Absensi :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Nip | Char | 4 | foreign key |
| Bulan | Tinyint |  | Val 1 s.d 12 |
| Jml\_msk | Int |  |  |

Tabel Kota :

|  |  |  |  |
| --- | --- | --- | --- |
| Nama field | Type data | Panjang | keterangan |
| Kode\_kota | char | 3 | Primary key |
| Namakota | char | 25 |  |

1. Isikan tabel tabel terebut dengan data sebagai berikut :

Tabel Jabatan

|  |  |
| --- | --- |
| 001 | Programer |
| 002 | Operator |
| 003 | System Analis |
| 004 | Pimpinan Proyek |

Tabel Pegawai

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 111 | Totok | Jln Ganesha | BDG | 001 | 700000 |
| 222 | Udin | Jln Thamrin | JKT | 002 | 500000 |
| 333 | Budi | Jln Kalasan | DIY | 004 | 1500000 |
| 444 | Ahmad | Jln Taman Sari | BDG | 003 | 1000000 |

Tabel Absensi

|  |  |  |
| --- | --- | --- |
| 1 | 111 | 22 |
| 1 | 222 | 21 |
| 1 | 333 | 21 |
| 1 | 444 | 20 |
| 2 | 111 | 20 |
| 2 | 222 | 21 |
| 2 | 333 | 23 |
| 2 | 444 | 20 |
| 3 | 111 | 22 |
| 3 | 222 | 20 |
| 3 | 333 | 21 |
| 3 | 444 | 18 |
| 4 | 111 | 19 |
| 4 | 222 | 21 |
| 4 | 333 | 20 |
| 4 | 444 | 20 |

Tabel Kota

|  |  |
| --- | --- |
| BDG | Bandung |
| JKT | Jakarta |
| CRB | Cirebon |
| DIY | Yogyakakarta |

1. **Tugas Praktikum**

Berdasarkan tabel-tabel diatas buatlah perintah SQL untuk memenuhi permintaan-permintaan berikut ini :

1. Tampilkan nama pegawa nip dan kodekota pegawai yang tinggal di bandung
2. Tampilkan seluruh pegawai dengan atribut nip, namapegawai, alamat kota, kode kota, kodejabatan, gaji beserta namakota dan namajabatan yang sesuai
3. Tampilkan sub total gaji untuk setiap kelompok kota dengan attribute kodekota, namakota dan total gaji per kota
4. Tampilkan sub total gaji untuk setiap kelompok kota dengan attribute kodekota, namakota dan total gaji per kota termasuk kota yang pegawainya tidak ada
5. Hitung jumlah hari masuk dari tabel absensi untuk setiap pegawai, selama empat bulan dengan atribut nip, namapegawai dan jml masuk
6. Tampilkan rata-rata jumlah hari masuk untuk setiap bulan dengan atribut bulan dan jmlmasuk