MODUL PRAKTIKUM

APLIKASI BASIS DATA

Disusun Oleh:

Dini Silvi Purnia, M.Kom

S1 Sistem Informasi

STMIK Nusa Mandiri Jakarta

Kata Pengantar

Puji Syukur kehadirat Tuhan Yang Maha Esa karena atas limpahan rahmat-Nya sehingga kami dapat menyelesaikan modul Aplikasi Basis Data. Modul ini disusun berdasarkan Standar RPS kampus STMIK Nusa Mandiri, Modul ini juga dilengkapi dengan latihan soal dan tugas untuk menguji pemahaman Mahasiswa terkait dengan materi yang terdapat pada modul. Kami menyadari masih banyak kekurangan dalam penyusunan modul ini. Oleh karena itu, kami sangat mengharapkan kritik dan saran demi perbaikan dan kesempurnaan modul ini.

Kami mengucapkan terima kasih kepada berbagai pihak yang telah membantu proses penyelesain modul ini,. Semoga modul ini dapat bermanfaat bagi kita semua, khususnya para peserta didik.

Jakarta, 24 Januari 2020

Penyusun

DAFTAR ISI

BAB 1

PENGENALAN MYSQL

MySQL adalah Sebuah program database server yang mampu menerima dan mengirimkan datanya sangat cepat, multi user serta menggunakan peintah dasar SQL (Structured Query Language). MySQL merupakan dua bentuk lisensi, yaitu FreeSoftware dan Shareware. MySQL yang biasa kita gunakan adalah MySQL FreeSoftware yang berada dibawah Lisensi GNU/GPL (General Public License). MySQL Merupakan sebuah database server yang free, artinya kita bebas menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensinya. MySQL pertama kali dirintis oleh seorang programmer database bernama Michael Widenius . Selain database server, MySQl juga merupakan program yang dapat mengakses suatu database MySQL yang berposisi sebagai Server, yang berarti program kita berposisi sebagai Client. Jadi MySQL adalah sebuah database yang dapat digunakan sebagai Client mupun server. Database MySQL merupakan suatu perangkat lunak database yang berbentuk database relasional atau disebut Relational Database Management System (RDBMS) yang menggunakan suatu bahasa permintaan yang bernama SQL (Structured Query Language).

Database MySQL memiliki beberapa kelebihan dibanding database lain, diantaranya :

- MySQL merupakan Database Management System (DBMS)
- MySQL sebagai Relation Database Management System (RDBMS) atau disebut dengan database Relational
- MySQL Merupakan sebuah database server yang free, artinya kita bebas menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensinya

- MySQL merupakan sebuah database client
- MySQL mampu menerima query yang bertupuk dalam satu permintaan atau MultiThreading
- MySQL merupakan Database yang mampu menyimpan data berkapasitas sangat besar hingga berukuran GigaByte sekalipun.
- MySQL diidukung oleh driver ODBC, artinya database MySQL dapat diakses menggunakan aplikasi apa saja termasuk berupa visual seperti visual Basic dan Delphi.
- MySQL adalah database menggunakan enkripsi password, jadi database ini cukup aman karena memiliki password untuk mengakses nya.
- MySQL merupakan Database Server yang multi user, artinya database ini tidak hanya digunakan oleh satu pihak orang akan tetapi dapat digunakan oleh banyak pengguna.
- MySQL mendukung field yang dijadikan sebagai kunci primer dan kunci uniq (Unique).
- MySQL memliki kecepatan dalam pembuatan table maupun peng-update an table.

Berbagai type data pada MySQL dapat dilihat pada tabel berikut :

Tabel 1. Type Data untuk Bilangan (Number)

Type Data	Keterangan
TINYINT	Ukuran 1 byte. Bilangan bulat terkecil, dengan jangkauan untuk bilangan bertanda: -128 sampai dengan 127 dan untuk yang tidak bertanda : 0 s/d 255. Bilangan tak bertandai dengan kata UNSIGNED

SMALLINT	Ukuran 2 Byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -32768 s/d 32767 dan untuk yang tidak bertanda : 0 s/d 65535
MEDIUMINT	Ukuran 3 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -8388608 s/ d 8388607 dan untuk yang tidak bertanda : 0 s/d 16777215
INT	Ukuran 4 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -2147483648 s/d 2147483647 dan untuk yang tidak bertanda : 0 s/d 4294967295
INTEGER	Ukuran 4 byte. Sinonim dari int BIGINT Ukuran 8 byte. Bilangan bulat terbesar dengan jangkauan untuk bilangan bertanda : -9223372036854775808 s/d 9223372036854775807 dan untuk yang tidak bertanda : 0 s/d 1844674473709551615
FLOAT	Ukuran 4 byte. Bilangan pecahan DOUBLE Ukuran 8 byte. Bilangan pecahan
DOUBLEPRECISION	Ukuran 8 byte. Bilangan pecahan
REAL	Ukuran 8 byte. Sinonim dari DOUBLE

DECIMAL (M,D)	Ukuran M byte. Bilangan pecahan, misalnya DECIMAL(5,2 dapat digunakan untuk menyimpan bilangan -99,99 s/d 99,99
NUMERIC (M,D)	Ukuran M byte. Sinonim dari DECIMAL, misalnya NUMERIC(5,2) dapat digunakan untuk menyimpan bilangan -99,99 s/d 99,99

Tabel 2. Type Data untuk Tanggal dan waktu

Type Data	Keterangan
DATETIME	Ukuran 8 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1000-01-01 00:00:00' s/d '9999-12-31 23:59:59'
DATE	Ukuran 3 Byte. Tanggal dengan jangkauan dari '1000-01-01' s/d '9999-12-31'
TIME STAMP	Ukuran 4 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1970-01-01 00:00:00' s/d '2037'
TIME	Ukuran 3 byte. Waktu dengan jangkauan dari '839:59:59' s/d '838:59:59'
YEAR	Ukuran 1 byte. Data tahun antara 1901 s/d 2155

Tabel 3. Type Data Karakter dan lain-lain

Type Data	Keterangan
CHAR	Mampu menangani data hingga 255 karakter. Tipe data CHAR mengharuskan untuk memasukkan data yang telah ditentukan oleh kita
VARCHAR	Mampu menangani data hingga 255 karakter. Tipe data VARCHAR tidak mengharuskan untuk memasukkan data yang telah ditentukan oleh kita
TINYBLOB, TINYTEXT	Ukuran 255 byte. Mampu menangani data sampai 2^8-1 data
BLOB, TEXT	Ukuran 65535 byte. Type string yang mampu menangani data hingga 2^16-1 (16M-1) data
MEDIUMBLOB, MEDIUMTEXT	Ukuran 16777215 byte. Mampu menyimpan data hingga 2^24-1 (16M-1) data
LONGBLOB, LONGTEXT	Ukuran 4294967295 byte. Mampu menyimpan data hingga berukuran GIGA BYTE. Tipe data ini memiliki batas penyimpanan hingga 2^32-1 (4G-1) data

ENUM(‘nilai1’,’nilai2’,…,’nilaiN’)	Ukuran 1 atau 2 byte. Tergantung jumlah nilai enumerasinya (maksimum 65535 nilai)
SET(‘nilai1’,’nilai2’,…,’nilaiN’)	1,2,3,4 atau 8 byte, tergantung jumlah anggota himpunan (maksimum 64 anggota)

BAB II

STRUCTURED QUERY LANGUAGE

SQL adalah sebuah bahasa permintaan database yang terstruktur. Bahasa SQL ini dibuat sebagai bahasa yang dapat merelasikan beberapa tabel dalam database maupun merelasikan antar database. SQL dibagi menjadi tiga bentuk Query, yaitu :

1. DDL (Data Definition Language)

DDL adalah sebuah metode Query SQL yang berguna untuk mendefinisikan data pada sebuah Database, Query yang dimiliki DDL adalah :

- CREATE : Digunakan untuk membuat Database dan Tabel
- Drop : Digunakan untuk menghapus Tabel dan Database
- Alter : Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat, baik menambah Field (Add), mengganti nama Field (Change) ataupun menamakannya kembali (Rename), dan menghapus Field (Drop).

2. DML (Data Manipulation Language)

DML adalah sebuah metode Query yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari Query DML ini untuk melakukan pemanipulasian database yang telah dibuat. Query yang dimiliki DML adalah :

- INSERT : Digunakan untuk memasukkan data pada Tabel Database
- UPDATE : Digunakan untuk pengubahan terhadap data yang ada pada Tabel Database
- DELETE : Digunakan untuk Penhapusan data pada tabel Database

3. DCL (Data Control Language)

DCL adalah sebuah metode Query SQL yang digunakan untuk memberikan hak otorisasi mengakses Database, mengalokasikan space, pendefinisian space, dan pengauditan

penggunaan database. Query yang dimiliki DCL adalah :

GRANT : Untuk mengizinkan User mengakses Tabel dalam Database.

REVOKE : Untuk membatalkan izin hak user, yang ditetapkan oleh perintah GRANT

COMMIT : Mentapkan penyimpanan Database • ROLLBACK : Membatalkan penyimpanan Database

MySQL adalah suatu perangkat lunak database relasi (Relational Database Management System atau RDBMS), seperti halnya ORACLE, Postgresql, MS SQL, dan sebagainya. MySQL adalah database yang paling banyak dipakai.

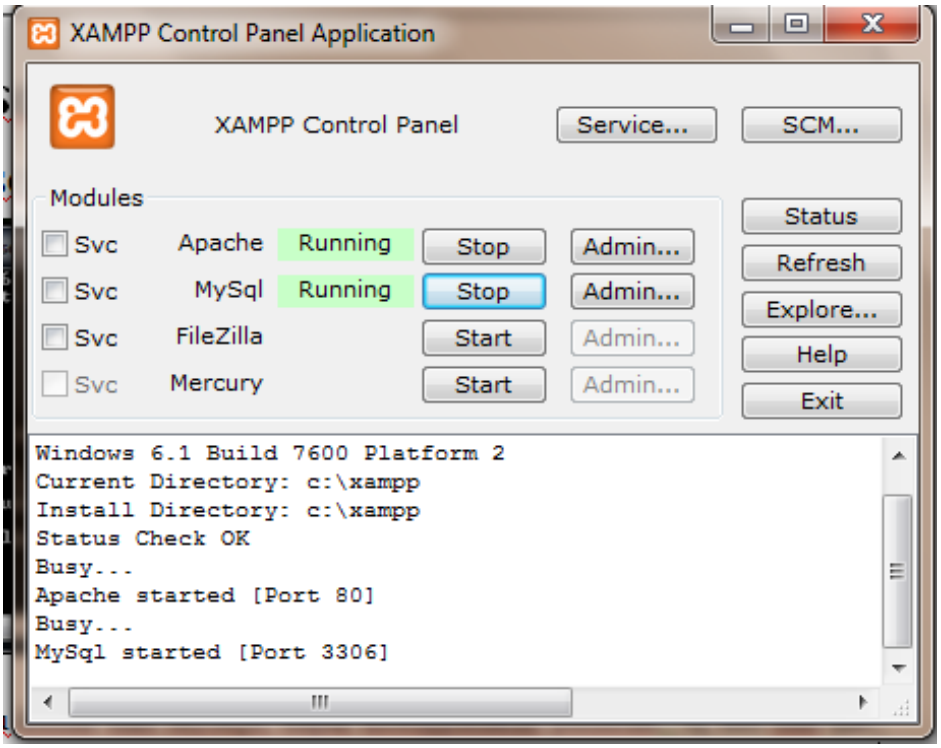
ketentuan-ketentuan memberi perintah pada MySQL:

- Perintah dapat berupa perintah SQL atau perintah khusus MySQL.
- Setiap perintah harus diakhiri dengan tanda titik koma, kecuali untuk perintah tertentu, misalnya : quit
- Setiap perintah akan disimpan dalam buffer (memori sementara) untuk menyimpan histori perintah-perintah yang pernah diberikan.
- Perintah-perintah dalam lingkungan MySQL tidak menerapkan aturan *case sensitive*, tetapi *case insensitive* yaitu perintah bisa dituliskan dalam huruf besar atau pun huruf kecil.

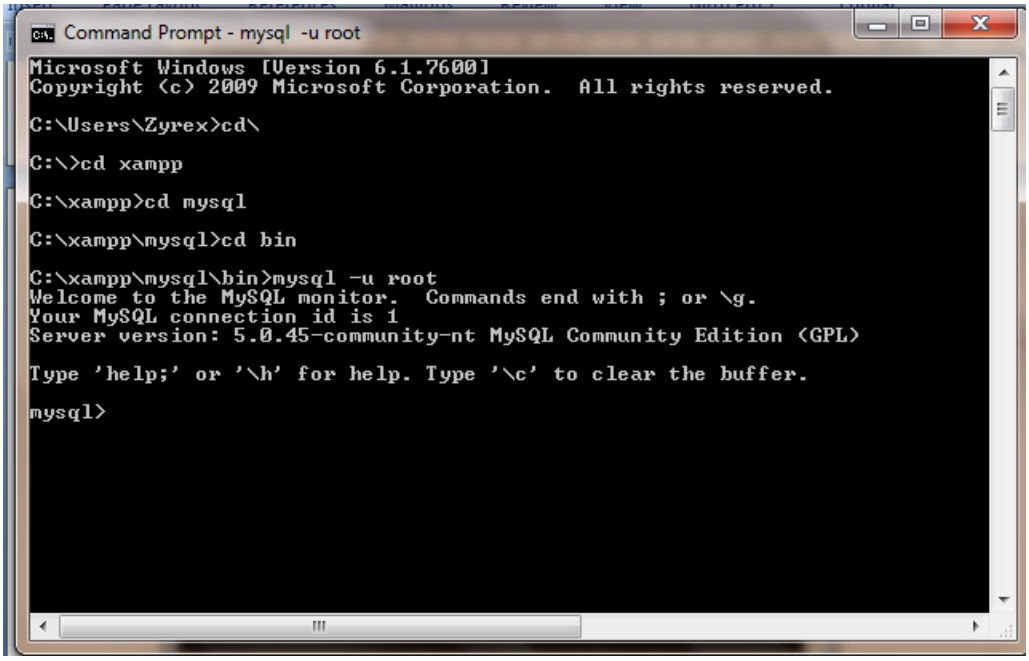
BAB III

CARA MEMULAI MYSQL

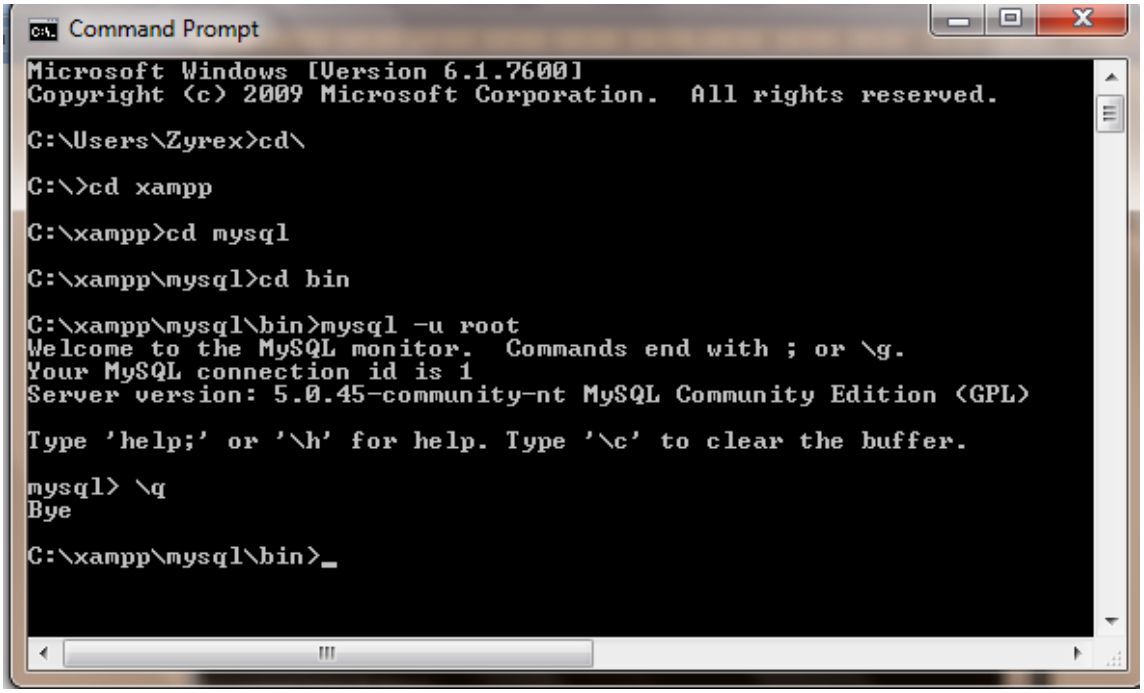
Berikut cara memulai MySQL dengan menggunakan bantuan XAMPP. Aktifkan Xampp Control Panel Application, klik start apache dan mysql.



Aktifkan command prompt, lalu ketik seperti gambar berikut:

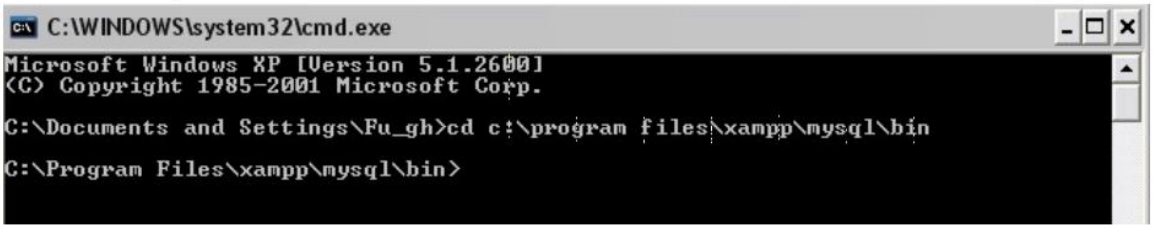


Sedangkan untuk stop atau keluar dari MySQL dapat menggunakan perintah : \q, exit dan quit.



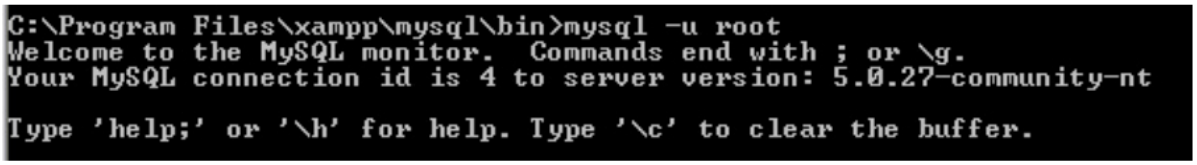
Untuk dapat menggunakan MySQL terlebih dahulu aktifkan Server MySQL dengan menghidupkan daemond MySQL. Program MySQL yang digunakan pada modul ini adalah XAMPP 1.7, maka untuk menjalankan daemond MySQL terdapat pada direktori yaitu C:\Program Files\Xampp\Mysql\Bin

Untuk masuk kedalam server MySQL, bukalah MS-DOS Prompt anda melalui Run kemudian ketik Command atau cmd. Maka anda dapat masuk ke dalam direktori MySQL melalui MS-DOS Promtp seperti dibawah ini.



Masuk dan Keluar dari Server MySQL

MySQL adalah sebuah database server yang sangat aman. MySQL memiliki kemampuan memanajemen user dalam mengakses. Jadi, tidak sembarang user dapat mengakses sebuah database yang diciptakan MySQL. Maka sebelum anda memiliki User untuk mengakses MySQL anda juga dapat Mengakses database MySQL menggunakan User Root. Berikut adalah perintah yang digunakan untuk mengkoneksikan kedalam Server Mysql : Shell > MySQL –u Root –p Enter Password: ********* Keterangan : Tanda –u menerangkan bahwa kita akan masuk menggunakan User Name bernama Root. Tanda –p menyatakan kita akan masuk menggunakan Password. Edited By Haris Saputro Halaman 5 Modul Pembelajaran Praktek Basis Data (MySQL) 2012 Berikut adalah perintah yang digunakan untuk mengkoneksikan kedalam Server Mysql melalui Root : Shell> Mysql –u root



Untuk dapat keluar dari Server MysQL kita dapat mengetikkan Intruksi quit ata \q : Mysql> quit Bye Mysql> \q Bye

Database MySQL menyediakan beberapa fasilitas bantuan yang berguna untuk mendokumentasikan atau memanipulasikan server yaitu dengan cara mengetikan intruksi \h atau \?. Mysql> \?

Semua Query harus diakhiri dengan tanda titik koma (;). Tanda ini menunjukkan bahwa query telah berakhir dan siap dieksekusi.

- Help (\h) : Digunakan untuk menampilkan file bantuan pada MySQL
- ? (\?) : Perintah ini sama dengan perintah Help
- Clear (\c) : Berguna untuk membersihkan atau menggagalkan semua perintah yang telah berjalan dalam suatu prompt
- Connect (\r) : untuk melakukan penyegaran koneksi ke dalam database yang ada pada Server Host
- Ego (\G) : berguna untuk menampilkan data secara horizontal. Go (\g) : member perintah server untuk mengeksekusi
- tee (\T) : mengatur tempat file yang akan didokumentasikan. Edited By Haris Saputro Halaman 6 Modul Pembelajaran Praktek Basis Data (MySQL) 2012
- Contoh :
- mysql> \T d:\belajar mysql.doc
- Logging to file ‘d:\data.doc;’

- Note (\t) : akhir dari (\T) yang berguna untuk mendokumentasikan semua query.
- Print (\p) : mencetak semua query yang telah kita perintahkan kelayar.
- Prompt (\R) : Mengubah prompt standar sesuai keinginan.
- Source (\.) : berguna untuk mengeksekusi query dari luar yang berbentuk .sql
- Use (\u) : berguna untuk memasuki database yang akan digunakan maupun mengganti database yang akan di gunakan.

Administrasi MySQL

MySQL Selaku database server yang mampu berjalan pada jaringan, tentu saja MySQL harus memiliki kemampuan khusus yang berguna untuk melakukan manajemen user atau mendukung system databaseyang bersifat client/server.

Membuat User baru Untuk dapat menciptkan user baru pada database mysql yang terdapat pada tabel user. Dapat dilakukan dengan menggunakan pernyataan SQL bernama INSERT. Sintax seperti berikut :

INSERT INTO user(host,user,password) VALUES(‘%’,’nama_user’,’password’);

Contoh :

mysql> INSERT INTO user(host,user,password) VALUES('localhost','haris',MD5('if060017'));

Query OK, 1 row affected, 4 warnings (0.00 sec)

Seteleh anda memberikan perintah diatas, berikan perintah :

FLUSH PRIVILEGES;

Contoh : mysql> FLUSH PRIVILEGES;

Query OK, 0 rows affected (0.00 sec)

Memberikan Wewenang Untuk User

Apabila User telah dibuat terlebih dahulu dan lupa untuk memberikan Hak Wewenang untuk User. Kita dapat memberikan hak wewenang dengan menggunakan Perintah Query UPDATE. Sintax yang digunakan seperti berikut :

```
UPDATE user
SET select_priv =’y’,
Insert_priv =’y’,
Update_priv =’y’,
Delete_priv =’y’,
Create_priv =’y’,
Drop_priv =’y’,
Alter_priv =’y’
WHERE user =’haris’;
```

BAB IV

DATA DEFENITION LANGUAGE

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. Syntax Membuat Database : CREATE DATABASE namadatabase;

   Namadatabase 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;

2. Memilih Database : USE namadatabase;

   Sebelum membuat suatu tabel, terlebih dahulu harus memilih salah satu database sebagai database aktif yang akan digunakan untuk menyimpan tabel-tabel

3. Syntax Menghapus Database : DROP DATABASE namadatabase;

   Database yang akan dihapus sesuai dengan namadatabase.

4. Membuat Tabel : CREATE TABLE namatabel2 (Field1 TipeData1,Field2 TipeData2);

   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 (,).

5. Menampilkan Tabel

   Untuk menampilkan daftar nama tabel yang ada pada database yang sedang aktif/digunakan (dalam hal ini database rental) : SHOW TABLES;

6. Menampilkan Atribut Tabel : DESC namatabel;

   Untuk menampilkan deskripsi tabel (dalam hal ini jenisfilm) syntaxnya adalah : DESC barang;

7. Syntax Menghapus Tabel : DROP TABLE namatabel;

   Tabel yang akan dihapus sesuai dengan namatabel, berikut ini perintah untuk menghapus tabel dengan nama jenisfilm : DROP TABLE BARANG;

8. Mendefinisikan Null/Not Null : CREATE TABLE namatabel (Field1 TipeData1 NOT NULL, Field2 TipeData2);

9. Mendefinisikan Primary Key Pada Tabel

   Terdapat tiga cara untuk mendefinisikan primary key. Berikut ini adalah Syntax mendefinisikan primary key untuk Field1

   CREATE TABLE namatabel(Field1 TipeData1 NOT NULL PRIMARY KEY, Field2 TipeData2);

   Atau

   CREATE TABLE namatabel (Field1 TipeData1, Field2 TipeData2, PRIMARY KEY(Field1));

   Atau

   ALTER TABLE namatabel ADD CONSTRAINT namaconstraint PRIMARY KEY (namakolom);

10. Menghapus Primary Key Pada Tabel

    Cara 1 : Jika primary key dibuat dengan menggunakan alter table :

    ALTER TABLE namatabel DROP CONSTRAINT namaconstraint;

Cara 2 : Jika primary key dibuat melalui create table :

ALTER TABLE namatabel DROP PRIMARY KEY;

11. Menambah Kolom Baru Pada Tabel : ALTER TABLE namatabel ADD fieldbaru tipe;

    Namatabel adalah nama tabel yang akan ditambah fieldnya. Fieldbaru adalah nama kolom yang akan ditambahkan, tipe adalah tipe data dari kolom yang akan ditambahkan.

    Berikut ini contoh perintah untuk menambah kolom keterangan dengan tipe data varchar(25):

    ALTER TABLE JENISFILM ADD KETERANGAN VARCHAR(25);

    Untuk meletakkan field diawal, tambahkan sintaks first :

    ALTER TABLE PELANGAN ADD COLUMN KODE CHAR(5) FIRST;

    Untuk menyisipkan field setelah field tertentu, tambahkan sintaks after :

    ALTER TABLE PELANGAN ADD COLUMN PHONE CHAR(5) AFTER ALAMAT;

12. Mengubah Tipe Data atau Lebar Kolom Pada Tabel : ALTER TABLE NAMATABEL MODIFY COLUMN FIELD TIPE

    Namatabel adalah nama tabel yang akan diubah tipe data atau lebar kolomnya. Field adalah kolom yang akan diubah tipe data atau lebarnya. Tipe adalah tipe data baru atau tipe data lama dengan lebar kolom yang berbeda. Berikut ini contoh perintah untuk mengubah tipe data untuk kolom keterangan dengan char(20) :

    ALTER TABLE JENISFILM MODIFY COLUMN KETERANGAN VARCHAR(20);

13. Mengubah Nama Kolom : ALTER TABLE namatabel CHANGE COLUMN namalamakolom namabarukolom tipedatabaru;

    Namatabel adalah nama tabel yang akan diubah nama kolomnya, namalamakolom adalah kolom yang akan diganti namanya, namabarukolom adalah nama baru kolom, tipedatanya adalah tipe data dari kolom tersebut. Berikut ini contoh perintah untuk

mengubah nama kolom keterangan menjadi ket :

```
ALTER TABLE JENISFILM CHANGE COLUMN KETERANGAN KET VARCHAR(20);
```

14. Menghapus Kolom Pada Tabel : ALTER TABLE namatabel DROP COLUMN namakolom;

BAB V

DATA MANIPULATION LANGUAGE

DML (Data Manipulation Language) DML adalah kelompok perintah yang berfungsi untuk memanipulasi data dalam basis data, misalnya untuk pengambilan, penyisipan, pengubahan dan penghapusan data. Perintah yang termasuk dalah kategori DML adalah : INSERT, DELETE, UPDATE dan SELECT.

1. I N S E RT

Perintah INSERT digunakan untuk menambahkan baris pada suatu tabel. Terdapat dua cara untuk menambah baris, yaitu:

Cara 1: Menambah baris dengan mengisi data pada setiap kolom :

```
INSERT INTO namatabel VALUES (nilai1,nilai2,nilai-n);
```

Cara 2 : Menambah baris dengan hanya mengisi data pada kolom tertentu :

```
INSERT INTO namatabel (kolom1,kolom2,kolom-n) VALUES (nilai1,nilai2,nilai-n);
```

Keterangan :

Jika data bertipe string, date atau time (contoh : action, horor, 2007-11-10) maka pemberian nilainya diapit dengan tanda petik tunggal ('horor') atau petik ganda ("horor"). Jika data bertipe numerik (2500, 400) maka pemberian nilainya tidak diapit tanda petik tunggal maupun ganda.

2. DELETE

Perintah DELETE digunakan untuk menghapus satu baris, baris dengan kondisi tertentu atau seluruh baris. Syntax : DELETE FROM namatabel [WHERE kondisi];

Perintah dalam tanda [] bersifat opsional untuk menghapus suatu baris dengan suatu kondisi tertentu.

3. UPDATE

Perintah UPDATE digunakan untuk mengubah isi data pada satu atau beberapa kolom pada suatu table. Syntax :

```
UPDATE namatabel SET kolom1 = nilai1, kolom2 = nilai2 [WHERE kondisi];
```

Perintah dalam tanda [] bersifat opsional untuk mengubah suatu baris dengan suatu kondisi tertentu.

4. SELECT

Perintah SELECT digunakan untuk menampilkan isi dari suatu tabel yang dapat dihubungkan dengan tabel yang lainnya.

a. Menampilkan data untuk semua kolom menggunakan asterisk (*) :

   ```
   SELECT * FROM namatabel;
   ```

b. Menampilkan data untuk kolom tertentu :

   ```
   SELECT kolom1,kolom2,kolom-n FROM namatabel;
   ```

c. Menampilkan data dengan kondisi data tertentu dengan klausa WHERE:

   ```
   SELECT * FROM namatabel WHERE kondisi;
   ```

   Beberapa operator perbandingan yang dapat digunakan pada klausa WHERE adalah "="(sama dengan) , > (lebih dari), < (kurang dari), < > (tidak sama dengan), >= (lebih dari atau sama dengan), <= (kurang dari atau sama dengan). Adapun operator lain, yaitu : AND, OR, NOT, BETWEEN-AND, IN dan LIKE.

Perintah—perintah Select

a. Memberikan nama lain pada kolom : SELECT namakolomlama AS namakolombaru FROM namatabel;

Berikut ini perintah untuk memberikan nama lain pada kolom jenis menjadi jenis_film pada tabel jenisfilm:

```
SELECT JENIS AS TYPE FROM JENISFILM;
```

b. Menggunakan alias untuk nama tabel: SELECT namalias .jenis, namalias .harga FROM namatabel namalias;

Berikut ini perintah untuk memberikan alias pada tabel jenisfilm :

```
SELECT J.JENIS, J.HARGA FROM JENISFILM J;
```

c. Menampilkan data lebih dari dua tabel: SELECT * FROM namatabel1, namatabel2, namatabel-n;

d. Nested Queries / Subquery (IN, NOT IN, EXISTS, NOT EXISTS)

Subquery berarti query di dalam query. Dengan menggunakan subquery, hasil dari query akan menjadi bagian dari query di atasnya. Subquery terletak di dalam klausa WHERE atau HAVING. Pada klausa WHERE, subquery digunakan untuk memilih baris-baris tertentu yang kemudian digunakan oleh query. Sedangkan pada klausa HAVING, subquery digunakan untuk memilih kelompok baris yang kemudian digunakan oleh query.

Contoh 1: perintah untuk menampilkan data pada tabel jenisfilm yang mana data pada kolomjenis-nya tercantum pada tabel film menggunakan IN :

```
SELECT * FROM JENISFILM WHERE JENIS IN (SELECT JENIS FROM FILM);
```

atau menggunakan EXISTS

```
SELECT * FROM JENISFILM WHERE EXISTS (SELECT * FROM FILM WHERE
HARGA > 2000);
```

Pada contoh di atas:

SELECT JENIS FROM FILM disebut subquery, sedangkan :

SELECT * F ROM JENISFILM berkedudukan sebagai query. Perhatikan, terdapat data jenis dan harga pada tabel jenisfilm yang tidak ditampilkan. Hal ini disebabkan data pada kolom jenis tidak terdapat pada kolom jenis di tabel film.

Contoh 2: perintah untuk menampilkan data pada tabel jenisfilm yang mana data pada kolom jenis-nya tidak tercantum pada tabel film menggunakan NOT IN:

```
SELECT * FROM JENISFILM WHERE JENIS NOT IN (SELECT JENIS FROM
FILM);
```

atau menggunakan NOT EXISTS

```
SELECT * FROM JENISFILM WHERE NOT EXISTS (SELECT * FROM FILM
WHERE HARGA > 2000);
```

e. Operator comparison ANY dan ALL

Operator ANY digunakan berkaitan dengan subquery. Operator ini menghasilkan TRUE (benar) jika paling tidak salah satu perbandingan dengan hasil subquery menghasilkan nilai TRUE. Ilustrasinya jika:

Gaji > ANY (S)

Jika subquery S menghasilkan G1, G2, ..., Gn, maka kondisi di atas identik dengan:

(gaji > G1) OR (gaji > G2) OR ... OR (gaji > Gn)

Contoh: perintah untuk menampilkan semua data jenisfilm yang harganya bukan yang terkecil:

```
SELECT * FROM JENISFILM WHERE HARGA > ANY (SELECT HARGA FROM
JENISFILM);
```

Operator ALL digunakan untuk melakukan perbandingan dengan subquery. Kondisi dengan ALL menghasilkan nilai TRUE (benar) jika subquery tidak menghasilkan apapun atau jika perbandingan menghasilkan TRUE untuk setiap nilai query terhadap hasil subquery.

Contoh : perintah untuk menampilkan data jenisfilm yang harganya paling tinggi:

```
SELECT * FROM JENISFILM WHERE HARGA >= ALL (SELECT HARGA FROM
JENISFILM);
```

f. Sintak ORDER BY

Klausa ORDER BY digunakan untuk mengurutkan data berdasarkan kolom tertentu sesuai dengan tipe data yang dimiliki. Contoh : perintah untuk mengurutkan data film berdasarkan kolom judul:

```
SELECT * FROM FILM ORDER BY JUDUL;
```

atau tambahkan ASC untuk pengurutan secara ascending (menaik) :

```
SELECT * FROM FILM ORDER BY JUDUL ASC;
```

atau tambahkan DESC untuk pengurutan secara descending (menurun):

```
SELECT * FROM FILM ORDER BY JUDUL DESC;
```

g. Sintak DISTINCT

Distinct adalah kata kunci ini untuk menghilangan duplikasi. Sebagai Contoh, buat sebuah tabel pelanggan yang berisi nama dan kota asal dengan beberapa record isi dan beberapa kota asal yang sama. Kemudian ketikkan perintah berikut:

```
SELECT DISTINCT KOTA FROM PELANGGAN;
```

Dengan perintah di atas maka nama kota yang sama hanya akan ditampilkan satu saja.

h. UNION, INTERSECT dan EXCEPT

UNION merupakan operator yang digunakan untuk menggabungkan hasil query, dengan ketentuan jumlah, nama dan tipe kolom dari masing-masing tabel yang akan ditampilkan datanya harus sama. Berikut ini perintah untuk memperoleh data pada tabel film dimana jenisnya action dan horor:

```
SELECT JENIS, JUDUL FROM FILM WHERE JENIS = 'ACTION' UNION SELECT
JENIS, JUDUL FROM FILM WHERE JENIS = 'HOROR';
```

Perintah di atas identik dengan:

```
SELECT JENIS, JUDUL FROM FILM WHERE JENIS = 'ACTION' OR JENIS =
'HOROR';
```

Namun tidak semua penggabungan dapat dilakukan dengan OR, yaitu jika bekerja pada dua tabel atau lebih.

INTERSECT merupakan operator yang digunakan untuk memperoleh data dari dua buah

query dimana data yang ditampilkan adalah yang memenuhi kedua query tersebut dengan ketentuan jumlah, nama dan tipe kolom dari masing-masing tabel yang akan ditampilkan datanya harus sama.

SELECT * FROM namatabel1 INTERSECT SELECT * FROM namatabel2;

Pada MySQL tidak terdapat operator INTERSECT namun sebagai gantinya dapat menggunakan operator IN seperti contoh 1 pada bagian Nested Queries.

EXCEPT / Set Difference merupakan operator yang digunakan untuk memperoleh data dari dua buah query dimana data yang ditampilkan adalah data yang ada pada hasil query 1 dan tidak terdapat pada data dari hasil query 2 dengan ketentuan jumlah, nama dan tipe kolom dari masing-masing tabel yang akan ditampilkan datanya harus sama.

SELECT * FROM namatabel1 EXCEPT SELECT * FROM namatabel2;

BAB VI

FUNGSI AGREGAT

Aggregate Functions (COUNT, SUM, AVG, MIN, MAX)

a. C OU N T

Perintah yang digunakan untuk menghitung jumlah baris suatu kolom pada tabel.

Contoh : Perintah untuk menghitung jumlah baris kolom jenis pada tabeljenisfilm:

```
SELECT COUNT(namafield) FROM nama_tabel;
```

b . SUM

Perintah yang digunakan untuk menghitung jumlah nilai suatu kolom pada tabel.

Contoh : perintah untuk menghitung jumlah nilai kolom harga pada tabel jenisfilm :

```
SELECT SUM(namafield) FROM nama_tabel;
```

c . AVG

Perintah yang digunakan untuk menghitung rata- rata dari ni lai suatu kolom pada tabel. Contoh : perintah untuk menghitung rata-rata dari kolom harga pada tabel jenisfilm:

```
SELECT AVG(namafield) FROM nama_tabel;
```

d . MIN

Perintah yang digunakan untuk menampilkan nilai terkeci l dari suatu kolom pada tabel. Contoh : perintah untuk menampilkan nilai terkecil dari kolom harga pada tabel jenisfilm:

```
SELECT MIN(namafield) FROM nama_tabel;
```

e . MAX

Perintah yang digunakan untuk menampilkan nilai terbesar dari suatu kolom pada table. Contoh : perintah untuk menampi lkan nilai terbesar dari kolom harga pada table jenisfilm :

SELECT MAX(namafield) FROM nama_tabel;

RETRIEVE SQL dengan GROUP BY dan HAVING

Klausa GROUP BY digunakan untuk melakukan pengelompokkan data. Syntax :

SELECT field1,SUM(field2) FROM namatable GROUP BY field1;

Sebagai contoh, terdapat table barang dengan data sebagai berikut :

kode_barang	nama_barang	satuan_barang	stok_barang	harga_barang
B1	HARDISK	BUAH	12	500000
B2	MP3 PLAYER	UNIT	30	200000
B3	TAPE	UNIT	25	350000
B4	FLASHDISK	BUAH	12	100000

Klausa HAVING digunakan untuk menentukan kondisi bagi klausa GROUP BY. Kelompok yang memenuhi HAVING saja yang akan dihasilkan. Syntax :

SELECT field1 FROM namatabel GROUP BY field1 HAVING COUNT(field2);

PATTERN MATCHING (Pencocokan Pola / Karakter)

Fungsi string digunakan untuk menampilkan data yang di dasarkan pada pencarian dengan karakter. Pada pencarian data digunakan syntax LIKE, pada dasarnya syntax LIKE hampir sama dengan syntax = .

Bedanya kalau syntax = , maka pencarian karakter harus sesuai dengan kata yang kita buat tetapi dengan menggunakan LIKE karakter yang akan kita tampilkan tidak harus lengkap hanya dengan menuliskan salah satu huruf atau kata saja, maka semua data yang akan kita cari akan ditampilkan.

SQL mempunyai dua symbol khusus yang dipakai untuk pencocokan pola :

1. % : digunakan untuk mencocokkan karakter sebelum atau sesudah tanda %;

2. _ : digunakan untuk mencari karakter sebanyak jumlah tanda _.

Contoh:

LIKE '%GLASGOW%' artinya digunakan untuk mencari data pada kolom tertentu yang megandung karakter „GLASGOW".

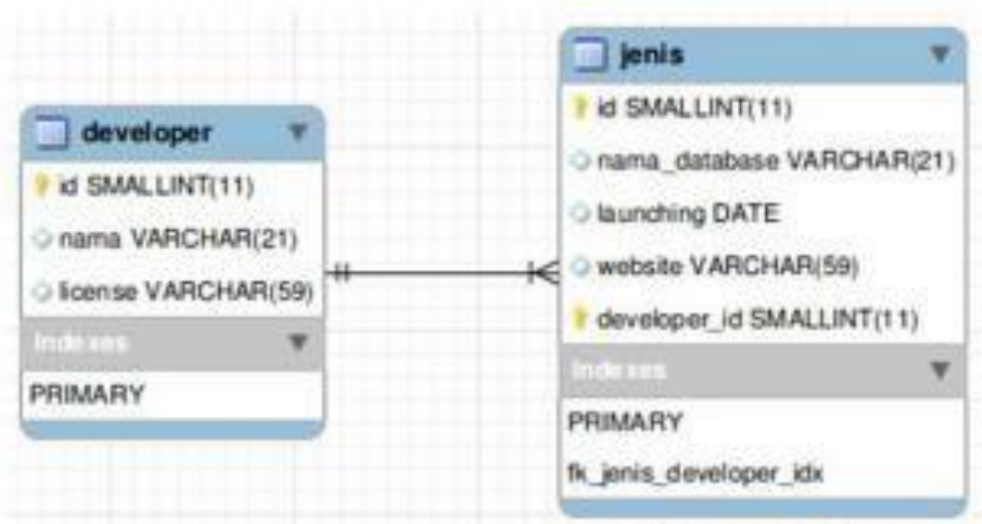
Bentuk umumnya :

```
SELECT * FROM nama_tabel WHERE nama_kolom LIKE 'char%';

SELECT * FROM nama_tabel WHERE nama_kolom LIKE '%char';

SELECT * FROM nama_tabel WHERE nama_kolom LIKE '%char%';

SELECT * FROM nama_tabel WHERE nama_kolom NOT LIKE '%char%';

SELECT * FROM nama_tabel WHERE nama_kolom LIKE '_';
```

BAB VII

RELASI DATABASE

Relasi tabel merupakan hubungan yang terjadi pada suatu tabel dengan lainnya yang mempresentasikan hubungan antar objek di dunia nyata dan berfungsi untuk mengatur mengatur operasi suatu database.



Ada 3 macam relasi tabel, diantaranya :

[1] One-To-One (1-1)

Mempunyai pengertian "Setiap baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel ke dua". Contohnya : relasi antara tabel mahasiswa dan tabel orang tua. Satu baris mahasiswa hanya berhubungan dengan satu baris orang tua begitu juga sebaliknya.

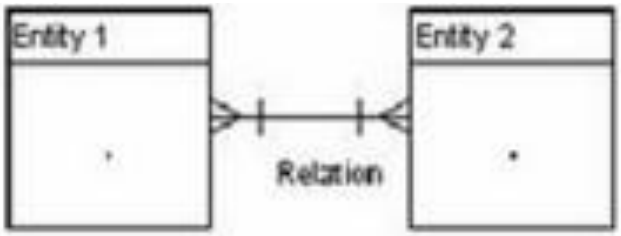
[2] One-To-Many (1-N)

Mempunyai pengertian "Setiap baris data dari tabel pertama dapat dihubungkan ke satu

baris atau lebih data pada tabel ke dua". Contohnya : relasi perwalian antara tabel dosen dan tabel mahasiswa. Satu baris dosen atau satu dosen bisa berhubungan dengan satu baris atau lebih mahasiswa.

[3] Many-To-Many (N-M)

Mempunyai pengertian "Satu baris atau lebih data pada tabel pertama bisa dihubungkan ke satu atau lebih baris data pada tabel ke dua". Artinya ada banyak baris di tabel satu dan tabel dua yang saling berhubungan satu sama lain. Contohnya : relasi antar tabel mahasiswa dan tabel mata kuliah. Satu baris mahasiswa bisa berhubungan dengan banyak baris mata kuliah begitu juga sebaliknya.



Latihan !

1. Buatlah database baru dengan nama library.
2. Selanjutnya buatlah tabel **category** dan tabel **book** dengan struktur sebagai berikut :

Tabel **category**

Field	Type	Null	Key	Default	Extra
category_id	int(10)	NO	PRI	NULL	auto_increment
category_name	varchar(30)	NO		NULL	

Tabel **book**

Field	Type	Null	Key	Default	Extra
book_id	int(10)	NO	PRI	NULL	
title	varchar(30)	NO		NULL	
author	varchar(30)	NO		NULL	
publisher	varchar(30)	NO		NULL	

Jangan lupa memastikan bahwa storage engine tabel yang dibuat bertipe INNODB karena hanya tipe ini yang mendukung relasi dan pembuatan foreign key

3. Setelah kita analisa, relasi antara tabel category dengan tabel book adalah 1:N dimana satu category buku bisa terdiri dari banyak buku. Karena relasi 1:N maka atribut kunci pada tabel category (category_id) bertamu ke tabel book sehingga ada penambahan atribut pada tabel book (category_id). Berikut struktur baru tabel book :

Field	Type	Null	Key	Default	Extra
book_id	int(10)	NO	PRI	NULL	
title	varchar(30)	NO		NULL	
author	varchar(30)	NO		NULL	
publisher	varchar(30)	NO		NULL	
category_id	int(10)	NO		NULL	

```
mysql> alter table book
    -> add foreign key(category_id)
    -> references category(category_id);
```

4. Langkah selanjutnya adalah menambahkan foreign key pada tabel book sebagai berikut :

   Sehingga tampilan baru tabel book sebagai berikut :

```
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| book_id     | int(10)     | NO   | PRI | NULL    |       |
| title       | varchar(30) | NO   |     | NULL    |       |
| author      | varchar(30) | NO   |     | NULL    |       |
| publisher   | varchar(30) | NO   |     | NULL    |       |
| category_id | int(10)     | NO   | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
```

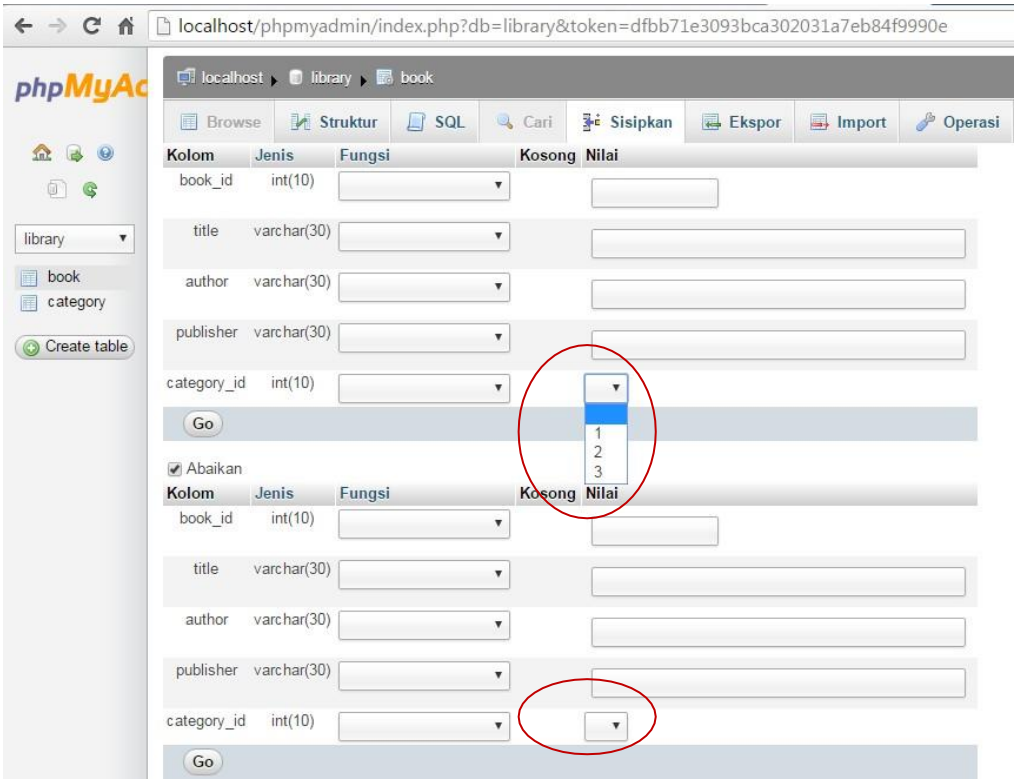
```
+-------------+-----------------+
| category_id | category_name   |
+-------------+-----------------+
|           1 | Komputer        |
|           2 | Telekomunikasi  |
|           3 | Jaringan        |
+-------------+-----------------+
```

5. Ok, sekarang lanjutkan dengan mengisi dulu tabel category sebagai berikut :

6. Sekarang kita akan cek apakah relasi 1:N yang kita buat telah berhasil apa tidak. Dalam hal ini kita akan menggunakan GUI XAMPP (phpmyadmin) untuk memudahkan :

   Cara 1 :

   Jika relasi berhasil, maka ketika menu insert dipilih pada tabel book maka kolom category_id yang berasal dari tabel category akan otomatis menggunakan menu combo box seperti berikut :


localhost/phpmyadmin/index.php?db=library&token=dfbb71e3093bca302031a7eb84f9990e
phpMyAd
localhost ▸ library ▸ book
Browse
Struktur
SQL
Cari
Sisipkan
Ekspor
Import
Operasi
library
book
category
Create table
Kolom Jenis Fungsi Kosong Nilai
book_id int(10)
title varchar(30)
author varchar(30)
publisher varchar(30)
category_id int(10)
1
2
3
Go
Abaikan
Kolom Jenis Fungsi Kosong Nilai
book_id int(10)
title varchar(30)
author varchar(30)
publisher varchar(30)
category_id int(10)
Go

Cara 2 :

Klik database library, kemudian pada menu operasi pilih desainer (perancang). Jika

relasi sukses dan benar maka akan menampilkan ERD secara physic sebagai berikut :

Catatan Penting :

Jika saat membuat relasi (menambahkan FK berhasil) tetapi saat dicek hasilnya tidak seperti diatas maka hal yang paling awal harus dilakukan adalah dengan mengecek storage engine dari tabel-tabel yang berelasi. Pastikan mesin penyimpanannya menggunakan INNODB, karena biasanya default ketika membuat tabel yaitu bertipe MYISAM. Jika ternyata bertipe MYISAM, harus segera diganti kemudian filed tamu juga dihapus dan dibuat ulang untuk menghilangkan pembacaan perintah ketika masih bertipe MYISAM.

book_id	title	author	publisher	category_id
131181	24 Jam Belajar PHP	Edy Winarni ST, M.Eng, Ali	Elex Media Komputindo	1
131182	Sistem Telekomunikasi di Indo	Gauzali Saydam	Alfabeta	2
131183	Pengantar Jaringan Komputer d	Iwan Sofana	Informatika	3

7. Baik, selanjutnya masukkan data ke tabel book sebagai berikut :
8. Tambahkan data pada tabel book sehingga datanya menjadi sebagai berikut :

BAB VIII

QUERY DAN VIEW

❖ Query

Query adalah pernyataan yang meminta pengguna mengambil informasi. Bagian DML yang terlibat dalam pengambilan informasi disebut bahasa query. Istilah bahasa query sering disamakan dengan istilah bahasa manipulasi data. Sedangkan SQL adalah sebuah sintaks untuk mengeksekusi query.

❖ View

Merupakan salah satu objek database, yang secara logika merepresentasikan sub himpunan dari data yang berasal dari satu atau lebih tabel. Kegunaan view adalah untuk membatasi akses database, membuat query kompleks secara mudah, mengijinkan independensi data dan untuk menampilkan view (pandangan) data yang berbeda dari data yang sama.

❖ Tujuan dari view adalah:

a. Menurunkan Network Traffic (beban Network).
b. Menyimpan suatu perintah SQL (terutama yang komplek) dimana perintah tersebut sering digunakan dan diakses.
c. Mencegah user untuk dapat mengakses suatu tabel sepenuhnya.
d. Misal user dapat mengakses nama dan nomor telepon tetapi tidak bisa mengakses tanggal lahir dan gaji.

❖ Sintak dari View adalah :

CREATE [OR REPLACE] [<algorithm attributes>]

VIEW [database.]< name> [(<columns>)]

AS <SELECT statement> [<check options>];

Contoh:

a) Membuat View:

```
CREATE VIEW pelanggan_simpati AS
SELECT nama,alamat,tgl_lahir,telepon
FROM pelanggan WHERE telepon REGEXP '^081[23]'
ORDER BY nama;
```

b) Cara mengaksesnya:

```
SELECT * FROM pelanggan_simpati;
SELECT nama,alamat FROM pelanggan_simpati;
```

View termasuk dalam komponen database. Secara default, suatu view baru dibuat ke dalam database yang diaktifkan. Untuk membuat secara eksplisit di dalam suatu database tertentu, maka buatlah nama view dengan format: db_name.view_name.

Contoh lain yang akan diberikan adalah view untuk menyimpan informasi detail mahasiswa, dalam hal ini melibatkan 2 tabel, yaitu mahasiswa dan prodi.

JOIN

- **Join**

Operasi Join, Join merupakan operasi yang digunakan untuk menggabungkan

```
mysql> select judul, harga from film,jenisfilm
    -> where jenisfilm.jenis=film.jenis;
+---------------+--------+
| judul         | harga  |
+---------------+--------+
| Spiderman 1   |   2500 |
| Spiderman 2   |   2500 |
| Spiderman 3   |   2500 |
| Love Story    |   1000 |
| Suster Ngesot |   4000 |
| Evil Death    |   4000 |
+---------------+--------+
6 rows in set (0.39 sec)
```

dua tabel atau lebih dengan hasil berupa gabungan dari kolom-kolom yang berasal dari tabel-tabel tersebut. Pada join sederhana, tabel-tabel digabungkan dan didasarkan pada pencocokan antara kolom pada tabel yang berbeda.

Pada contoh di atas , jenisfilm.jenis=film.jenis merupakan kondisi untuk menco cokkan data antara kolom jenis milik tabel jenisfilm dan film.

a. Inner Join

Inner join digunakan untuk menampilkan data dari dua tabel yang berisi data sesuai dengan syarat dibelakang on (tidak boleh null), dengan kata lain semua data dari tabel kiri mendapat pasangan data dari tabel sebelah kanan . Berikut ini perintah untuk menampilkan data dari tabel jenisfilm dan film dengan syarat berdasarkan kolom jenis:

SELECT * FROM JENISFILM INNER JOIN FILM ON (JENISFILM.JENIS = FILM.JENIS);

b. Left Join

Left join digunakan untuk menampilkan semua data dari tabel sebelah kiri per intah left join beserta pasangannya dari tabel sebelah kanan. Meskipun terdapat data dari sebelah kiri tidak memiliki pasangan, tetap akan ditampilkan dengan pasangannya berupa

nilai NULL.

```
SELECT * FROM JENISFILM LEFT JOIN FILM ON (JENISFILM.JENIS =
FILM.JENIS);
```

c. Right Join

Right join digunakan untuk menampilkan semua data dari tabel sebelah kanan perintah right join beserta pasangannya dari tabel sebelah kiri. Meskipun terdapat data dari sebelah kanan tidak memiliki pasangan , tetap akan ditampilkan dengan pasangannya berupa nilai NULL.

```
SELECT * FROM JENISFILM RIGHT JOIN FILM ON (JENISFILM.JENIS =
FILM.JENIS);
```

d. Natural Join

Natural join digunakan untuk menampilkan semua data dari dua tabel dimana jika terdapat kolom yang sama, maka yang akan ditampilkan hanya salah satunya saja, yaitu kolom dari tabel sebelah kiri perintah natural join.

```
SELECT * FROM JENISFILM NATURAL JOIN FILM;
```

Terdapat Penggabungan Natural Join dengan Left dan Right Join:

1. Natural Left Join

Natural left join digunakan untuk menampilkan semua data dari tabe l sebelah kiri perintah natural left join beserta pasangannya dari tabel sebelah kanan. Meskipun terdapat data dari sebelah kiri tidak memiliki pasangan, tetap akan ditampilkan dengan pasangannya berupa nilai NULL.

```
SELECT * FROM JENISFILM NATURAL LEFT JOIN FILM;
```

2. Natural Right Join

Natural right join digunakan untuk menampilkan semua data dari tabel sebelah kanan perintah natural right join beserta pasangannya dari tabel sebelah kiri. Meskipun terdapat data dari sebelah kanan tidak memiliki pasangan, tetap akan ditampilkan dengan pasangannya berupa nilai NULL.

```
SELECT * FROM JENISFILM NATURAL RIGHT JOIN FILM;
```

DAFTAR PUSTAKA

[1] Adi Nugroho, ST., MMSI, 2004, Konsep Perancangan Sistem Basis Data, Andi Offset, Yogyakarta

[2] Bambang Hariyanto, Ir. MT., 2004, Sistem Manajemen Basis Data, Informatika, Bandung

[3] Didik Dwi Prasetyo, 2003, Administrasi Database Server MySQL, Flex Media Komputindo, Jakarta

[4] Fathansyah Ir, 2002, Basis Data, Informatika, Bandung

[5] Jogiyanto H.M., 2002, Analisis Dan Desain Sistem Informasi, Andi Offset, Yogyakarta

[6] Kadir Abdul, 2008, Belajar database menggunakan MySQL, Andi Offset, Yogyakarta