SMK Negeri 1 Sumbawa

Basis Data
Lembar Kerja Peserta Didik (LKPD)
Tahun Ajaran 2022 / 2023

03

Kelas : XI - RPL / XI SIJA
Pengampu : Ahmad Mukhlisyaini

Kelompok Perintah Dalam Basis Data (DDL dan DML)

MySQL

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

Cerita tentang MySQL bermula pada tahun 1979, ketika Michael Widenius (a.k.a. Monty), seorang programmer asal Swedia, mengembangkan sebuah sistem database sederhana yang dinamakan UNIREG. UNIREG ini menggunakan koneksi low-level ISAM database engine dengan indexing (baca : sistem database sederhana yang mendukung index).

Monty bekerja pada perusahaan bernama TcX di Swedia. TcX pada tahun 1994 mulai mengembangkan aplikasi berbasis web, dan berencana menggunakan UNIREG sebagai sistem databasenya. Namun malang bagi Monty, UNIREG dianggagap tidak cocok untuk database yang dinamis seperti web.

TcX mencoba mencari alternatif sistem database lainnya, salah satunya adalah mSQL (miniSQL), sebuah RDBMS yang tidak terlalu mahal dan hampir open source, maksudnya jika anda membeli aplikasi ini, anda juga akan mendapatkan source code nya juga. Namun mSQL versi 1 ini juga memiliki kekurangan, yaitu tidak mendukung indexing, sehingga performanya tidak terlalu bagus.

Dengan hasutan petinggi-petinggi TcX, Monty mencoba menghubungi David Hughes (programmer yang mengembangkan mSQL) untuk menanyakan apakah ia tertarik mengembangkan sebuah konektor di mSQL yang dapat dihubungkan dengan UNIREG ISAM sehingga mendukung indexing. Pada saat itu Hughes menolak, dengan alasan sedang mengembangkan teknologi indexing yang independen untuk mSQL versi 2.

Dikarenakan penolakan David Hughes, TcX (dan juga Monty) akhirnya memutuskan untuk merancang dan mengembangkan sendiri konsep sistem database baru. Sistem ini merupakan gabungan dari UNIREG dan mSQL (source codenya dapat bebas digunakan). Sehingga pada May 1995, sebuah RDBMS baru, yang dinamakan MySQL dirilis.

David Axmark dari Detron HB, rekanan TcX mengusulkan agar MySQL di 'jual' dengan model bisnis baru. Ia mengusulkan agar MySQL dikembangkan dan dirilis dengan gratis. Pendapatan perusahaan selanjutnya di dapat dari menjual jasa "support" untuk perusahaan yang ingin mengimplementasikan MySQL. Konsep bisnis yang juga diterapkan perusahaan-perusahaan Open Source lainnya.

MySQL dan MySQL AB

Kembali ke kisah MySQL, Pada tahun 1995 itu juga, TcX berubah nama menjadi MySQL AB, dengan Michael Widenius, David Axmark dan Allan Larsson sebagai pendirinya. Titel "AB" dibelakang MySQL, adalah singkatan dari "Aktiebolag", istilah PT (Perseroan Terbatas) bagi perusahaan Swedia.

MySQL AB menjadi perusahaan di belakang MySQL, menyediakan jasa dan bertanggung jawab dalam mengembangkan, memasarkan, dan menyediakan dukungan untuk MySQL. MySQL sendiri dirilis dengan "dual licencing", atau dua lisensi yakni versi gratis dan versi berbayar.

Lisensi pertama di rilis dibawah GNU GPL (General Public License – atau dikenal juga dengan Hak Pakai Lisensi). Lisensi GPL ini membebaskan anda menggunakan MySQL tanpa membayar royalti kepada MySQL AB, dengan beberapa syarat tertentu. Misalnya, jika anda menggunakan MySQL dalam aplikasi yang anda buat, aplikasi tersebut juga harus bersifat gratis dan berada di bawah lisensi GPL.

Lisensi kedua di peruntukkan bagi perusahaan-perusahaan komersil, maupun pengembang software yang berniat menjual aplikasinya, dan menggunakan MySQL sebagai databasenya. Untuk keperluan ini, anda diharuskan membeli lisensi komersial dari MySQL AB. Lebih lanjut tentang permasalahan seputar lisensi MySQL, dapat mengunjungi situs MySQL.

MySQL AB juga memegang hak copyright dari source code MySQL dan pemilik hak merk dagang "MySQL". Dengan kata lain, walaupun kita memiliki source code MySQL, namun sistem database maupun aplikasi yang kita buat tidak boleh menggunakan merk "MySQL" tanpa membayar royalti kepada pihak MySQL AB. Hal ini pula yang menjelaskan mengapa salah satu aplikasi administasi MySQL berbasis web PhpMyAdmin, tidak menggunakan kata "MySQL" pada nama programnya.

MariaDB

MariaDB merupakan versi pengembangan terbuka dan mandiri dari MySQL. Sejak diakuisisinya MySQL oleh Oracle pada September 2010, Monty sebagai penulis awal kode sumber MySQL memisahkan diri dari pengembangan dan membuat versi yang lebih mandiri yakni MariaDB.

Semua kemampuan MySQL dimiliki pula oleh MariaDB yakni:

1. Portabilitas. MariaDB dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
2. Perangkat lunak sumber terbuka. MariaDB didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
3. Multi-user. MariaDB dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. 'Performance tuning', MariaDB memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. Ragam tipe data. MariaDB memiliki ragam tipe data yang sangat kaya, seperti signed / unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.
6. Perintah dan Fungsi. MariaDB memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (query).
7. Keamanan. MariaDB memiliki beberapa lapisan keamanan seperti level subnetmask,

nama host, dan izin akses user dengan sistem perizinan yang mendetail serta sandi terenkripsi.

8. Skalabilitas dan Pembatasan. MariaDB mampu menangani basis data dalam skala besar, dengan jumlah rekaman (records) lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. Konektivitas. MariaDB dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).
10. Pelokalan Bahasa. MariaDB dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. Antar Muka. MariaDB memiliki antar muka (interface) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).
12. Klien dan Peralatan. MariaDB dilengkapi dengan berbagai peralatan (tool) yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.
13. Struktur tabel. MariaDB memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

Format Perintah MySQL

Berikut adalah ketentuan-ketentuan memberi perintah pada MySQL:

- Setiap perintah harus diakhiri dengan tanda titik koma , kecuali untuk perintah tertentu,
  misal : quit
- 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.
- Perintah-perintah dalam lingkungan MySQL tidak menerapkan aturan case sensitive, tetapi case insensitive yaitu perintah bisa dituliskan dalam huruf besar atau pun huruf
- kecil.
- Aturan case sensitive diterapkan pada penamaan objek-objek dalam database seperti nama database atau nama table, namun aturan ini hanya ada dalam lingkungan Unix dan Linux.

Tipe Data

Sebagaimana fokus database dalam mengolah data, DBMS seperti MySQL mengelompokkan data-data berdasarkan tipe tertentu, yaitu:

KARAKTER

- CHAR: String dengan maksimal 255 (1-255) karakter, bersifat tetap
- VARCHAR: String maksimal 255 karakter dan bersifat variabel (berubah sesuai isi)
- TEXT: Teks dengan panjang maksimal 65535

BILANGAN

- TINYINT: Bilangan 1 byte (8 bit), jangkauan: -128 s/d 127
- SMALLINT: Bilangan 2 byte (16 bit), jangkauan: -32.768 s/d 32.767
- MEDIUMINT: Bilangan 3 byte (24 bit), jangkauan: -8.388.608 s/d 8.388.607
- INT atau INTEGER: Bilangan 4 byte, jangkauan: -2.147.483.648 s/d 2.147.483.647
- BIGINT: Bilangan 8 byte (64 bit), jangkauan: ± 9,22 x 1018
- FLOAT: Bilangan pecahan 4 byte (32 bit), jangkauan: -3.402823466E+38 s/d -1.175494351E-38, 0, dan 1.175494351E-38 s/d 3.402823466E+38.
- DOUBLE atau REAL: Bilangan pecahan 8 byte (64 bit), jangkauan: -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308
- DECIMAL(M, D) atau NUMERIC(M, D): Bilangan pecahan 8 byte (64 bit), jangakauan: -1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308

LAIN-LAIN

- DATE: Tanggal, format: YYYY-MM-DD
- DATETIME: Waktu (tanggal dan jam), format: YYYY-MM-DD HH:MM:SS
- TIME: Jam, format: HH:MM:SS
- YEAR: untuk menyimpan tahun, format: YYYY, jangkauan: 1900 s/d 2155
- ENUM('nilai1', 'nilai2', …): Nilai enumerasi (kumpulan data), jangkauan: sampai dengan 65535 string
- BOOLEAN: tipe benar atau salah

Contoh penggunaan:
sintax dasar : namadata **TIPEDATA**;
Contoh: id_siswa INT, uang_setoran DOUBLE, username VARCHAR(50), deskripsi TEXT, tanggal_daftar DATE, password CHAR(100).

DDL (Data Definition Language)

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

Contoh penerapannya dapat dilihat pada tabel berikut:

Perintah Dasar	Kegunaan	Contoh implementasi
Create	Membuat database, tabel, function, procedure, index	Membuat database: CREATE DATABASE namadatabase; Catatan: namadatabase bebas diganti sesuai kebutuhan membuat tabel: CREATE TABLE table_name (column1 datatype, column2 datatype, column3 datatype, ...

Perintah Dasar	Kegunaan	Contoh implementasi
		);
Alter	Untuk memodifikasi struktur table	ALTER TABLE table_name ADD column_name datatype; ALTER TABLE table_name MODIFY COLUMN column_name datatype; ALTER TABLE table_name MODIFY column_name datatype; Contoh: ALTER TABLE Customers ADD Email varchar(255); Menambahkan filed email pada tabel Customer
Drop	Untuk menghapus database atau table	DROP DATABASE databasename; DROP TABLE tablename; Contoh: DROP TABLE Customer; untuk menghapus tabel Customer

Penggunaan DDL

Membuat database

Syntax Membuat Database : **CREATE DATABASE namadatabase;**

Namadatabase tidak boleh mengandung spasi dan tidak boleh memiliki nama yang sama antar database. Berikut ini perintah untuk membuat database dengan nama CV_SEJAHTERA : CREATE DATABASE CV_SEJAHTERA;

```
mysql> create database CV_SEJAHTERA;
Query OK, 1 row affected (0.01 sec)
```

Syntax tambahan untuk menampilkan daftar nama database yang ada pada mysql menggunakan perintah : SHOW DATABASES;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| akademik           |
| akademik_t         |
| belajar_ajah       |
| belajarif          |
| cdcol              |
| contohlooping      |
| cv_sejahtera       |
| data_mahasiswa     |
| dataku             |
| dataku2            |
| db_akademik        |
```

Memilih Database

Memilih Database : **USE namadatabase;**

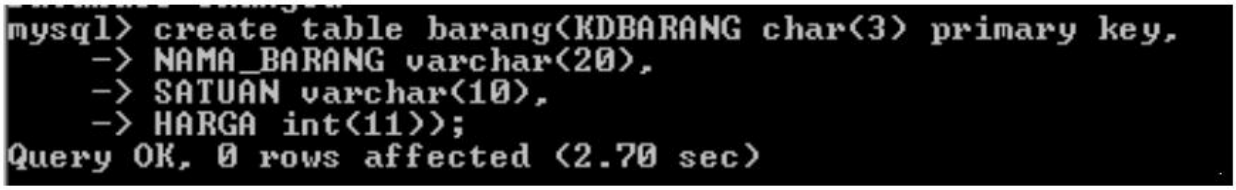
Sebelum membuat suatu tabel, terlebih dahulu harus memilih salah satu database sebagai database aktif yang akan digunakan untuk menyimpan tabel-tabel, Berikut ini perintah untuk menggunakan database dengan nama CV_SEJAHTERA:
USE CVSEJAHTERA;

Menghapus Database
Syntax Menghapus Database : DROP DATABASE namadatabase;

Database yang akan dihapus sesuai dengan namadatabase. Berikut ini perintah untuk menghapus database dengan nama database:
DROP CV_SEJAHTERA;

Membuat Tabel
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 (,).
Berikut ini perintah untuk membuat tabel dengan nama barang :



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

Menampilkan Atribut Tabel

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

```
mysql> desc barang;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| KDBARANG    | char(3)     | NO   | PRI |         |       |
| NAMA_BARANG | varchar(20) | YES  |     | NULL    |       |
| SATUAN      | varchar(10) | YES  |     | NULL    |       |
| HARGA       | int(11)     | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
4 rows in set (0.05 sec)
```

Menghapus Table
Syntax Menghapus Tabel : DROP TABLE namatabel;

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

Not Null
Mendefinisikan Null/Not Null : CREATE TABLE namatabel (Field1 TipeData1 **NOT NULL**, Field2 TipeData2);
Tujuannya agar user wajib mengisi field yang dibuat atau data tidak boleh kosong.

Primary Key
Mendefinisikan Primary Key (kode unik) 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
Buat table terlebih dahulu :
CREATE TABLE namatabel (Field1 TipeData1, Field2 TipeData2);
Baru kemudian ditambahkan Primary Key
ALTER TABLE namatabel ADD CONSTRAINT namaconstraint PRIMARY KEY (Field1);
atau
ALTER TABLE namatabel ADD PRIMARY KEY (Field1);
Catatan:
namaconstraint : Boleh memakai nama apa saja, karena seperti variabel.

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;

Menambah Kolom Baru Pada Tabel
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;

Mengubah Tipe Data atau Lebar Kolom Pada Tabel
Menggunakan sintax
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);

Mengubah Nama Kolom
Menggunakan :
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);

Menghapus Kolom Pada Tabel

Menggunakan :
ALTER TABLE namatabel DROP COLUMN namakolom;

Tugas Praktikum DDL

Buatlah database dengan format : Penjualan+namaKalian.
Contoh: Penjualan_Syukron atau penjualanSyukron - ingat, ini case sensitive, artinya huruf besar atau kecil akan sangat berpengaruh.

Isi database kalian dengan 5 buah tabel, deskripsi tabel sesuai selera pembuat. Rekomendasi tabel:

1. Penjual/ Kasir
2. Pembeli/ Customer
3. Barang
4. Transaksi
5. Stok

Setiap tabel berisi paling sedikit 4 field.

DML (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**.

INSERT

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);

```
mysql> insert into barang values('B01','BUKU TULIS','LUSIN',50000),
    -> ('B02','PULPEN','LUSIN',80000),
    -> ('B03','PENGHAPUS','LUSIN',20000);
Query OK, 3 rows affected (2.61 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

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

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

```
mysql> insert into barang (KDBARANG,NAMA_BARANG,SATUAN,HARGA)
    -> values('B04','PERAUT','LUSIN',45000);
Query OK, 1 row affected (0.00 sec)
```

Keterangan :

Jika data bertipe string (char, varchar, text), 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.

Cara 3: **INSERT INTO nama_tabel SET field1='nilai1', field2='nilai2',...;**

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.

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];**

```
mysql> update barang set NAMA_BARANG='PERAUT'
    -> where KDBARANG='B03';
Query OK, 1 row affected (2.65 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

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

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;

```
mysql> select*from barang;
+----------+--------------+--------+-------+
| KDBARANG | NAMA_BARANG  | SATUAN | HARGA |
+----------+--------------+--------+-------+
| B01      | BUKU TULIS   | LUSIN  | 50000 |
| B02      | PULPEN       | LUSIN  | 80000 |
| B03      | PERAUT       | LUSIN  | 20000 |
| B04      | PERAUT       | LUSIN  | 45000 |
+----------+--------------+--------+-------+
4 rows in set (0.00 sec)
```

b. Menampilkan data untuk kolom tertentu :
SELECT kolom1,kolom2,kolom-n FROM namatabel;

```
mysql> select KDBARANG,NAMA_BARANG from barang;
+----------+--------------+
| KDBARANG | NAMA_BARANG  |
+----------+--------------+
| B01      | BUKU TULIS   |
| B02      | PULPEN       |
| B03      | PERAUT       |
| B04      | PERAUT       |
+----------+--------------+
4 rows in set (0.00 sec)
```

c. Menampilkan data dengan kondisi data tertentu dengan klausa WHERE:
SELECT * FROM namatabel WHERE kondisi;

```
mysql> select *from barang where KDBARANG='B02';
+----------+--------------+--------+-------+
| KDBARANG | NAMA_BARANG  | SATUAN | HARGA |
+----------+--------------+--------+-------+
| B02      | PULPEN       | LUSIN  | 80000 |
+----------+--------------+--------+-------+
1 row in set (0.00 sec)
```

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**.

d. 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;

e. 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;

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

g. 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 * FROM film disebut subquery, sedangkan :
SELECT * FROM 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);

h. 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);
```

i. 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;
```

j. 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.

k. 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;
Pada MySQL tidak terdapat operator EXCEPT namun sebagai gantinya dapat menggunakan operator NOT IN seperti contoh 2 pada bagian Nested Queries.

Tugas Praktikum DML

1. Buatlah Database dengan nama hotel+namapanggilan, Contoh: hotel_Swingli
2. Buatlah tabel dengan nama Penginap, dengan data sbb:

Field Name	Data type	Field Size	Keterangan
KdPenginap	Varchar	9	Kode Penginap
NamaPenginap	Varchar	50	Nama Penginap
AlamatPenginap	Varchar	50	Alamat Penginap
Usia	Integer	3	Usia Penginap
TlpPenginap	Varchar	15	Telephon Penginap

3. Tampilkan deskripsi dari table Penginap yang sudah pernah dibuat.
4. Tambahkan kolom baru bernama Status dengan posisi setelah kolom/field Usia dengan tipe ENUM dengan keterangan Menikah atau Belum Menikah pada tabel Penginap
5. Rubahlah kolom TlpPenginap menjadi TeleponPenginap dengan typedata sama pada table Penginap tersebut
6. Masukkan 10 data kedalam tabel Penginap
7. Rubahlah nama tabel Penginap dengan DataPenginap
8. Tampikan data Kode, Nama, dan Usia pada tabel Penginap
9. Ubah salah satu AlamatPenginap pada tabel Penginap dengan perintah UPDATE
10. Hapus salah satu data penginap dengan perintah DELETE

.....Selamat mencoba.....

Daftar Pustaka:
https://www.w3schools.com/sql

“Yesterday is history, tomorrow is a mystery, but **today is a gift**. That’s why it is called **the present**”

- Master Oogway, Kung Fu Panda -

“Yesterday is history, tomorrow is a mystery, **today is a gift of God**, which is why we call it **the present**.”

— Bill Keane