

SISTEM BASIS DATA MYSQL

PADA KONSEP JARINGAN KLIEN SERVER

ABDUL ROZAQ



2019



Diterbitkan Atas Kerjasama
Deepublish dengan Politeknik Banjarmasin



**SISTEM BASIS DATA MYSQL
PADA KONSEP
JARINGAN KLIEN SERVER**

UU No 28 tahun 2014 tentang Hak Cipta

Fungsi dan sifat hak cipta Pasal 4

Hak Cipta sebagaimana dimaksud dalam Pasal 3 huruf a merupakan hak eksklusif yang terdiri atas hak moral dan hak ekonomi.

Pembatasan Pelindungan Pasal 26

Ketentuan sebagaimana dimaksud dalam Pasal 23, Pasal 24, dan Pasal 25 tidak berlaku terhadap:

- i. penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
- ii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
- iii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
- iv. penggunaan untuk kepentingan pendidikan dan pengembangan ilmu pengetahuan yang memungkinkan suatu Ciptaan dan/atau produk Hak Terkait dapat digunakan tanpa izin Pelaku Pertunjukan, Produser Fonogram, atau Lembaga Penyiaran.

Sanksi Pelanggaran Pasal 113

1. Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

SISTEM BASIS DATA MYSQL

PADA KONSEP

JARINGAN KLIEN SERVER

Abdul Rozaq



SISTEM BASIS DATA MYSQL PADA KONSEP JARINGAN KLIEN SERVER

Penulis :
Abdul Rozaq

e-ISBN :
978-623-92412-6-1 (PDF)

Editor dan Penyunting :
Reza Fauzan

Desain Sampul dan Tata Letak :
Eko Sabar Prihatin; Rahma Indera

Penerbit :
POLIBAN PRESS
Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang
Dilarang memperbanyak karya tulis ini dalam bentuk
dan dengan cara apapun tanpa ijin tertulis dari penerbit

Redaksi :
Politeknik Negeri Banjarmasin, Jl. Brigjen H. Hasan Basry,
Pangeran, Komp. Kampus ULM, Banjarmasin Utara
Telp : (0511)3305052
Email : press@poliban.ac.id

Dicetak oleh :
PERCETAKAN DEEPUBLISH
Jl.Rajawali, G. Elang 6, No 3, Drono, Sardonoharjo, Ngaglik, Sleman
Jl.Kaliurang Km.9,3 – Yogyakarta 55581
Telp/Faks: (0274) 4533427
Website: www.deepublish.co.id
www.penerbitdeepublish.com
E-mail: cs@deepublish.co.id

Katalog Dalam Terbitan (KDT)

Abdul Rozaq—Cet. I. — Sistem Basis Data Mysql pada Konsep Jaringan Klien Server :
Poliban Press, 2019.

x; 80 hlm.; 15.5x23 cm

Ucapan Terima Kasih

Ucapan terima kasih kami sampaikan kepada Poliban Press karena telah mempercayakan proses percetakan buku Sistem Basis Data MySql pada Konsep Jaringan Klien Server kepada Penerbit Deepublish. Semoga buku ini dapat memberikan manfaat kepada seluruh pembaca dan kerja sama ini dapat terus terjalin.



Kata Pengantar

Puji syukur kehadiran Allah SWT atas limpahan rahmat dan karunianya sehingga Buku Sistem Basis Data MySql Pada Konsep Jaringan Klien Server tahun 2019 telah dapat diselesaikan. Buku ini merupakan pengantar bagi mahasiswa Program Studi Manajemen Informatika dalam perkuliahan Sistem Basis Data di Politeknik.

Terima kasih disampaikan kepada Joni Riadi S.ST., M.T. selaku Direktur Politeknik Negeri Banjarmasin dan Nurmahaludin, S.T., M.T. selaku Ketua Pusat Penelitian dan Pengabdian Masyarakat beserta sekretaris dan staf. Terima kasih juga disampaikan kepada Faris Ade Irawan, Reza Fauzan, Eko Sabar Prihatin dan Rahma Indera yang telah berkontribusi dalam editing serta seluruh tim Poliban Press dan semua pihak yang telah ikut membantu dalam penyelesaian buku ini. Kami menyadari masih terdapat kekurangan dalam buku ini untuk itu kritik dan saran terhadap penyempurnaan buku ini sangat diharapkan. Semoga buku ini dapat memberi manfaat bagi mahasiswa Politeknik khususnya dan bagi semua pihak yang membutuhkan.

Banjarmasin, Agustus 2019

Ketua Poliban Press

Prakata

Assalamuallaikum Wr. Wb.

Tak cukup terungkap kata untuk mengucapkan syukur kepada Ilahi Robi yang telah memberikan limpahan rahmat dan hidayah pada Kita semua, yaitu diberkahinya hidayah iman dan Islam serta memberikan karunia ilmu yang bermanfaat bagi diri sendiri dan orang lain.. Sehingga dapat terselesaikannya Buku Ajar dengan judul “**Sistem Basis Data MySql pada Konsep Jaringan Klien Server**” untuk bahan perkuliahan khususnya di Program Studi Manajemen Informatika Jurusan Administrasi Bisnis Politeknik Negeri Banjarmasin.

Rasa terima kasih yang tak terhingga kepada Kedua Orang Tua yang selama ini telah dengan sabar mendidik dan memberikan kesempatan kepada penyusun untuk menimba ilmu, dan semoga apa yang telah diberikan mendapat balasan yang lebih dari Allah SWT.

Kesempurnaan hanyalah Allah SWT yang memilikinya, dan Kita hanyalah manusia biasa yang tak luput dari salah dan dosa. Kiranya para pembaca dalam mencermati buku ajar ini bisa memberikan sumbang saran dan kritik yang nantinya bisa digunakan dalam mengoreksi serta mengevaluasi bahan perkuliahan ini. Atas kritik dan saran yang diberikan, diucapkan banyak terima kasih, semoga Allah membalas kebaikan yang pembaca sampaikan dengan berlipat ganda amin.

Akhirnya, semoga apa yang disampaikan dapat bermanfaat bagi penulis khususnya dan semua pihak yang berkepentingan pada umumnya.

Wassalamuallaikum Wr. Wb.

Banjarmasin, Agustus 2019

Penulis

Daftar Isi

Ucapan Terima Kasih	v
Kata Pengantar	vi
Prakata.....	vii
Daftar Isi	viii
BAB 1 SISTEM JARINGAN KLIEN SERVER.....	1
1.1 Konfigurasi Jaringan Adhoc	2
1.2 Jaringan Server Based/Wireless Infrastruktur.....	10
1.3 Soal Latihan	16
BAB 2 SISTEM KEAMANAN PADA MYSQL.....	17
2.1 Database.....	17
2.2 Keamanan Database.....	21
2.3 Soal Latihan	25
BAB 3 DATA DEFINITION LANGUAGE.....	27
3.1 DDL (Data Definition Language)	27
3.2 Contoh Studi Kasus.....	33
3.3 Soal Latihan	35
BAB 4 DATA MANIPULATION LANGUAGE	37
4.1 Data Manipulation Language (DML)	37
4.2 Contoh Studi Kasus.....	40
4.3 Soal Latihan	41
BAB 5 VIEW DAN INDEX.....	43
5.1 View dan Index.....	43
5.2 Contoh Studi Kasus.....	46
5.3 Soal Latihan	48
BAB 6 TRIGGER.....	49
6.1 Trigger.....	49

6.2	Contoh Studi Kasus.....	50
6.3	Soal Latihan	52
BAB 7	STORE PROSEDURE.....	53
7.1	Store Procedure	53
7.2	Contoh Studi Kasus.....	54
7.3	Soal Latihan	56
BAB 8	FUNCTION.....	59
8.1	Function	59
8.2	Contoh Studi Kasus.....	60
8.3	Soal Latihan	70
BAB 9	TRANSACT SQL.....	71
9.1	Transact SQL.....	71
9.2	Contoh Studi Kasus.....	72
9.3	Soal Latihan	77
	Daftar Pustaka	80

BAB 1

SISTEM JARINGAN KLIEN SERVER

Capaian Pembelajaran:

1. Mampu memahami konfigurasi jaringan klien server untuk implementasi Sistem Basis Data
2. Mampu melakukan konfigurasi jaringan klien server untuk implementasi Sistem Basis Data

Klien-Server adalah sebuah paradigma teknologi informasi yang merujuk kepada cara untuk mendistribusikan aplikasi ke dalam dua pihak: Pihak Klien dan Pihak Server. Klien-server sebagai arsitektur yang paling banyak digunakan saat ini. Dimana klien dapat melakukan proses sendiri, ketika klien meminta data, server akan mengirimkan data sesuai yang diminta, kemudian proses akan dilakukan di klien.

- **Klien** adalah sembarang sistem atau proses yang melakukan suatu permintaan data atau layanan ke server.
- **Server** adalah sistem atau proses yang menyediakan data atau layanan yang diminta oleh klien. Secara fisik, sebuah server dapat berupa komputer mainframe, mini-komputer, workstation, ataupun PC atau peranti lain seperti printer, server tidak harus berupa sistem fisik, tetapi juga suatu proses.

Dalam model Klien/Server, sebuah aplikasi dibagi menjadi dua bagian yang terpisah, tapi masih merupakan sebuah kesatuan yakni komponen Klien dan komponen Server. Komponen klien juga sering disebut sebagai *front-end*, sementara komponen server disebut sebagai *back-end*. Komponen klien dari aplikasi tersebut dijalankan

dalam sebuah workstation dan menerima masukan data dari pengguna. Komponen klien tersebut akan menyiapkan data yang dimasukkan oleh pengguna dengan menggunakan teknologi pemrosesan tertentu dan mengirimkannya kepada komponen server yang dijalankan di atas mesin server, umumnya dalam bentuk *request* terhadap beberapa layanan yang dimiliki oleh server. Komponen server akan menerima request dari klien dan langsung memprosesnya lalu mengembalikan hasil pemrosesan tersebut kepada klien. Klien pun menerima informasi hasil pemrosesan data yang dilakukan server dan menampilkannya kepada pengguna, dengan menggunakan aplikasi yang berinteraksi dengan pengguna.

Pada kesempatan kali ini, konsep jaringan klien server akan dibangun dalam rangka mengimplementasikan sistem basis data. Adapun konsep jaringan yang dimaksud akan fokus pada dua konfigurasi Tipe Jaringan Wireless yang dapat digunakan, yaitu:

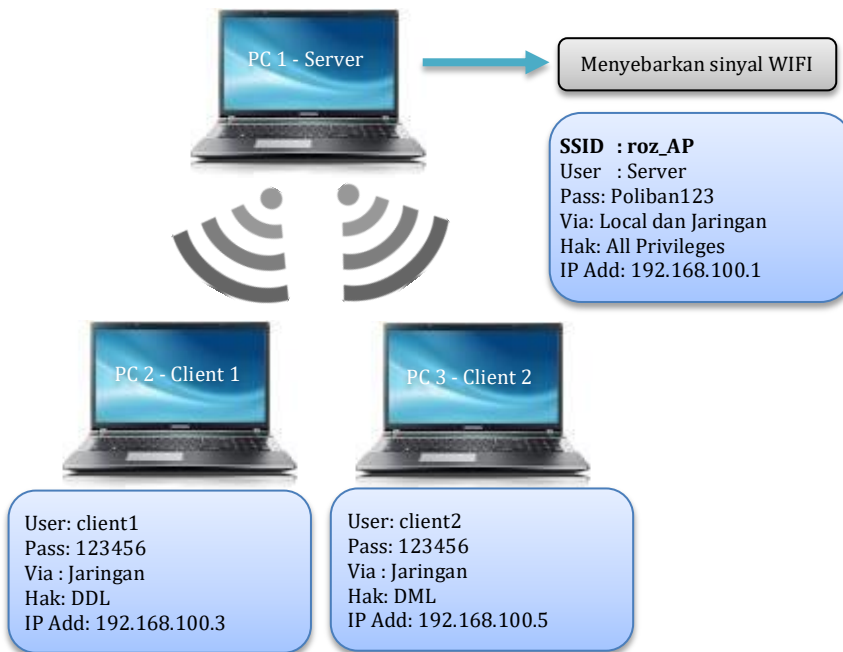
1. Jaringan peer to peer/Ad Hoc Wireless LAN Komputer dapat saling berhubungan berdasarkan nama SSID (Service Set Identifier). SSID adalah nama identitas komputer yang memiliki komponen nirkabel.
2. Jaringan Server Based/Wireless Infrastruktur Sistem Infrastruktur membutuhkan sebuah komponen khusus yang berfungsi sebagai Access Point.

1.1 Konfigurasi Jaringan Adhoc

Konfigurasi jaringan Adhoc akan dijelaskan dengan sebuah skenario studi kasus, yaitu Suatu minimarket ingin membuat sebuah sistem basis data untuk menangani sistem informasi penjualan minimarket tersebut. Di minimarket itu memiliki 3 buah PC yang belum dikonfigurasi. Dari dua buah PC tersebut salah satunya akan dibuat menjadi PC SERVER, dan sisanya lagi menjadi PC klien yang tugasnya berbeda. PC klien tersebut akan ditugaskan masing – masing dapat mengoperasikan perintah DDL (Create, Alter, dan Drop) dan DML (Insert, Select, Update, dan Delete).

Dapat diketahui:

- PC 1 sebagai Server (Menyebarkan Wifi/Access Point)
- PC 2 Sebagai Klien 1 dengan Hak Akses DDL
- PC 3 Sebagai Klien 2 dengan Hak Akses DML



Gambar 1.1. Konsep Jaringan Adhoc

A. Langkah-langkah untuk membuka hotspot di Windows 8 - 10 (Sebagai Server)

1. Tekan tombol Windows + X. Lalu pilih **Command Prompt Administrator** atau klik fitur pencarian (Search), lalu ketik CMD. Kemudian klik kanan, pilih '**Run as Administrator**'
2. Ketik '**netsh wlan set hostednetwork mode=allow ssid=sbd key=poliban123**' (tanpa tanda kutip) dan tekan tombol Enter. SSID dan KEY bisa diisi sesuai dengan kebutuhan kita.
3. Kemudian, ketikkan '**netsh wlan start hostednetwork**' untuk memulai hotspot dan tekan tombol Enter.

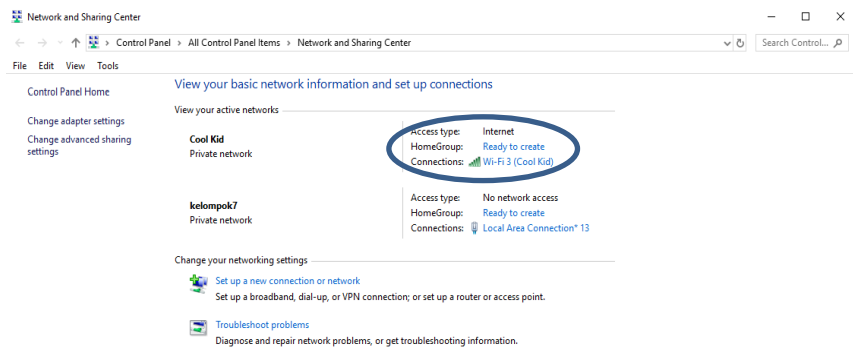
```
Administrator: Command Prompt
C:\WINDOWS\system32>netsh wlan set hostednetwork mode=allow ssid=kelompok7 key=poliban123
The hosted network mode has been set to allow.
The SSID of the hosted network has been successfully changed.
The user key passphrase of the hosted network has been successfully changed.

C:\WINDOWS\system32>netsh wlan start hostednetwork
The hosted network started.

C:\WINDOWS\system32>
```

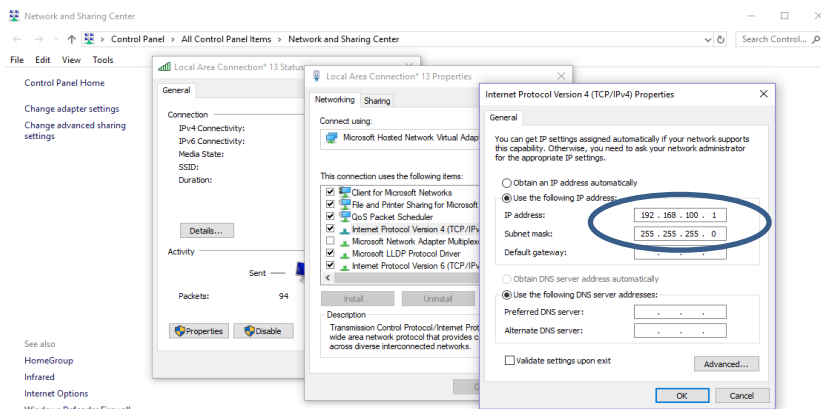
Gambar 1.2. Setting Adhoc pada Win 8-10

4. Buka Control Panel, lalu klik '**Network and Sharing Center**'.
Kemudian klik '**Wi-Fi**'.



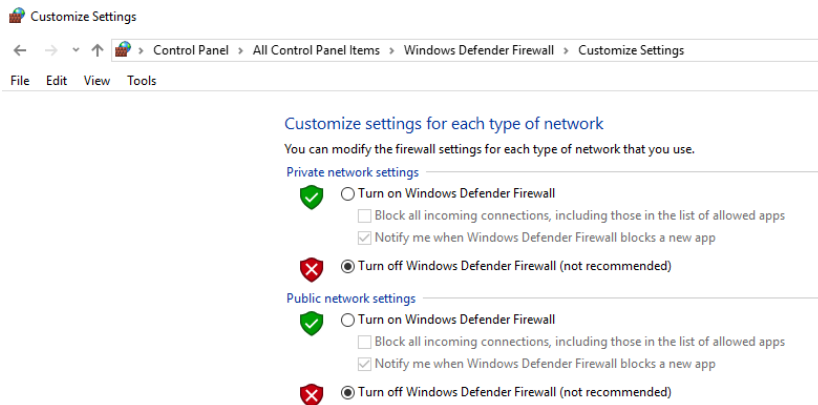
Gambar 1.3. Konfigurasi Wi-Fi

5. Klik '**Properties**', kemudian cari dan pilih '**Internet Protocol Version 4**'. Ketika tombol properties sudah tersedia, kemudian klik tombol tersebut. Selanjutnya kita isi **IP address** secara **manual**, kalau sudah selesai maka klik **Subnet Mask**. Maka subnet mask akan terisi sendiri secara otomatis.



Gambar 1.4. Setting Manual IP Address

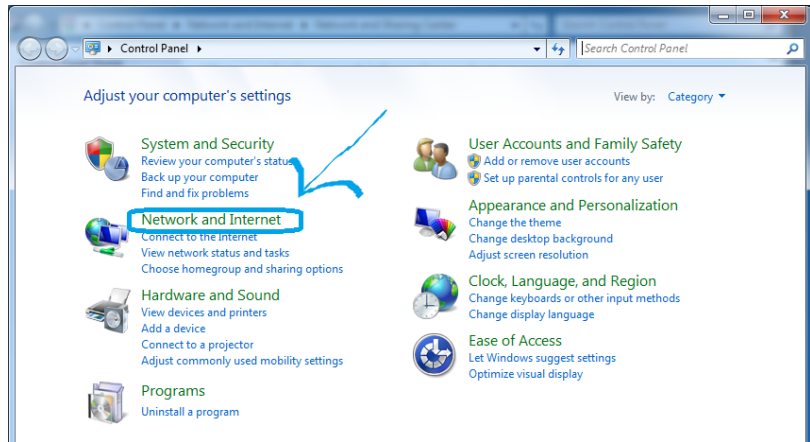
6. Buka **Control Panel**, pilih **'Window Firewall'**. Lalu pilih **'Turn Window Firewall On or Off'**, kemudian Pilih **'Turn Off'** pada kedua fitur yang ada.



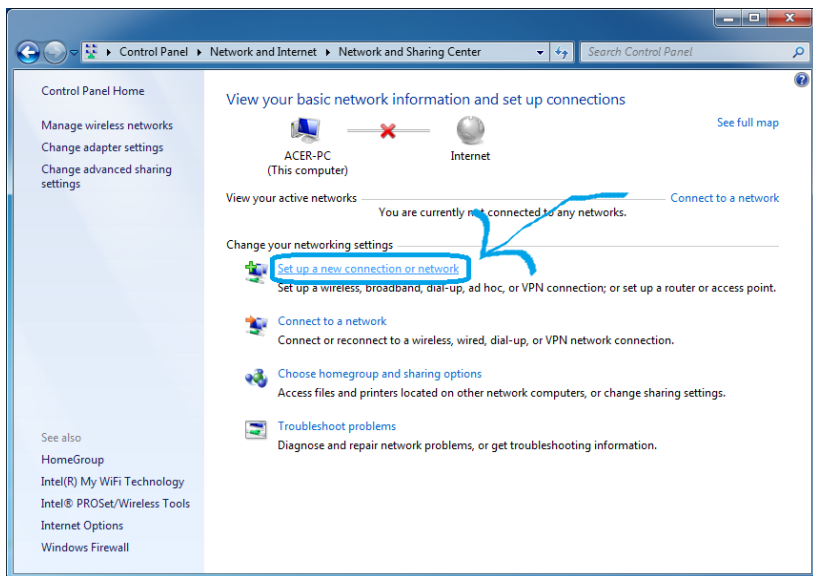
Gambar 1.5. Setting Firewall pada Win 8-10

B. Langkah-langkah untuk membuka hotspot di Windows 7

1. Klik Tombol Start, lalu pilih **'Control Panel'**. Kemudian pilih **'Netwok and Sharing' Center** dan klik **'Set Up a New Connection or Network'** yang berada dibawah **'Change Your Networking Settings'**

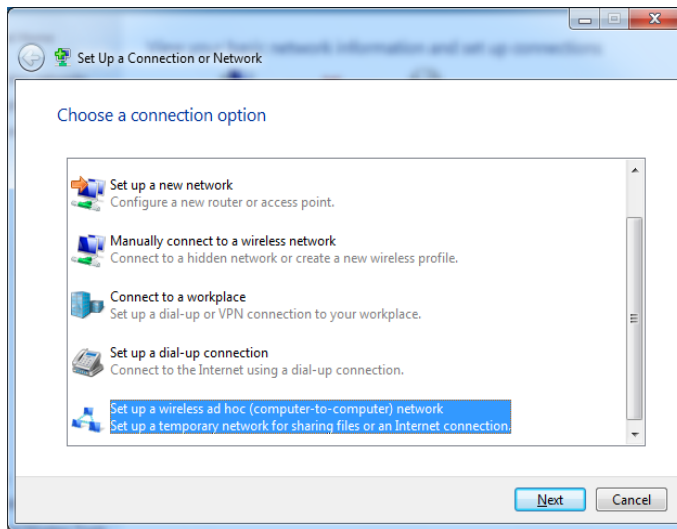


Gambar 1.6. Setting Network and Internet



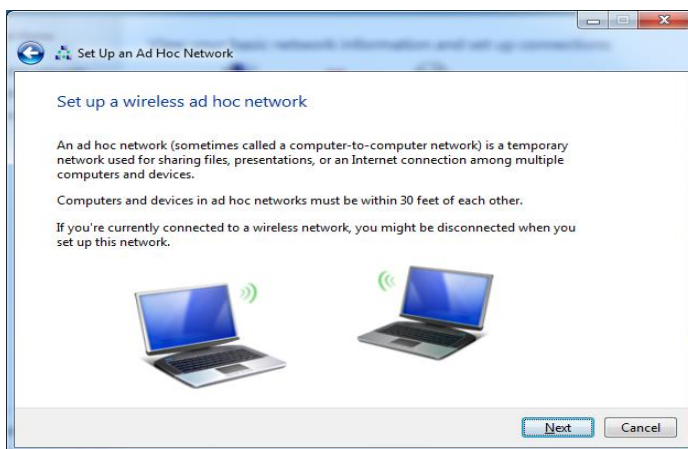
Gambar 1.7. Set up a new connection or network

Selanjutnya akan muncul tulisan '**Choose a Connection Option**', kita pilih '**Set up a Wireless Ad Hoc (Computer to Computer) Network**' dan klik Next.



Gambar 1.8. Set up a wireless ad hoc

2. Lalu akan muncul lagi tampilan **'Set up a Wireless Ad Hoc Network'** dan kli Next saja.

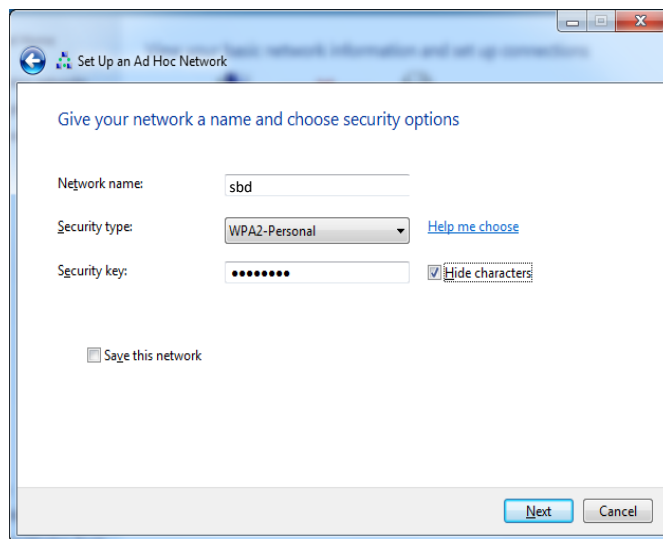


Gambar 1.9. Set up a wireless ad hoc network

3. Kemudian akan muncul tampilan **'Give Your Network a Name and Choose Security Option'**. Pada tahap ini kita akan

membuat nama jaringan Ad Hoc dan keamanannya. Dalam keamanan ada 3 tipe, yaitu:

- **No Authentication (Open)**, membuat jaringan Ad Hoc tanpa nantinya kita akan memerlukan password untuk masuk ke jaringan Ad Hoc tersebut.
- **WEP (Wired Equivalent Privacy)**, merupakan suatu metode untuk kita membuat pengamanan atau password sebelum masuk ke dalam Acces Point. Dan kita dapat membuat password minimal dengan 5 karakter sampai 26 karakter.
- **WPA - 2 (Wi-Fi Protected Acces 2)**, pengamanan jaringan jenis terbaru dan lebih bagus daripada WEP yaitu lebih sulit untuk di crack atau di bobol. kita dapat membuat password minimal dengan 8 karakter sampai 64 karakter.



Gambar 1.10. Set up a wireless ad hoc network name

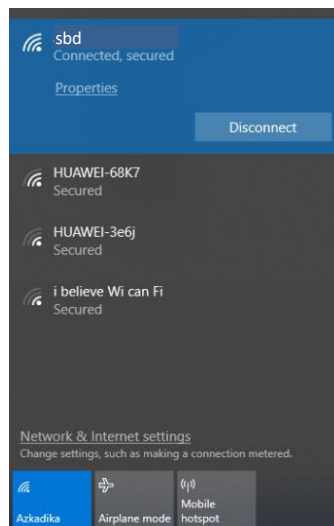
4. Setelah kita pilih salah satu dari metode pengamanan di atas, ceklist '**Save This Network**' untuk menyimpan jaringan Ad Hoc pada laptop atau bisa juga tidak di ceklist jika kita hanya

memakai jaringan Ad Hoc dan akan hilang jika sudah tidak digunakan lagi. Kemudian klik Next.

5. Kemudian jika muncul tulisan **'The PTI Network is Ready to Use'**, klik close.
6. Kembali lagi ke **'Network and Sharing Center'**. Sekarang kita akan mensetting IP Address dengan cara mengklik **'Wireless Network Connection (PTI)'**.
7. Klik **'Properties'**, Kemudian cari dan pilih **'Internet Protocol Version 4'**. Ketika tombol properties sudah tersedia, kemudian klik tombol tersebut. Selanjutnya kita isi **IP address** secara manual, kalau sudah selesai maka klik **Subnet Mask**. Maka subnet mask akan terisi sendiri secara otomatis.
8. Buka **Control Panel**, pilih **'Window Firewall'**. Lalu pilih **'Turn Window Firewall On or Off'**, kemudian Pilih **'Turn Off'** pada kedua fitur yang ada.

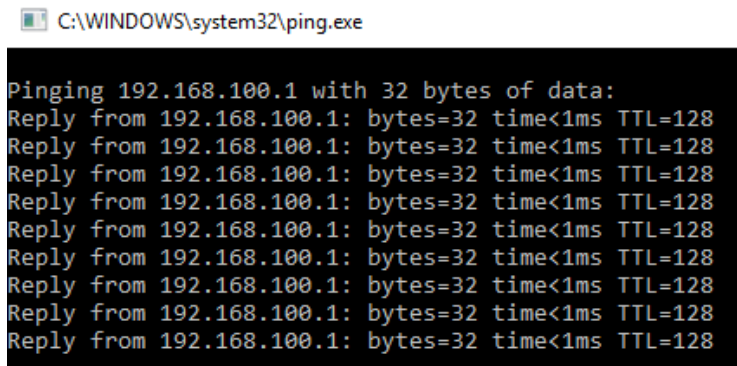
C. Mengetahui Server & Klien Terhubung

1. Pertama, connect kan PC Klien ke hotspot PC Server Sesuai Gambar 1.2 dengan SSID **"sbd"**



Gambar 1.11 Wifi SSID "sbd"

2. Dikarenakan PC Server kita setting IP Address nya secara manual, maka untuk IP Address PC Klien juga harus di setting secara manual sesuai dengan gambar 1.1 dan gambar 1.4.
3. Setelah Connect, tekan tombol **Windows + R**. Lalu ketikkan '**Ping 192.168.100.1 -t**'. IP address bisa kita sesuaikan dengan IP address Server. Kemudian tekan Tombol Enter.



```
C:\WINDOWS\system32\ping.exe

Pinging 192.168.100.1 with 32 bytes of data:
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
```

Gambar 1.12. Koneksi klien ke server berhasil

1.2 Jaringan Server Based/Wireless Infrastruktur

Konfigurasi Jaringan Server Based/Wireless Infrastruktur Sistem Infrastruktur membutuhkan sebuah komponen khusus yang berfungsi sebagai Access Point. Sebagai contoh adalah Alat berupa **Access Point** dan **Handphone** yang dapat difungsikan sebagai Access Point melalui fasilitas **tethering** atau **hotspot selular**.

Pada kesempatan kali ini kita akan mencoba menggunakan perangkat Handphone yang kita fungsikan sebagai hotspot. Adapun konsep jaringan yang dimaksud dapat dilihat seperti gambar 1.13 di bawah ini:

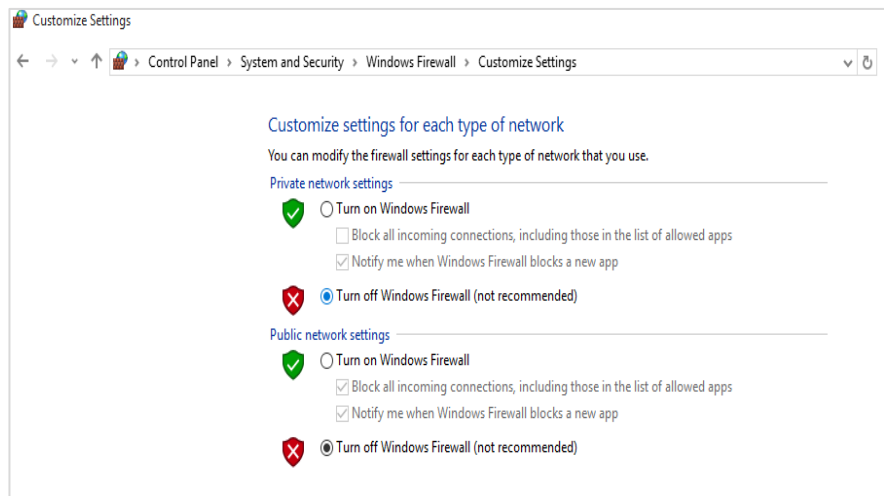
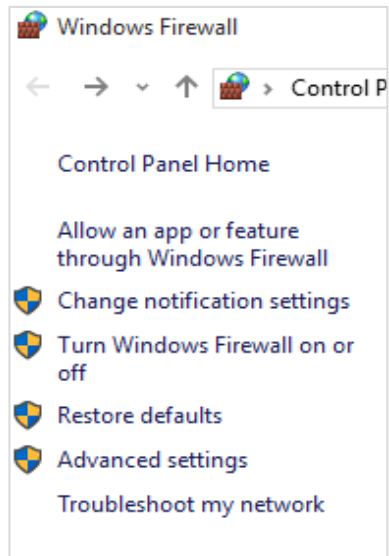


Gambar 1.13 Konsep Jaringan Wireless Infrastruktur

A. Mengatur Server Jaringan

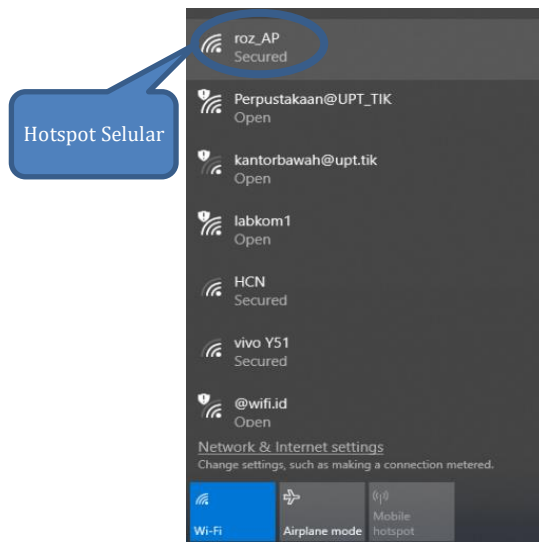
Disini karena kami menggunakan laptop dengan OS Windows 10 dan jaringan Ad-Hoc di ganti menggunakan **Hotspot Seluler**. Maka akan dijelaskan bagaimana cara mengonfigurasi jaringan dengan menggunakan **Hotspot Seluler** dengan Windows 10.

1. Terlebih dahulu matikan Windows Firewall nya untuk Windows 2010. Pilih "Change Notfication Settings". Kemudian pilih "Turn Off Windows Firewall (Not Recommended)". Kemudian Klik "OK".



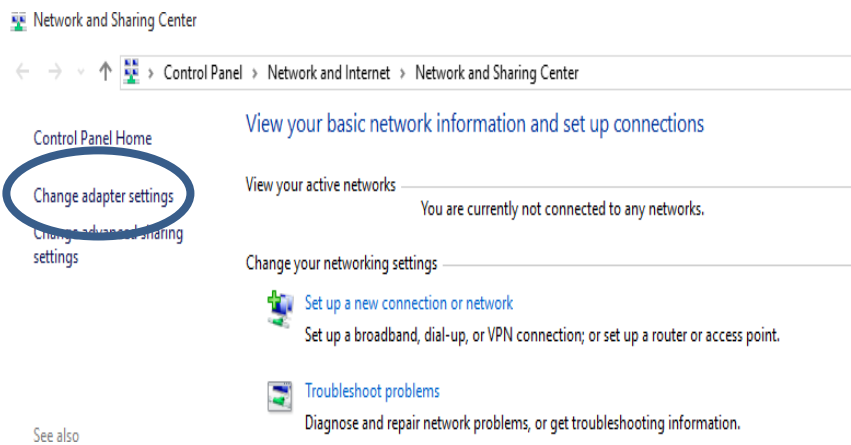
Gambar 1.14 Turn Off Windows Firewall

2. Sambungkan Wifi Laptop Ke Hotspot Seluler.



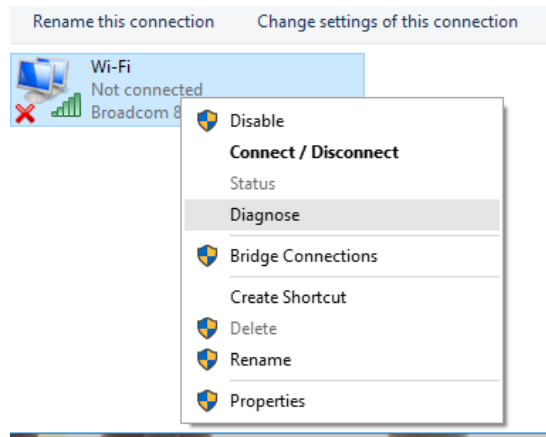
Gambar 1.15 Koneksi ke hotspot seluler

3. Selanjutnya Atur “IP Address” dengan Mengklik “change adapter settings”



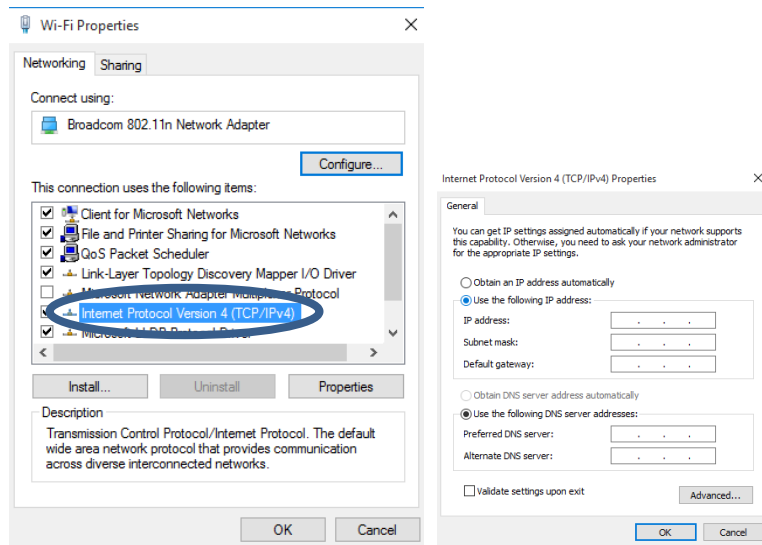
Gambar 1.16 change adapter setting

4. Selanjutnya Klik Kanan “Wi-Fi” pilih “Properties”



Gambar 1.17 Wi-fi Propertis

5. Selanjutnya masih berhubungan dengan Nomer “4”, setelah itu pilih “Internet Protocol Version 4 (TCP/Ipv4)”. Pilih “Properties”.



Gambar 1.18 Konfigurasi IP Address

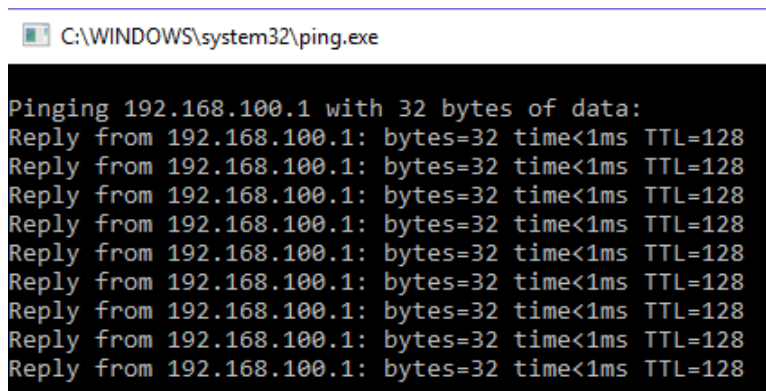
6. Selanjutnya kita langsung bisa mengatur “IP Address”. Pilih “Use the following IP address:” dan pilih “Use the Following DNS server addresses”.

Kemudian pilih “OK”. Jangan lupa yah isi kolom yang kosong tersebut. Contoh untuk server “192.168.100.1”, selanjutnya bisa meilihat gambar 1.12 dan sekarang sudah bisa digunakan. ☺

Jika system jaringan diataur sebagai Dinamic Host Control Protocol (DHCP), maka IP address akan mengikuti atau diberi oleh server dan atau Hotspot (WIFI, HP).

B. Mengetahui Server & Klien Terhubung

1. Pertama, connect kan PC Klien ke hotspot selular Sesuai Gambar 1.14 dengan SSID “**roz_AP**” atau sesuai hotspot selular yang sudah diaktifkan di masing-masing Handphone.
2. Dikarenakan PC Server kita setting IP Address nya secara manual, maka untuk IP Address PC Klien juga harus di setting secara manual sesuai dengan gambar 1.12 dan gambar 1.7.
3. Setelah Connect, tekan tombol **Windows + R**. Lalu ketikan ‘**Ping 192.168.100.1 -t**’. IP address bisa kita sesuaikan dengan IP address Server. Kemudian tekan Tombol Enter.



```
C:\WINDOWS\system32\ping.exe

Pinging 192.168.100.1 with 32 bytes of data:
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
Reply from 192.168.100.1: bytes=32 time<1ms TTL=128
```

Gambar 1.19. Koneksi klien ke server berhasil

1.3 Soal Latihan

Pusat Penelitian dan Pengabdian Kepada Masyarakat (P3M) Poliban ingin mengembangkan sistem basis data yang di dukung oleh perangkat jaringan yang sudah tersedia di lingkungan kampus Poliban. Diantaranya sudah tersedia perangkat jaringan wireless di setiap sudut kampus dengan sistem konfigurasi DHCP dan 2(dua) perangkat komputer yang tersedia di ruang P3M yang dapat digunakan sebagai klien. Sedangkan sistem basis data tersimpan di server UPT. TIK.

Dari uraian di atas, anda diminta untuk melakukan:

1. Gambarkan konsep jaringan yang dapat diimplementasikan dari studi kasus di atas!
2. Lakukan konfigurasi jaringan yang sudah anda konsepkan, yang selanjutnya buat dokumentasinya dari setiap tahapan yang sudah dilakukan sampai berhasil terkoneksi antara klien dengan server!

BAB 2

SISTEM KEAMANAN PADA MYSQL

Capaian Pembelajaran:

1. Mampu memahami sistem keamanan pada MySQL dengan melakukan pembagian hak akses untuk setiap user
2. Mampu menerapkan sistem keamanan pada MySQL dengan melakukan pembagian hak akses untuk setiap user

2.1 Database

Dalam sebuah Perusahaan, Organisasi maupun Institusi harus bisa bersaing di era Teknologi sekarang. Salah satu teknologi yang harus di miliki oleh sebuah Perusahaan, Organisasi maupun Institusi yaitu Sistem Pengolahan Basis Data atau Database. Penggunaan Database sangat berguna dalam suatu Perusahaan, Organisasi, dan Institusi misalnya, mempermudah pencarian nama karyawan, mencari jumlah barang atau harga yang akan dijual dan masih banyak lagi yang lainnya

A. Pengertian Database

Database adalah sekumpulan data yang sudah disusun sedemikian rupa dengan ketentuan atau aturan tertentu yang saling berelasi sehingga memudahkan pengguna dalam mengelola data dan juga mempermudah dalam memperoleh informasi. Selain itu, Database dapat juga disebut sebagai kumpulan file, tabel, atau arsip yang saling terhubung yang disimpan dalam media elektronik.

B. Manfaat Database

1. Kecepatan dan Kemudahan

Database dapat menyeleksi data menjadi suatu kelompok yang terurut tepat dan cepat.

2. Kontrol *Data Terpusat*

Suatu Perusahaan pasti memiliki banyak bagian atau divisi. Akan tetapi, database yang diperlukan tetap satu saja. Hal ini mempermudah kontrol data seperti ketika mengupdate data karyawan, maka kita hanya perlu mengupdate data di divisi yang bersangkutan. Tetapi, cukup di satu database yang ada di server pusat

3. Pemakaian *Bersama – sama*

Suatu Database dapat digunakan oleh siapa. Contohnya dalam sebuah kampus, Database mahasiswa dapat diakses oleh beberapa bagian seperti bagian admin, bagian akademik, bagian jurusan, bagian keuangan

4. Keamanan *Data*

Hampir semua Aplikasi manajemen database sekarang memiliki fasilitas manajemen pengguna. Manajemen pengguna ini mampu membuat hak akses yang berbeda-beda disesuaikan dengan kepentingan maupun posisi pengguna. Selain itu data yang tersimpan di database diperlukan password untuk mengaksesnya.

5. Menghemat *Biaya Perangkat*

Dengan memiliki database secara terpusat maka di masing-masing divisi tidak memerlukan perangkat untuk menyimpan database terhubung database yang dibutuhkan hanya satu yaitu yang disimpan di server pusat, ini tentunya memangkas biaya pembelian perangkat.

6. Memudahkan *Dalam Pembuatan Aplikasi Baru*

Dalam Hal ini, perusahaan yang memiliki database yang sangat baik tidak perlu lagi membuat database baru lagi atau tidak perlu lagi mengubah struktur database yang sudah ada. Sehingga sang programmer hanya membuat atau mengatur antarmuka aplikasinya saja.

7. Meminimalisir *Kesalahan*

Database dapat mengurangi kesalahan mekanis yang disebabkan oleh faktor manusia yang sebaiknya dikerjakan oleh mesin. Misalnya memasukkan data karyawan yang ada di suatu divisi atau bagian.

C. **Penyalahgunaan Database**

1. Tidak Disengaja
 - a) Kerusakan sistem selama proses transaksi
 - b) Anomali yang disebabkan oleh akses database yang konkuren misalnya *Lost Update Problem*, yaitu Masalah operasi update yang sukses dari seorang pengguna kemudian ditimpali oleh operasi update dari pengguna lain.
 - c) Anomali yang disebabkan oleh pendistribusian data pada beberapa komputer
 - d) Logika error yang mengancam kemampuan transaksi untuk mempertahankan konsistensi database.
2. Disengaja
 - a) Pengambilan data/pembacaan data oleh pihak yang tidak berwenang
 - b) Pengubahan dan penghapusan data oleh pihak yang tidak berwenang

D. **Keamanan Database**

Keamanan merupakan suatu proteksi terhadap pengrusakan data dan pemakaian data oleh pemakai yang tidak punya kewenangan. Untuk menjaga keamanan basis data dapat dengan:

- 1) Penentuan perangkat lunak database server yang handal
- 2) Pemberian otoritas kepada user mana saja yang berhak mengakses, serta memanipulasi data-data yang ada.

E. **Tujuan Keamanan Basis Data**

1. Secrecy/Confidentiality
Informasi tidak boleh diungkapkan kepada pengguna yang tidak sah. Sebagai contoh, seorang karyawan tidak boleh

memberitahu informasi database perusahaannya kepada karyawan perusahaan lain.

2. Integrity

Hanya pengguna yang berwenang harus diizinkan untuk memodifikasi data. Sebagai contoh, Mahasiswa hanya bisa melihat nilainya. Akan tetapi dia tidak bisa mengubah nilainya, karena dia tidak diberi wewenang untuk memodifikasi nilainya sendiri tanpa persetujuan jurusan atau akademik atau bagian yang bersangkutan.

3. Availability

Maksud dari *availability* adalah memastikan sumber daya yang ada siap diakses kapanpun oleh *user/application/sistem* yang membutuhkannya. Serta menjamin pengguna yang valid selalu bisa mengakses informasi dan sumber daya miliknya sendiri. Untuk memastikan bahwa orang-orang yang berhak tidak ditolak untuk mengakses informasi yang memang menjadi haknya. Contohnya, akun email yang diberi password. Hanya pemiliknya akun tersebut lah yang bisa mengakses akun email mereka sendiri, tidak menutup kemungkinan Hacker bisa membobol akun email tersebut.

4. Legitimate Use

Menjamin kepastian bahwa sumber daya tidak dapat digunakan oleh orang yang tidak berhak. Contohnya, Penggunaan copyright, watermarking, dan digital signature agar data atau dokumen agar tidak dipalsukan atau diambil oleh orang yang tidak berwenang.

F. Ancaman Terhadap Keamanan Database

1. Secrecy/Confidentiality

– Password Strength

Lemahnya password yang digunakan, sehingga mudah ditebak ataupun di-bruteforce

– Malware

Masuknya virus yang dapat membuat backdoor ke sistem ataupun mengumpulkan informasi pengguna

- Social Engineering
Lemahnya security awareness pengguna dimana mudah sekali untuk 'dibohongi' oleh attacker, yang biasanya adalah orang yang sudah dikenalnya.
- 2. Integrity
 - Modification
Pemakai atau bagian yang tidak berhak mengakses sumber daya tapi juga merusak sumber daya sistem komputer
 - Fabrication
Pemakai atau bagian yang tidak berhak menyisipkan objek palsu kedalam sistem.
- 3. Availability
 - Interruption
Sumber daya basis data dirusak atau menjadi tidak dapat dipakai
 - Distributed Denial of Service
Sebuah usaha untuk membuat suatu sumber daya komputer menjadi tidak bisa dipakai oleh user-nya, dengan menggunakan ribuan zombie system yang 'menyerang' secara bersamaan. Tujuannya negatif, yakni agar sebuah website atau layanan online tidak bisa bekerja dengan efisien atau bahkan mati sama sekali, untuk sementara waktu atau selama-lamanya. DDoS attack adalah salah satu model dari DoS (denial-of-service) attack.
- 4. Legitimate Use
 - Authorization Violation
Pelanggaran Penyalahgunaan wewenang legal

2.2 Keamanan Database

A. Pengaturan Keamanan Database

1. Pemberian wewenang atau hak istimewa (Priviledge) untuk mengakses sistem atau objek database
2. Kendali otorisasi (kontrol akses) dapat dibangun pada perangkat lunak dengan 2 fungsi:

- Mengendalikan sistem atau objek yang dapat diakses
 - Mengendalikan bagaimana pengguna menggunakannya
3. Sistem administrasi yang bertanggungjawab untuk memberikan hak akses dengan membuat account pengguna

B. Memahami Hak Akses (Privileges) di MySQL

MySQL pada dasarnya merupakan sistem database yang aman. Di MySQL kita dapat mengatur hak akses tiap user terhadap data di database. MySQL memungkinkan kita mengatur hak akses user sampai pada tingkat kolom. Artinya kita dapat mengatur kolom tertentu dapat diakses oleh user siapa saja. Tentu, kita juga dapat mengatur hak akses user terhadap tabel, dan database. Semua pengaturan hak akses (*privileges*) tersimpan di database **mysql** yang secara *default* sudah ada di sistem MySQL. Di dalam database **mysql** antara lain terdapat tabel-tabel sebagai berikut:

- **user.** Tabel ini digunakan untuk menyimpan informasi user MySQL yang mencakup informasi user, password dan host user, serta informasi hak akses user.
- **db.** Tabel ini digunakan untuk menyimpan informasi mengenai hak akses user terhadap database.
- **host.** Tabel ini digunakan untuk menyimpan daftar komputer (bisa berupa alamat IP, nama komputer, atau %) yang berhak mengakses suatu database.
- **tables_priv.** Tabel ini digunakan untuk menyimpan informasi mengenai hak akses user terhadap tabel. Dengan kata lain menyimpan tabel ini dapat diakses oleh siapa dengan hak akses apa saja.
- **columns_priv.** Tabel ini digunakan untuk menyimpan informasi mengenai hak akses user terhadap kolom.
- **procs_priv.** Tabel ini digunakan untuk menyimpan informasi mengenai hak akses user terhadap procedure.
- **proc.** Tabel ini digunakan untuk menyimpan informasi mengenai daftar procedure dalam MySQL.

- **func.** Tabel ini digunakan untuk menyimpan informasi mengenai function yang didefinisikan di MySQL.

C. GRANT dan REVOKE di MySQL

Perintah GRANT dan REVOKE dapat digunakan untuk membuat user baru maupun mengatur hak akses user yang sudah ada dengan hak akses (privileges) tertentu. Tingkatan hak akses user dapat terbagi menjadi tingkatan global (tersimpan di tabel mysql.user), database (tersimpan di tabel mysql.host dan mysql.db), tabel (tersimpan di tabel mysql.tables_priv) dan kolom (tersimpan di tabel mysql.columns_priv). Setiap perubahan hak akses di MySQL, termasuk menambahkan user baru, tidak akan berlaku sebelum diakhiri dengan perintah **FLUSH PRIVILEGES**.

Berikut ini bentuk umum perintah GRANT dan REVOKE secara sederhana:

```
GRANT priv_type
ON {tbl_name | * | *.* | db_name.*}
TO user_name [IDENTIFIED BY 'password']
[WITH GRANT OPTION]

REVOKE priv_type
ON {tbl_name | * | *.* | db_name.*}i
FROM user_name
```

Berikut ini pilihan untuk priv_type dalam bentuk umum perintah GRANT dan REVOKE di atas:

All Priviliges	File	Reload
Alter	Index	Select
Create	Procces	Shutdown
Delete	Insert	Update
Drop	References	Usage

D. Menambahkan dan Mengatur Hak Akses User

Untuk menambahkan dan mengatur hak akses (*privileges*) user di MySQL, kita dapat menggunakan 2 cara.

1. **Pertama** langsung melakukan INSERT atau UPDATE ke tabel mysql.user, dan tabel-tabel lain sesuai dengan hak aksesnya. Cara ini tidak disarankan karena mengandung risiko terjadi kesalahan.
2. **Kedua** adalah dengan perintah GRANT dan REVOKE. Perintah ini mudah dipahami dan diterapkan karena lebih sederhana. MySQL secara otomatis akan menyimpan informasi user ke tabel sesuai dengan hak aksesnya.

Berikut ini beberapa contoh menambahkan user baru di MySQL:

- a. Menambahkan user baru dengan nama user '**admin**' yang dapat mengakses semua database dari komputer '**localhost**' dengan password '**admin123**'. User ini juga berhak menjalankan perintah GRANT untuk user lain.

```
GRANT ALL PRIVILEGES ON *.* TO
'admin'@'localhost'
IDENTIFIED BY 'admin123' WITH GRANT OPTION;
```

- b. Menambahkan user baru dengan nama user '**citra**', tidak dapat mengakses database (*.*), dapat mengakses dari komputer dengan IP '**192.168.1.5**' dan password '**123456**'.

```
GRANT USAGE ON *.* TO 'citra'@'192.168.1.5'
IDENTIFIED BY '123456';
```

- c. Menambahkan user baru dengan nama user '**super**', hanya dapat mengakses database '**mysql**', hanya dapat mengakses dari komputer '**localhost**' dan dengan password '**super123**'.

```
GRANT ALL PRIVILEGES ON mysql.* TO
'super'@'%'
IDENTIFIED BY 'super123';
```

Berikut ini beberapa contoh mengatur hak akses user yang sudah ada di MySQL:

- a. Mengubah hak akses user '**citra**' agar dapat mengakses database '**sipma**'.

```
GRANT ALL PRIVILEGES ON sipma.* TO  
'citra'@'192.168.1.5';
```

```
FLUSH PRIVILEGES ;
```

- b. Mengubah hak akses user '**super**' agar dapat CREATE di database '**sipma**'.

```
GRANT CREATE ON sipma.* TO  
'super'@'%';
```

```
FLUSH PRIVILEGES ;
```

E. Mengganti Password User

Untuk mengganti password suatu user di MySQL, kita tinggal menjalankan perintah UPDATE terhadap field Password di tabel mysql.user. Password tersebut dienkripsi dengan fungsi PASSWORD()

Berikut ini perintah SQL yang dapat digunakan untuk mengganti password user:

```
UPDATE user SET Password=PASSWORD('123456')  
WHERE  
User='super' AND Host='localhost';
```

```
SET PASSWORD FOR super@localhost = PASSWORD  
( '123456' );
```

```
FLUSH PRIVILEGES;
```

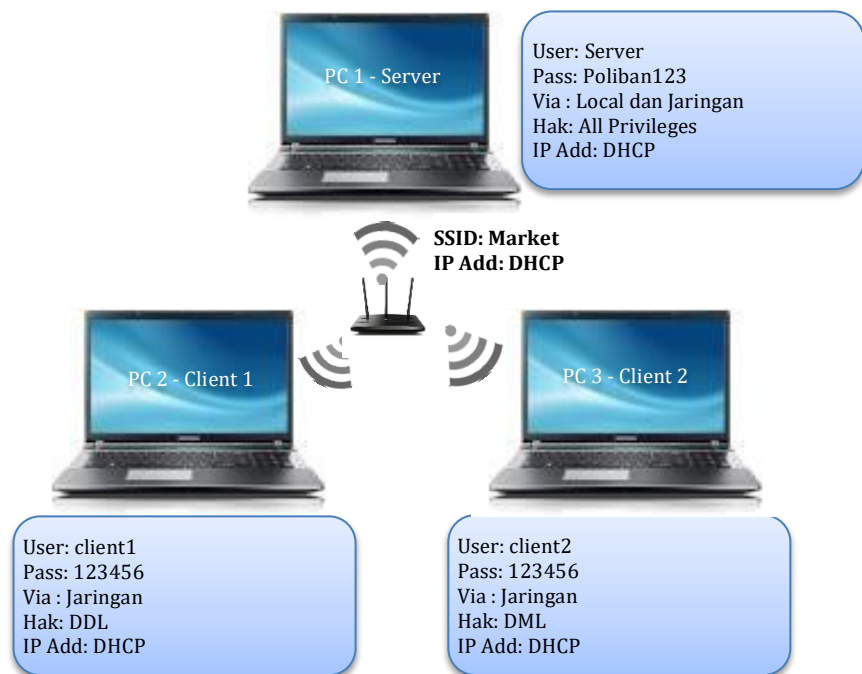
2.3 Soal Latihan

Suatu minimarket ingin membuat sebuah sistem basis data untuk menangani sistem informasi penjualan minimarket tersebut. Di minimarket itu memiliki 3 buah PC yang belum di konfigurasi. Dari dua buah PC tersebut salah satunya akan dibuat menjadi PC SERVER dengan nama database "**minimarket**" yang dibuat oleh user **server**,

dan sisa nya lagi menjadi PC client yang tugas nya berbeda. PC client tersebut akan ditugaskan masing – masing dengan hak DDL (Create, Alter, dan Drop) dan DML (Insert, Select, Update, dan Delete).

Dapat diketahui:

- PC 1 sebagai Server Admin
- PC 2 Sebagai Client1 dengan Hak Akses DDL
- PC 3 Sebagai Client2 dengan Hak Akses DML



Dari deskripsi studi kasus di atas, anda diminta untuk melakukan:

1. Konfigurasi sistem jaringan yang sesuai dengan gambar! Jika tidak tersedia perangkat Wifi, maka bisa menggunakan perangkat lain seperti HP yang bisa difungsikan sebagai perangkat Wifi.
2. Lakukan konfigurasi untuk pengaturan user dan pembagian hak aksesnya!

BAB 3

DATA DEFINITION LANGUAGE

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL (DDL) pada DBMS MySql
2. Mampu menerjemahkan dan mengimplementasikan konsep Entity Relationship Diagram (ERD) ke dalam bentuk DDL pada DBMS MySql

3.1 DDL (Data Definition Language)

Data Definition Language (DDL) merupakan sub bahasa SQL yang di gunakan untuk membuat, merubah maupun menghapus struktur atau definisi tipe data dari objek yang ada pada database. Atau juga merupakan kelompok perintah yang berfungsi untuk mendefinisikan atribut-atribut database, tabel, atribut kolom, batasan-batasan terhadap suatu atribut serta hubungan antar tabel.

FUNGSI UTAMA DDL:

- Membuat (create) objek tabel
- Modifikasi (modify) objek tabel
- Menghapus (delete) objek tabel

STATEMENT DDL:

Perintah yang digunakan untuk menjelaskan objek dari database.

Berikut beberapa statement DDL:

- **Create:** Perintah ini digunakan untuk membuat, termasuk diantaranya membuat database baru, tabel baru, view baru, dan kolom.

- **Alter:** Perintah ini digunakan untuk mengubah struktur tabel yang telah dibuat. Pekerjaannya mencakup mengganti nama tabel, menambah kolom, mengubah kolom, menghapus kolom, maupun memberikan atribut pada kolom.
- **Drop:** Perintah yang digunakan untuk menghapus objek database.

Berikut adalah perintah-perintah pada DDL (Data Definition Language)

1. Perintah untuk membuat/create

- **Create Database**

Berfungsi untuk membuat database baru.

Untuk perintah CREATE DATABASE tidaklah sesulit. Aturan penamaan sebuah database sama seperti aturan penamaan sebuah variabel, dimana **secara umum nama database boleh terdiri dari huruf, angka dan underscore (_)**. Jika database yang akan dibuat sudah ada, maka akan muncul pesan error. Bentuk umumnya adalah sebagai berikut:

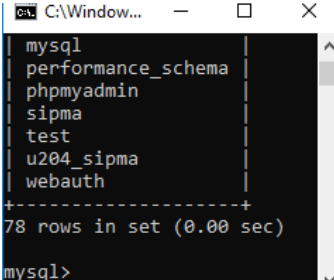
CREATE DATABASE nama_database;

Setelah pembuatan database, kita coba cek apakah benar sudah ada atau belum. Perintahnya adalah:

SHOW DATABASES;

- Create database sipma;
- Show databases;

Maka akan muncul semua database yang ada.



```

C:\Window...
mysql
| performance_schema |
| phpmyadmin         |
| sipma              |
| test               |
| u204_sipma         |
| webauth            |
+-----+
78 rows in set (0.00 sec)

mysql>

```

- Create Table yaitu perintah yang digunakan untuk membuat table baru pada database

Contoh:

CREATE TABLE nama_table (nama_kolom tipe_data NOT NULL/NULL, PRIMARY KEY (nama_kolom_primary_key), FOREIGN KEY (nama_kolom_foreign_key) REFERENCES table_referensi);

- Create Index berfungsi untuk membuat index pada database

CREATE INDEX index_name ON table_name (column_name);

- Create Trigger berfungsi untuk membuat Trigger pada database

**CREATE TRIGGER nama_trigger
Waktu_trigger kejadian_trigger
ON nama_tabel FOR EACH ROW pernyataan**

Keterangan:

- nama_trigger adalah nama trigger yang dikaitkan dengan suatu database.
- waktu_trigger menyatakan waktu trigger akan dipanggil secara otomatis. Dapat berupa BEFORE yang berarti “sebelum” atau AFTER yang berarti “sesudah”.
- Kejadian_trigger adalah kejadian yang membuat trigger dipanggil secara otomatis. Kemungkinan nilainya tercantum dalam Tabel 3.1.
- Nama_tabel menyatakan nama tabel yang terkait dengan trigger.
- Pernyataan dapat berupa pernyataan tunggal atau pernyataan majemuk(dengan menggunakan BEGIN..END).

Tabel 3.1 Kejadian untuk trigger

Kejadian	Keterangan
INSERT	Trigger dijalankan ketika terdapat operasi penambahan sebuah baris data ke tabel bersangkutan.
UPDATE	Trigger dijalankan ketika terdapat operasi perubahan

Kejadian	Keterangan
	sebuah baris dalam tabel bersangkutan.
DELETE	Trigger dijalankan ketika terdapat operasi penghapusan sebuah baris pada tabel bersangkutan.

Terkait dengan trigger terdapat alias dengan nama OLD dan NEW.

Penggunaannya:

NEW.nama_kolom

OLD.nama_kolom

Keterangan:

- OLD.nama_kolom menyatakan nilai nama_kolom sebelum dihapus atau diubah.
- NEW.nama_kolom menyatakan nilai nama_kolom setelah diubah atau setelah data baru dimasukkan.
- Create Procedure berfungsi untuk membuat procedure pada database

**CREATE PROCEDURE nama_sp ([param_pros [.....]])
bagian_kode**

Dalam hal ini:

✚ *nama_sp* menyatakan nama prosedur

✚ *param_pros* menyatakan definisi untuk parameter prosedur

✚ *bagian_kode* berupa pernyataan-pernyataan SQL

Adapun bentuk *param_pros* berupa:

[IN | OUT | INOUT] *nama_param tipe_data*

IN berarti parameter sebagai masukan bagi prosedur, OUT berarti parameter sebagai keluaran, dan INOUT berarti sebagai masukan dan sekaligus keluaran. Contoh penggunaan tentang hal ini akan dibahas belakangan. Namun, perlu diketahui, secara bawaan(default) parameter dianggap sebagai IN jika tidak disebutkan IN, OUT, atau INOUT.

Contoh berikut memberikan gambaran pembuatan sebuah prosedur yang sangat sederhana, yang ditujukan untuk menampilkan semua data dari table guest:

```
mysql> delimiter //
mysql> create procedure coba1()
-> begin
-> select * from guest;
-> end //
Query OK, 0 rows affected (0.44 sec)
```

Penjelasannya:

- ✚ Perintah “DELIMITER //” digunakan untuk menyatakan bahwa pengakhiran perintah tidak lagi berupa ; tetapi berupa //. Hal ini untuk mencegah pengekseskuan ketika menuliskan pernyataan SQL dalam pendefinisian prosedur.
 - ✚ Perintah “CREATE PROCEDURE coba1()” merupakan pernyataan untuk membuat prosedur bernama coba1, tanpa mengandung parameter.
 - ✚ Bagian BEGIN...END berisi pernyataan-pernyataan, tetapi pada contoh ini hanya ada satu pernyataan SELECT. Bagian ini bersifat opsional (boleh tidak disertakan, kalau prosedur hanya melibatkan satu pernyataan).
 - ✚ Tanda // digunakan untuk mengakhiri pembuatan prosedur. Tanda ini sesuai yang tercantum dalam perintah “DELIMITER //”.
- Create Function berfungsi untuk membuat fungsi pada database
CREATE FUNCTION nama_Ft ([param_fung (.....)])

RETURNS tipe

Bagian_kode

Keterangan:

- Nama_ft menyatakan nama fungsi tersimpan,
- Param_fung menyatakan variabel untuk parameter fungsi tersimpan,
- RETURNS tipe berguna untuk menentukan tipe nilai balik,
- Bagian_kode berupa pernyataan -pernyataan SQL

2. Perintah untuk merubah/alter

- Alter Table yaitu perintah yang digunakan untuk merubah struktur dari sebuah tabel

+ Menambah kolom baru

ALTER TABLE namatabel ADD fieldbaru typedata(lebar);

Contoh:

ALTER TABLE dosen ADD COLUMN telepon CHAR(15)
AFTER umur;

+ Mengubah Tipe data atau lebar kolom

ALTER TABLE namatabel MODIFY COLUMN field
type(lebar);

Contoh:

ALTER TABLE mahasiswa MODIFY COLUMN telepon(12);

+ Mengubah Nama Kolom

ALTER TABLE namatabel CHANGE COLUMN
namakolomlama namakolombaru typedatabaru(lebarbaru);

Contoh:

ALTER TABLE mahasiswa CHANGE COLUMN telepon phone
CHAR(25);

+ Menghapus kolom pada tabel

ALTER TABLE namatabel DROP COLUMN namakolom;

Contoh:

ALTER TABLE mahasiswa DROP COLUMN phone;

+ Menambah primary pada tabel

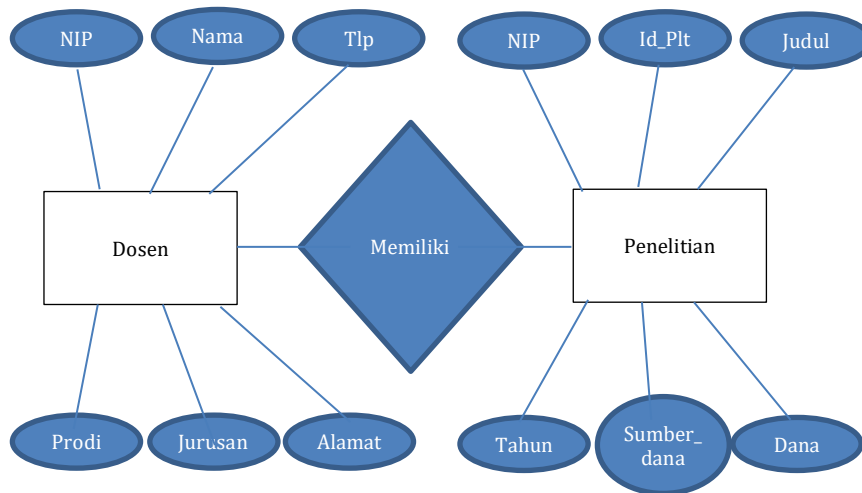
ALTER TABLE mahasiswa ADD CONSTRAINT
namaconstraint PRIMARY KEY(namakolom);

3. Perintah untuk menghapus/drop

- Drop database yaitu perintah yang berfungsi untuk menghapus database.
- Drop tabel yaitu perintah yang di gunakan untuk menghapus tabel pada database.

3.2 Contoh Studi Kasus

Pusat penelitian dan pengabdian kepada masyarakat (P3M) Poliban ingin memiliki database seluruh dosen yang telah melaksanakan penelitian, sehingga ketika informasi yang diperlukan tentang dosen Poliban telah melakukan penelitian apa saja maka dapat disajikan dengan mudah dan cepat. Berdasarkan deskripsi sederhana tersebut dibuatlah ERD sebagai berikut:



Gambar 3.1 ERD Info Penelitian dosen

Penyelesaian 1:

Dari ERD di atas maka dapat dibuat tabel yang diperlukan pada database sipma:

Use sipma;

Create table dosen (nip varchar (18) not null primary key, nama varchar(5), tlp varchar(13), prodi varchar(25), jurusan varchar(25), alamat varchar (5));

```
mysql> desc dosen;
```

Field	Type	Null	Key	Default	Extra
nip	varchar(18)	NO	PRI	NULL	
nama	varchar(5)	YES		NULL	
tlp	varchar(13)	YES		NULL	
prodi	varchar(25)	YES		NULL	
jurusan	varchar(25)	YES		NULL	
alamat	varchar(5)	YES		NULL	

6 rows in set (0.02 sec)

Create table Penelitian (nip varchar (18) NOT NULL, id_plt varchar(11), judul text, tahun varchar(4), sumber_dana varchar(25), dana float, FOREIGN KEY fk_nip(nip) REFERENCES dosen (nip));

```
Command Prompt - mysql.exe -uroot -p
```

```
-> ;
```

Field	Type	Null	Key	Default	Extra
nip	varchar(18)	NO	MUL	NULL	
id_plt	varchar(11)	YES		NULL	
judul	text	YES		NULL	
tahun	varchar(4)	YES		NULL	
sumber_dana	varchar(25)	YES		NULL	
dana	float	YES		NULL	

6 rows in set (0.30 sec)

Pada kesempatan kali ini kita perlu melengkapi **primary key** pada table Penelitian dikarenakan untuk membedakan penelitian yang satu dengan yang lain untuk setiap dosen yang berelasi, sehingga kita perlu menambah **primary key** pada table Penelitian untuk filed **id_plt**

```
ALTER TABLE Penelitian ADD CONSTRAINT pk_1 PRIMARY KEY(id_plt);
```

```
mysql> DESC PENELITIAN;
```

Field	Type	Null	Key	Default	Extra
nip	varchar(18)	NO	MUL	NULL	
id_plt	varchar(11)	NO	PRI		
judul	text	YES		NULL	
tahun	varchar(4)	YES		NULL	
sumber_dana	varchar(25)	YES		NULL	
dana	float	YES		NULL	

```
6 rows in set (0.01 sec)
```

Selanjutnya kita akan mengganti field **judul** pada table **penelitian** menjadi **judul_penelitian** dengan cara sebagai berikut:
ALTER TABLE penelitian CHANGE COLUMN judul judul_penelitian text;

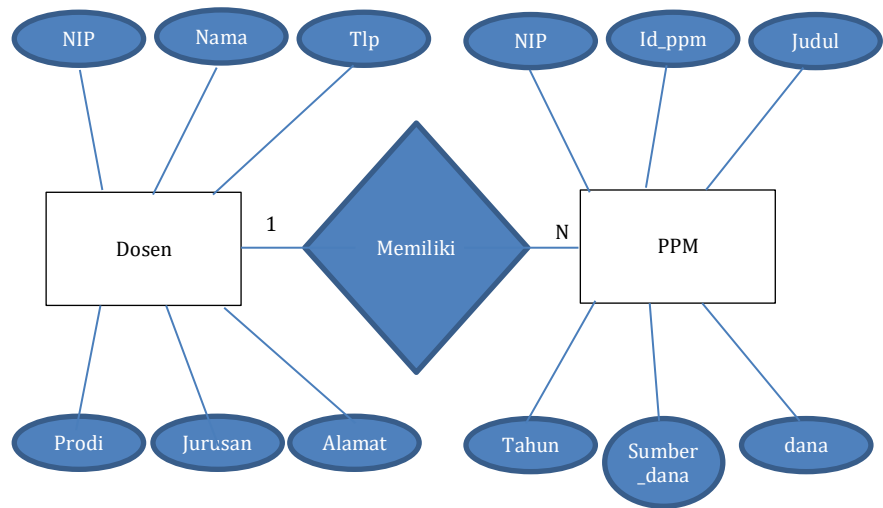
```
mysql> desc penelitian;
```

Field	Type	Null	Key	Default	Extra
nip	varchar(18)	NO	MUL	NULL	
id_plt	varchar(11)	NO	PRI		
judul_penelitian	text	YES		NULL	
tahun	varchar(4)	YES		NULL	
sumber_dana	varchar(25)	YES		NULL	
dana	float	YES		NULL	

```
6 rows in set (0.22 sec)
```

3.3 Soal Latihan

Dalam melengkapi tugas dosen di lingkungan Poliban, maka dosen juga harus melaksanakan pengabdian kepada masyarakat. P3M sebagai merupakan unit yang memiliki wewenang untuk mengumpulkan data dosen yang melaksanakan pengabdian baik dilaksanakan melalui dana hibah internal maupun hibah kompetisi yang diselenggarakan oleh ristekdikti. Pada kesempatan kali ini anda diminta untuk dapat menerjemahkan ERD yang menunjukkan bahwa dosen memiliki pengabdian kepada masyarakat ke dalam bahasa SQL khususnya DDL.



Gambar 3.2 ERD Info PPM dosen

Dari paparan deskripsi di atas, maka tabel apa yang perlu dibuat melanjutkan contoh studi kasus, sehingga informasi dosen memiliki data pengabdian kepada masyarakat dapat tersaji sebagai bentuk pelaksanaan tri dharma. Selanjutnya implementasikan pembuat tabel yang diperlukan menggunakan DDL pada MySQL.

BAB 4

DATA MANIPULATION LANGUAGE

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL (DML) pada DBMS MySql
2. Mampu menerjemahkan dan mengimplementasikan konsep Entity Relationship Diagram (ERD) ke dalam bentuk DML pada DBMS MySql

4.1 Data Manipulation Language (DML)

Bahasa manipulasi data (data manipulation language) merupakan bahasa yang digunakan untuk memanipulasi basis data. Manipulasi basis data dapat berupa menghapus (delete), memperbarui (update), memasukkan (insert), dan menarik informasi tertentu (select), atau Data Manipulation Language (DML) merupakan sub bahasa SQL yang digunakan untuk memanipulasi data dalam database yang telah terbuat.

1. Pengambilan informasi yang disimpan dalam basis data
2. Penempatan informasi baru dalam basis data
3. Penghapusan informasi dari basis data
4. Modifikasi informasi yang disimpan dalam basis data

Adapun perintah DML diantaranya:

- **INSERT**

Perintah Insert: Perintah ini digunakan untuk menyisipkan atau memasukkan data baru ke dalam tabel. Penggunaannya setelah database dan tabel selesai dibuat.

- Memasukkan data baru
Bentuk Umum:

```
Mysql> INSERT INTO nama_tabel VALUES (nilai_kolom1,  
nilai_kolom2,..);
```

Contoh

```
Mysql> INSERT INTO mahasiswa VALUES ('Irfan','E020317069');
```

- **SELECT**

Perintah Select ini digunakan untuk mengambil data atau menampilkan data dari satu tabel atau beberapa tabel dalam relasi. Data yang diambil dapat kita tampilkan dalam layar prompt MySQL secara langsung maupun ditampilkan pada tampilan aplikasi.

- Menampilkan data pada tabel
Bentuk Umum:

```
Mysql> SELECT * from nama_table
```

Contoh:

```
Mysql> SELECT * from Identitas;
```

- Menampilkan beberapa kolom saja
Bentuk Umum:

```
Mysql> SELECT nama_kolom1, nama_kolom2,... FROM nama_tabel;
```

Contoh:

```
Mysql> SELECT stb, alamat,... FROM Identitas;
```

- **UPDATE**

Perintah Update ini digunakan untuk memperbarui data lama menjadi data terkini. Jika Anda memiliki data yang salah atau kurang up to date dengan kondisi sekarang, maka dapat diubah isi datanya menggunakan perintah UPDATE.

Contoh:

- Modifikasi data dalam tabel
Bentuk Umum:

```
Mysql> UPDATE nama_tabel SET nama_kolom = nilaibaru WHERE  
nama_kolom = nilai;
```

Contoh:

```
Mysql> UPDATE Identitas SET nama ='Hasran' Where stb  
='13020150065';
```

- **DELETE**

Perintah Delete ini digunakan untuk menghapus data dari tabel. Biasanya data yang dihapus merupakan data yang sudah tidak diperlukan lagi. Pada saat menghapus data, perintah yang telah dijalankan tidak dapat digagalkan, sehingga data yang telah hilang tidak dapat dikembalikan lagi.

Contoh:

- Menghapus baris pada tabel
Bentuk Umum:

```
Mysql> DELETE FROM nama_tabel WHERE nama_kolom = nilai;
```

Contoh:

```
Mysql> DELETE FROM Identitas WHERE stb='13020130001';
```

4.2 Contoh Studi Kasus

Dosen

```
mysql> desc dosen;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nip   | varchar(18)   | NO   | PRI | NULL    |       |
| nama  | varchar(5)    | YES  |     | NULL    |       |
| tlp   | varchar(13)   | YES  |     | NULL    |       |
| prodi | varchar(25)   | YES  |     | NULL    |       |
| jurusan | varchar(25) | YES  |     | NULL    |       |
| alamat | varchar(5)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.02 sec)
```

Isian data Dosen

Berdasarkan struktur tabel di atas, maka kita dapat memasukkan data dosen dengan menggunakan perintah Insert, sebagai berikut:

```
Insert into dosen (nip,nama,tlp,prodi,jurusan,alamat)  
values ("198309172005011003", "Abdul Rozaq",  
"085251966688", "Manajemen Informatika", "Administrasi  
Bisnis", "Banjarmasin");
```

Sedangkan untuk menampilkan informasi dosen dapat dilakukan dengan perintah sebagai berikut:

```
Select * from dosen;
```

nip	nama	tlp	prodi	jurusan	alamat
197306202003011001	Hidayat	085251966789	Manajemen Informatika	Administrasi Bisnis	Banjarbaru
198101192004011003	Umi	085251912786	Manajemen Informatika	Administrasi Bisnis	Martapura
▶ 198309172005011002	Abdul Rozaq	085251966688	Manajemen Informatika	Administrasi Bisnis	Banjarmasin

Penelitian

```
mysql> desc penelitian;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nip            | varchar(18)   | NO   | MUL | NULL    |       |
| id_plt         | varchar(11)   | NO   | PRI |         |       |
| judul_penelitian | text          | YES  |     | NULL    |       |
| tahun         | varchar(4)    | YES  |     | NULL    |       |
| sumber_dana    | varchar(25)   | YES  |     | NULL    |       |
| dana          | float         | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.22 sec)
```

Isian data Penelitian

Selanjutnya untuk memasukan data penelitian seperti di bawah, dapat dilakukan dengan menggunakan perintah insert seperti memasukan data dosen di atas.

nip	id_plt	judul_penelitian	tahun	sumber_dana	dana
198309172005011002	p001	Sistem Informasi Penelitian dan Pengabdian Kepada Masyarakat	2018	DIPA Poliban	15000000
198101192004011003	p002	Mengukur Tingkat Kesiapan Sistem Informasi	2018	RISTEKDIKTI	20000000
198309172005011002	p003	AUDIT Mutu Internal	2019	DIPA Poliban	20000000

4.3 Soal Latihan

Berdasarkan ERD 3.1 yang tabelnya telah dibuat beserta isian datanya sesuai dengan contoh studi kasus, maka selanjutnya anda diminta untuk menampilkan informasi penelitian yang telah dilaksanakan oleh dosen pada tahun 2018 dengan menggunakan perintah select yang di dalamnya menampilkan hasil seperti di bawah ini:

nip	nama	judul_penelitian	tahun	sumber_dana
198309172005011002	Abdul Rozaq	Sistem Informasi Pen	2018	DIPA Poliban
198101192004011003	Umi	Mengukur Tingkat K	2018	RISTEKDIKTI

BAB 5

VIEW DAN INDEX

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL view dan index pada DBMS MySQL
2. Mampu mengimplementasikan pembuatan view dan index yang diperlukan pada DBMS MySQL

5.1 View dan Index

a. View

1. Pengertian View dan Implementasinya

View adalah objek di dalam Database yang berisi kumpulan kolom yang dihasilkan dari perintah **SELECT**. Dengan kata lain yang lebih sederhana, View adalah objek yang menyimpan hasil Query, baik dari satu tabel atau lebih, di dalam Database View juga dinamakan sebagai “Tabel Virtual”, karena View sebenarnya tidak memiliki data dan tabel ini bisa berasal dari tabel lain, atau gabungan dari beberapa tabel. Data yang ditampilkan oleh sebuah View diambil dari tabel-tabel aktual yang disertakan dalam **SELECT**.

Tujuan dari pembuatan view adalah untuk kenyamanan (mempermudah penulisan query), untuk keamanan (menyembunyikan beberapa kolom yang bersifat rahasia), atau dalam beberapa kasus bisa digunakan untuk mempercepat proses menampilkan data (terutama jika kita akan menjalankan query tersebut secara berulang).

2. Tipe View

Definisi View adalah hasil (**result**) dari sebuah query terhadap relasi-relasi dasar (atau relasi real). Hasil (**view**) ini tidak disimpan dalam database seperti relasi dasar. Sebuah view adalah sebuah jendela dinamik, dalam artian bahwa ia mencerminkan semua update yang dilakukan terhadap database. Disamping pemakainya di dalam skema eksternal view juga berguna untuk menjamin data – security dengan cara yang sederhana. Dengan memilih sub set dari database, view dapat menyembunyikan beberapa data. Jika user mengakses database melalui view, mereka tak dapat melihat atau memanipulasi hidden – data, dengan demikian data akan menjadi secure.

Dengan view kita dapat menerapkan pembatasan pada pengaksesan guna pengamanan keamanan data seperti dibawah ini:

- Kolom /field pada tabel tertentu.
- Baris/ Record pada tabel tertentu.
- Field dan record pada tabel tertentu.
- Turunan dari view lain.
- Record menggunakan operasi join.
- Data statistik dari tabel.

Sumber data view dapat berasal dari tabel atau view lain. Mirip dengan tabel, anda dapat melakukan update, delete dan **INSERT** pada **VIEW** sehingga perubahan itu akan direpleksikan pada base tabel nya. Berbeda dengan tabel, view tidak menyimpan data, view hanya menyimpan definisi query pada data dictionary dan tidak memerlukan ruang penyimpanan data. Penerapan view dapat diaplikasikan pada situasi berikut:

- Membatasi akses sesuai otoritas user.
- Memudahkan pemahaman terhadap kolom penampung data yang mungkin berbeda dengan definisi kolom pada tabel dasar.
- Menyederhanakan pandangan user terhadap data.
- Menangani data kompleks.
- Memudahkan penggunaan query yang berulang karena disimpan sebagai stored query.

3. Perintah Membuat Tabel dan Tabel Virtual/ view

```
SELECT (namafield) as (namafield yang ingin ditampilkan), as....  
FROM (nama tabel)  
WHERE (namafield) = '(data yang ingin kita tampilkan saja)';
```

Terkait user yang memiliki wewenang dalam pembuatan view, maka perlu diatur hak akses user tersebut dengan cara sebagai berikut:

```
GRANT create, alter, drop, create view, select ON sipma.* TO  
'super'@'%' IDENTIFIED BY 'super123';
```

```
CREATE VIEW (Nama Tabel View Yang kita inginkan) as  
SELECT (namafield) as (namafield yang ingin ditampilkan), as....  
FROM (nama tabel)  
WHERE (namafield) = '(data yang ingin kita tampilkan saja)';
```

4. Menghapus View

```
Drop View (Nama Tabel View);
```

5. Perintah **SELECT** untuk mengetahui apakah tabel view berfungsi

```
SELECT*from (nama tabel view);
```

b. Index

1. Pengertian Index

Index adalah sebuah objek dalam system database yang dapat mempercepat proses pencarian (**query**) data. Saat database dibuat tanpa menggunakan index, maka kinerja server database dapat menurun secara drastis.

Alasan mengapa index diperlukan:

- 1) Kolom sering digunakan dalam klausa Where
- 2) Kolom berisi nilai jangkauan yang luas
- 3) Kolom berisi banyak nilai null

- 4) Tabel berukuran besar dan sebagian besar query menampilkan data kurang dari 2-4%

2. Perintah melihat tabel Index

```
Show index from (nama Table);
```

3. Menambahkan Index pada Tabel

```
ALTER TABLE (nama tabel) add (nama field);  
CREATE INDEX IDX (namafield) on (nama tabel) (namafield);
```

```

C:\> Select C:\Windows\system32\cmd.exe - mysql -u server -p

mysql> CREATE INDEX nama_idx ON siswa (nama_siswa);
Query OK, 3 rows affected (0.28 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> show index from siswa;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| siswa | 0 | PRIMARY | 1 | Nomor_Siswa | A | 3 | NULL | NULL | | BTREE | |
| siswa | 1 | nama_idx | 1 | Nama_Siswa | A | 3 | NULL | NULL | | BTREE | |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

5.2 Contoh Studi Kasus

Dosen

nip	nama	tlp	prodi	jurusan	alamat
197306202003011001	Hidayat	085251966789	Manajemen Informatika	Administrasi Bisnis	Banjarbaru
198101192004011003	Umi	085251912786	Manajemen Informatika	Administrasi Bisnis	Martapura
▶ 198309172005011002	Abdul Rozaq	085251966688	Manajemen Informatika	Administrasi Bisnis	Banjarmasin

Penelitian

nip	id_plt	judul_penelitian	tahun	sumber_dana	dana
198309172005011002	p001	Sistem Informasi Penelitian dan Pengabdian Kepada Masyarakat	2018	DIPA Poliban	15000000
▶ 198101192004011003	p002	Mengukur Tingkat Kesiapan Sistem Informasi	2018	RISTEKDIKTI	20000000
198309172005011002	p003	AUDIT Mutu Internal	2019	DIPA Poliban	20000000

Berdasarkan informasi yang terlihat pada tabel dosen dan penelitian, maka selanjutnya kita diminta untuk menampilkan info_penelitian_dosen yang dieksekusi melalui pembuatan view dengan kondisi yang harus dipenuhi seperti pada tabel di bawah ini:

nip	nama	judul_penelitian	tahun	sumber_dana	dana
198101192004011003	Umi	Mengukur Tingkat Kesiapan Sistem Informasi	2018	RISTEKDIKTI	20000000

Adapun perintah view yang diperlukan untuk menampilkan info seperti tabel di atas adalah sebagai berikut:

```
Create view v_info_penelitian_ristekdikti as select dosen.nip AS nip,dosen.nama AS nama, penelitian.judul_penelitian AS judul_penelitian,penelitian.tahun AS tahun, penelitian.sumber_dana AS sumber_dana, penelitian.dana AS dana from (dosen join penelitian on((dosen.nip = penelitian.nip))) where ((penelitian.tahun = 2018) and (penelitian.sumber_dana = 'RISTEKDIKTI'));
```

Sedangkan untuk memanggil view menggunakan perintah:

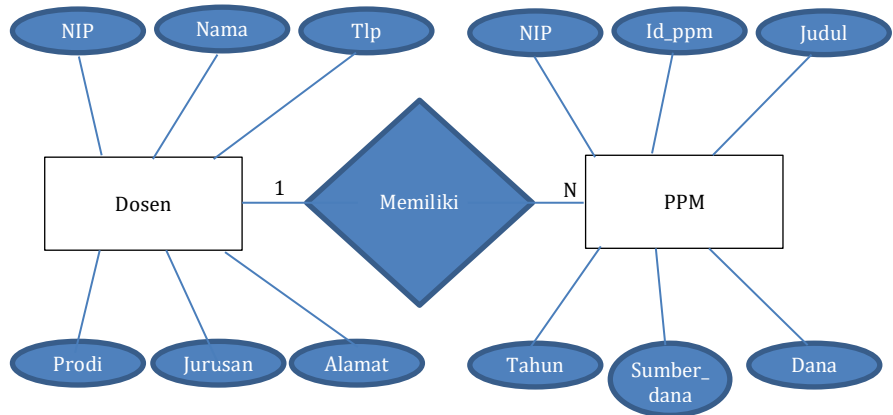
```
Select * from v_info_penelitian_ristekdikti;
```

```
mysql> select * from v_info_penelitian_ristekdikti;
+-----+-----+-----+-----+-----+-----+
| nip          | nama | judul_penelitian | tahun | sumber_dana | dana |
+-----+-----+-----+-----+-----+-----+
| 198101192004011003 | Umi  | Mengukur Tingkat Kesiapan Sistem Informasi | 2018 | RISTEKDIKTI | 20000000 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

5.3 Soal Latihan

Pada ERD Dosen Memiliki Pengabdian kepada masyarakat seperti gambar di bawah ini:



Anda diminta untuk membuat view semua info_pengabdian_dosen dan info_pengabdian_dosen_ristekdikti yang dibiaya oleh ristekdikti.

BAB 6

TRIGGER

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL Trigger pada DBMS MySql
2. Mampu mengimplementasikan pembuatan trigger yang diperlukan pada DBMS MySql

6.1 Trigger

Trigger adalah objek database yang merupakan aksi yang terjadi jika terjadi perubahan pada suatu *row*. Triggers merupakan kumpulan pernyataan SQL yang dibuat untuk dieksekusi oleh pernyataan INSERT, UPDATE, atau DELETE, dengan tujuan untuk menjaga konsistensi data.

Triggers dibuat dengan menggunakan pernyataan **CREATE TRIGGER**. Sintaksnya:

```
CREATE  
[DEFINER = { user | CURRENT_USER }]  
TRIGGER trigger_name trigger_time trigger_event  
ON tbl_name FOR EACH ROW trigger_body
```

Keterangan:

- **trigger_name**: nama trigger.
- **trigger_time**: kapan kita mengeksekusi trigger, apakah sebelum atau sesudah perubahan pada row data tabel. Jadi pilihannya adalah **AFTER** atau **BEFORE**.

- **trigger_event:** merupakan event atau peristiwa yang menyebabkan trigger dilakukan. Pilihan event tersebut adalah **INSERT, UPDATE, DELETE**.
- **tbl_name:** nama tabel.
- **trigger_body:** *statement-statement* perintah SQL yang akan dilakukan. Jika perintahnya lebih dari satu maka gunakan dalam blok statement **BEGIN... END**.
- Jika **DEFINER** dispesifikasikan maka kita memutuskan trigger tersebut dijalankan hanya oleh user tertentu (dalam format penulisan **user@host**). Jika tidak dispesifikasikan, maka user yang melakukan perubahan (**CURRENT_USER**) adalah pilihan *default*.

Pada trigger terdapat alias dengan nama OLD dan NEW.

Penggunaannya:

NEW.nama_kolom

OLD.nama_kolom

Keterangan:

- **OLD**.nama_kolom menyatakan nilai nama_kolom sebelum dihapus atau diubah.
- **NEW**.nama_kolom menyatakan nilai nama_kolom setelah diubah atau setelah data baru dimasukkan.

6.2 Contoh Studi Kasus

Berikut contoh penggunaan trigger untuk event setelah penghapusan (**AFTER DELETE**) pada table "**tr_penjualan**"- database **toko**. Langkah yang akan kita lakukan adalah sebagai berikut:

1. Buat tabel log monitor dengan nama "**tr_penjualan_hapus**" yang berisi baris data yang dihapus dari table "**tr_penjualan**" dengan tambahan dua field, yaitu tanggal penghapusan (**tgl_perubahan**) dan user MySQL yang melakukan hal tersebut (**nama_user**).

Berikut tahapannya:

```

CREATE DATABASE toko;
USE toko;
CREATE TABLE tr_penjualan (tgl_transaksi datetime,
kode_cabang varchar(15), kode_kasir varchar(15), kode_item
varchar(10), kode_barang varchar(15), jumlah int);
CREATE TABLE `tr_penjualan_hapus` LIKE `tr_penjualan`;

```

Tabel **tr_penjualan** sudah ada di database toko, sehingga dalam pembuatan table **tr_penjualan_hapus** mengambil struktur table dari **tr_penjualan** menggunakan perintah di atas. Selanjutnya kita akan menambah dua field dengan cara menggunakan **Alter** seperti perintah di bawah ini:

```

ALTER TABLE `tr_penjualan_hapus` ADD
(tgl_perubahan` DATETIME, `nama_user` VARCHAR(200));

```

2. Selanjutnya membuat trigger yang akan melakukan penyimpanan data yang dihapus dari "tr_penjualan" ke table "tr_penjualan_hapus".

Berikut adalah perintahnya:

```

DELIMITER |
CREATE TRIGGER hapus_tr_penjualan AFTER DELETE
ON tr_penjualan FOR EACH ROW
BEGIN
INSERT INTO tr_penjualan_hapus
(tgl_transaksi, kode_cabang, kode_kasir, kode_item,
kode_produk, jumlah_pembelian, tgl_perubahan, nama_user)
VALUES
(OLD.tgl_transaksi, OLD.kode_cabang, OLD.kode_kasir,
OLD.kode_item, OLD.kode_produk,OLD.jumlah_pembelian,
SYSDATE(),CURRENT_USER);END;|
DELIMITER;

```

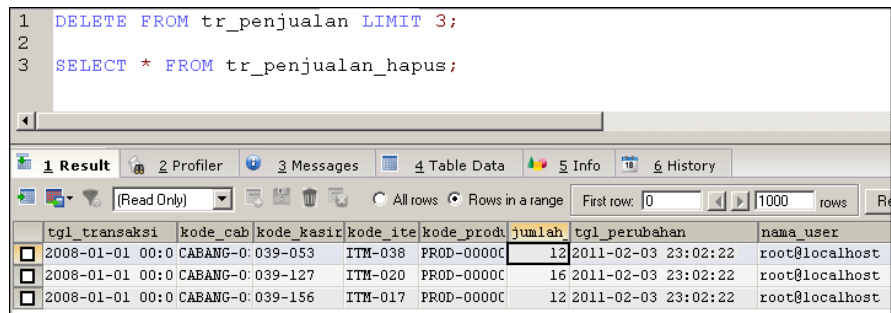
3. Setelah trigger dibuat, selanjutnya kita melakukan pengujian. Dengan menghapus tiga baris data dari table "**tr_penjualan**" dan kita bisa cek apakah trigger yang sudah dibuat berfungsi

sesuai dengan kode program yang telah dibuat dengan menyimpan baris data pada table "**tr_penjualan_hapus**".

Jalankan perintah berikut:

```
DELETE FROM tr_penjualan LIMIT 3;
```

```
SELECT * FROM tr_penjualan_hapus;
```



The screenshot shows a database management tool interface. The top panel displays three SQL commands: 1. `DELETE FROM tr_penjualan LIMIT 3;`, 2. (blank), and 3. `SELECT * FROM tr_penjualan_hapus;`. The bottom panel shows the results of the third command, a table with 8 columns: `tgl_transaksi`, `kode_cab`, `kode_kasir`, `kode_ite`, `kode_prodi`, `jumlah`, `tgl_perubahan`, and `nama_user`. Three rows are visible, each with a checkbox on the left, indicating they have been deleted from the original table and moved to the `tr_penjualan_hapus` table.

	tgl_transaksi	kode_cab	kode_kasir	kode_ite	kode_prodi	jumlah	tgl_perubahan	nama_user
<input type="checkbox"/>	2008-01-01 00:0	CABANG-0	039-053	ITM-038	PROD-0000C	12	2011-02-03 23:02:22	root@localhost
<input type="checkbox"/>	2008-01-01 00:0	CABANG-0	039-127	ITM-020	PROD-0000C	16	2011-02-03 23:02:22	root@localhost
<input type="checkbox"/>	2008-01-01 00:0	CABANG-0	039-156	ITM-017	PROD-0000C	12	2011-02-03 23:02:22	root@localhost

Terlihat pada gambar di atas 3 row yang dihapus telah "pindah" ke table "**tr_penjualan_hapus**" dengan tambahan informasi waktu penghapusan dan user yang menghapus.

6.3 Soal Latihan

Pada database sipma terdapat data yang disimpan pada tabel penelitian, pada kesempatan kali ini anda diminta untuk membuat trigger dengan ketentuan:

1. Trigger dibuat dengan nama **log_update_penelitian**
2. Trigger berfungsi untuk menyimpan baris data yang di update, beserta tanggal perubahan dan user yang melakukan perubahan.
3. Trigger dieksekusi setelah (after) dilakukan update pada tabel penelitian.

BAB 7

STORE PROSEDURE

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL Store Procedure pada DBMS MySQL
2. Mampu mengimplementasikan pembuatan Store Procedure yang diperlukan pada DBMS MySQL

7.1 Store Procedure

Stored Procedure merupakan sekumpulan perintah SQL yang diberi nama dan disimpan di database server. Jika kita menggunakan DBMS MySQL, maka *store procedure* merupakan salah satu objek *routine* yang tersimpan pada database MySQL dan dapat digunakan untuk menggantikan berbagai kumpulan perintah yang sering kita perlukan dalam menyelesaikan setiap studi kasus, seperti misalkan menampilkan sejumlah row dari beberapa tabel dengan relasi, filter data dan di dalamnya terdapat logika If.

```
CREATE  
[DEFINER = { user | CURRENT_USER }]  
PROCEDURE nama_sp (parameter-sp[...])  
[characteristic...] routine_body
```

Keterangan:

- **Nama_sp**: nama stored procedure.
- **Parameter_sp**: parameter input / output dari stored procedure tersebut (opsional).

- **characteristic:** menjelaskan karakteristik dari stored procedure (COMMENT, LANGUAGE SQL, dan lain-lain).
- **routine_body:** kumpulan perintah pada stored procedure tersebut.
- Jika **DEFINER** dispesifikasikan maka kita memutuskan trigger tersebut dijalankan hanya oleh user tertentu (dalam format penulisan **user@host**). Jika tidak dispesifikasikan, maka user yang melakukan perubahan (**CURRENT_USER**) adalah pilihan *default*.

Untuk menjalankan *store procedure* yang telah dibuat, dapat digunakan perintah CALL (beberapa DBMS ada yang menggunakan EXECUTE).

Kelebihan stored procedure diantaranya meningkatkan performa, mereduksi trafik jaringan, reusable, dan meningkatkan kontrol sekuriti. Sedangkan kekurangannya yaitu berpotensi meningkatkan beban server dan penulisnya tidak mudah (memerlukan pengetahuan yang spesifik).

7.2 Contoh Studi Kasus

Pada contoh studi kasus pembuatan store procedure kali ini kita tetap menggunakan tabel yang sebelumnya sudah dibuat di dalam database **sipma**. Adapun studi kasusnya yaitu:

Anda diminta membuat store procedure sederhana menampilkan data dosen dengan nama **tampil_data_dosen** dan store procedure menambah data dosen menggunakan parameter dengan nama **tambah_dosen**.

1. Menampilkan data dosen dengan perintah **select**

```
mysql> select * from dosen;
```

nip	nama	tlp	prodi	jurusan	alamat
197306202003011001	Hidayat	085251966789	Manajemen Informatika	Administrasi Bisnis	Banjarbaru
198007212006011002	Sakha	085251688899	Teknik Informatika	Teknik Elektro	Banjarmasin
198101102004011003	Umi	085251912786	Manajemen Informatika	Administrasi Bisnis	Martapura
198309172005011002	Abdul Rozaq	085251966688	Manajemen Informatika	Administrasi Bisnis	Banjarmasin

```
4 rows in set (0.00 sec)
mysql>
```

Gambar 7.1 Menampilkan data dosen dengan menggunakan perintah **select**

2. Membuat store procedure **tampil_data_dosen**

```
mysql> delimiter //
mysql> create procedure tampil_data_dosen()
-> begin
-> select * from dosen where prodi="Manajemen Informatika";
-> end //
Query OK, 0 rows affected (0.21 sec)
```

Gambar 7.2 Membuat store procedure **tampil_data_dosen**

3. Memanggil store procedure **tampil_data_dosen**

```
mysql> call tampil_data_dosen;
-> //
+-----+-----+-----+-----+-----+-----+
| nip      | nama      | tlp      | prodi      | jurusan      | alamat      |
+-----+-----+-----+-----+-----+-----+
| 197306202003011001 | Hidayat   | 085251966789 | Manajemen Informatika | Administrasi Bisnis | Banjarbaru |
| 198101192004011003 | Umi       | 085251912786 | Manajemen Informatika | Administrasi Bisnis | Martapura  |
| 198309172005011002 | Abdul Rozaq | 085251966688 | Manajemen Informatika | Administrasi Bisnis | Banjarmasin |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.11 sec)
Query OK, 0 rows affected (0.15 sec)
mysql>
```

Gambar 7.3 Memanggil store procedure **tampil_data_dosen** dengan perintah **Call**

4. Membuat store procedure **tambah_dosen**

```
mysql> create procedure tambah_dosen
-> (p_nip varchar(20),p_nama varchar(25),p_tlp varchar(13)
-> ,p_prodi varchar(25),p_jurusan varchar(25),p_alamat varchar(50))
-> begin
-> insert into dosen
-> values(p_nip,p_nama,p_tlp,p_prodi,p_jurusan,p_alamat);
-> end //
Query OK, 0 rows affected (0.13 sec)
mysql> delimiter ;
```

Gambar 7.4 Membuat store procedure **tambah_dosen**

5. Memanggil store procedure **tambah_dosen**

```
mysql> call tambah_dosen
-> ("1973091719980200201", "Madina", "081356776578", "Otomotif", "Teknik Mesin", "Banjarasin");
Query OK, 1 row affected, 1 warning (0.17 sec)

mysql> select * from dosen;
```

nip	nama	tlp	prodi	jurusan	alamat
197306202003011001	Hidayat	085251966789	Manajemen Informatika	Administrasi Bisnis	Banjarbaru
197309171998020020	Madina	081356776578	Otomotif	Teknik Mesin	Banjarasin
198007212006011002	Sakha	085251688899	Teknik Informatika	Teknik Elektro	Banjarasin
198101192004011003	Umi	085251912786	Manajemen Informatika	Administrasi Bisnis	Martapura
198309172005011002	Abdul Rozaq	085251966688	Manajemen Informatika	Administrasi Bisnis	Banjarasin

```
5 rows in set (0.00 sec)

mysql>
```

Gambar 7.5 Memanggil store procedure **tambah_dosen**

6. Contoh lain

Membuat store procedure untuk menampilkan pada semester tertentu dan dengan sks yang diinput oleh user.

```
DELIMITER //
CREATE PROCEDURE getMkBySemSks(
    IN Smt INT(2),
    IN Sks INT(2))
BEGIN
    SELECT * FROM matakuliah
    WHERE semester = Smt
    AND sks = Sks ;
END //
DELIMITER;
```

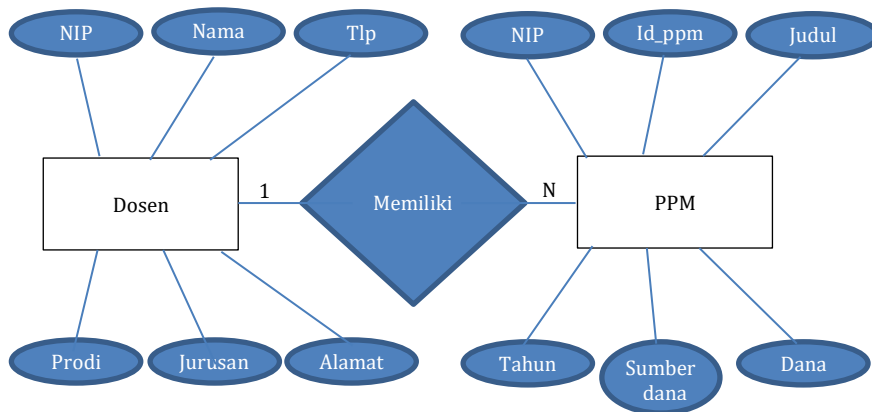
Pemanggilan stored procedure di atas tentunya akan memerlukan dua buah argumen.

```
CALL getMkBySemSks(3,2);
```

kd_mk	nama_mk	sks	semester	kode_dos
PTI777	Sistem Informasi	2	3	99

7.3 Soal Latihan

Pada ERD Dosen Memiliki Pengabdian kepada masyarakat seperti gambar di bawah ini:



Gambar 7.6 ERD Dosen memiliki pengabdian pada masyarakat

Pada kesempatan kali ini anda diminta untuk:

1. Membuat store procedure untuk menampilkan dosen (**p_dosen_ppm**) memiliki pengabdian kepada masyarakat yang mengambil dari dua table yang berelasi seperti terlihat pada erd di atas dan difilter hanya tahun tertentu sesuai inputan user!
2. Membuat store procedure untuk memasukan data pengabdian pada masyarakat (**p_input_ppm**) dengan menggunakan parameter input

BAB 8

FUNCTION

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL function pada DBMS MySQL
2. Mampu mengimplementasikan pembuatan function yang diperlukan pada DBMS MySQL

8.1 Function

FUNCTION adalah feature yang terdapat di Mysql 5.0 yang merupakan kumpulan-kumpulan SQL berupa routine yang di simpan dalam database MySQL Server. Biasanya function ini dikombinasikan dengan *store pocedure* atau bisa juga tidak di kombinasikan dengan store procedure

Function merupakan Statement kode SQL yang akan mengembalikan sebuah nilai balik pada pemanggilnya. Nilai yang dihasilkan Function harus ditampung kedalam sebuah variabel.

Sedangkan Perbedaan antara function dengan store procedure, yaitu:

Function akan mengembalikan suatu nilai pada pemanggilnya, sedangkan store procedure tidak mengembalikan nilai apapun pada fungsi pemanggilnya.

Membuat Function

```
Create FUNCTION nama_function ([parm_fung [...]])  
RETURNS tipe  
bagian_kode
```

Keterangan:

- Nama_ft menyatakan nama fungsi

- Param_fung menyatakan definisi untuk parameter fungsi
- Returns tipe berguna untuk menentukan tipe nilai balik
- Bagian_kode berupa pernyataan-pernyataan SQL

```
mysql> delimiter //
mysql> create function test (kata varchar (20), jumlah int)
-> returns varchar (30)
-> begin
-> return repeat (kata,jumlah);
-> end
-> //
Query OK, 0 rows affected (0.13 sec)

mysql> DELIMITER ;
mysql>
```

8.2 Contoh Studi Kasus

Contoh dibawah ini kasus tentang sistem informasi kampus, dengan tujuan membuat function yang akan mengembalikan nilai berupa jumlah mahasiswa dari setiap prodi.

Struktur tabel mahasiswa

```
CREATE TABLE mahasiswa (`nim` char(10) NOT NULL,`nama`
varchar(255) NOT NULL,`prodi` int(25) NOT NULL)
ENGINE=MyISAM DEFAULT CHARSET=latin1
```

Function untuk mengembalikan jumlah data dari setiap kelas

```
DELIMITER $$
CREATE FUNCTION f_tampil_mhs_prodi (p_prodi int) RETURNS
INT DETERMINISTIC
BEGIN
DECLARE jml INT;
SELECT COUNT(*) AS jml_prodi INTO jml FROM mahasiswa
WHERE
prodi = p_prodi;
RETURN jml;
```

END\$\$

DELIMITER ;

Penjelasannya sebagai berikut:

- **DELIMITER** = adalah untuk memberi tahu kepada mysql soal delimiter yang digunakan, secara default menggunakan ; jadi bila ada tanda ; mysql akan mengartikan akhir dari statement, pada contoh di atas delimiter yang digunakan \$\$ jadi akhir statementnya adalah \$\$
- **CREATE FUNCTION** = adalah header untuk membuat function
- **RETURNS** = adalah untuk menentukan tipe data yang di returnkan oleh function
- **DETERMINISTIC/ NOT DETERMINISTIC** = adalah untuk menentukan yang bisa menggunakan function ini adalah user pembuatnya saja (determinisric) atau user siapa saja (not determinisric).
- **BEGIN END** = adalah body dari function jadi semua SQL nya di tulis disini.

Cara pemanggilannya seperti dibawah ini:

```
select sf_tampil_mhs_prodi("2");
```

Sebuah function hanya bisa memberikan return berupa nilai saja dan tidak bisa berupa resultset, sedangkan untuk penulisan **DETEMINISTIC** bisa ditulis secara implisit dengan memberikan setting global pada mysql dan secara default benilai **NOT DETEMINISTIC**, caranya dibawah ini:

```
SET GLOBAL log_bin_trust_function_creators = 1;
```

Contoh Penggunaan function dengan menggunakan perintah case pada kasus diskon


```

* DELIMITER //
* CREATE FUNCTION getDiskon(jumlah INT)
* RETURNS int(11)
* BEGIN
*   DECLARE diskon INT; CASE
*     WHEN (jumlah >= 100) THEN SET
*       diskon = 10;
*     WHEN (jumlah >= 50 AND jumlah < 100) THEN SET
*       diskon = 5;
*     WHEN (jumlah >= 20 AND jumlah < 50) THEN SET
*       diskon = 3;
*     ELSE SET diskon = 0; END CASE;
*   RETURN diskon;
* END//

```

Cara memanggil function getdiskon


Select getdiskon("80");

Maka hasilnya mendapatkan diskon 5.

Latihan:

Pada kesempatan kali ini kita akan membuat function dari studi kasus di kampus dalam menampilkan nama mahasiswa, menghitung IP, dan bobot penilaian. Untuk tahap pertama akan kita buat dulu database **kampusku** yang di dalamnya terdapat beberapa tabel dengan struktur tabel yang dapat dilihat di bawah ini:

Struktur Tabel **Dosen**

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	<input type="checkbox"/>	 1
NIK	varchar	15	0	<input type="checkbox"/>	
nama	varchar	45	0	<input type="checkbox"/>	

Isian Data Tabel Dosen

id	NIK	nama
1	033231212	Sinta
2	033431113	Bagus
3	039834314	Wahyu
4	039824515	Adjie

Struktur Tabel Jurusan

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	<input type="checkbox"/>	1
code	varchar	10	0	<input type="checkbox"/>	
nama	varchar	255	0	<input type="checkbox"/>	
jenjang	varchar	10	0	<input type="checkbox"/>	

Isian Data Tabel Jurusan

id	code	nama	jenjang
1	TS	Teknik Sipil	DIII
2	TM	Teknik Mesin	DIII
3	TE	Teknik Elektro	DIII
4	AK	Akuntansi	DIII
5	AB	Administrasi Bisnis/MI	DIII


Struktur Tabel Kuliah

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	<input type="checkbox"/>	1
mahasiswa_id	int	11	0	<input checked="" type="checkbox"/>	
matakuliah_id	int	11	0	<input checked="" type="checkbox"/>	
dosen_id	int	11	0	<input checked="" type="checkbox"/>	
nilai	varchar	2	0	<input checked="" type="checkbox"/>	

Isian Data Tabel Kuliah

id	mahasiswa_id	matakuliah_id	dosen_id	nilai
1	2	1	1	A-
2	2	2	2	B+
3	2	3	3	B
4	2	4	4	B
5	2	5	5	B
6	2	6	6	C-
7	2	7	7	B
8	2	8	8	B
9	2	9	9	B
10	2	10	10	A
11	2	11	11	A-
12	2	12	12	B+
13	8	1	1	B
14	8	2	2	B+
15	8	3	3	B
16	8	4	4	C
17	8	5	5	D+
18	8	6	6	B
19	8	7	7	C-
20	8	8	8	A
21	8	9	9	A
22	8	10	10	B
23	8	11	11	B
24	8	12	12	B
25	5	1	1	B
26	5	2	2	A

Struktur Tabel Mahasiswa

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	<input type="checkbox"/>	 1
nim	int	11	0	<input type="checkbox"/>	
nama	varchar	100	0	<input type="checkbox"/>	
tgl_lahir	date	0	0	<input type="checkbox"/>	
alamat	text	0	0	<input type="checkbox"/>	
jurusan_id	int	11	0	<input type="checkbox"/>	

Isian Data Tabel Mahasiswa

id	nim	nama	tgl_lahir	alamat	jurusan_id
1	1703130060	Yudi	1988-11-11	Sultan Adam, Banjarmasin	1
2	1703130061	Irfan	1988-12-12	Hasan Basri, Banjarmasin	1
3	1703130062	Erik	1988-11-11	Landasan Ulin, Banjarbaru	1
5	1703130063	Susi	1993-09-12	Liang Anggang, Banjarbaru	1
6	1703130064	Rendi	1998-02-12	Teluk Dalam, Banjarmasin	1
7	1703130065	Asep	1991-02-01	Sungai Jingah, Banjarmasin	1
8	1703130066	Devi	1989-02-16	Balitra, Banjarbaru	1

Struktur Tabel Matakuliah

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	<input type="checkbox"/>	1
nama	varchar	45	0	<input type="checkbox"/>	
sks	int	11	0	<input type="checkbox"/>	
jurusan_id	int	11	0	<input type="checkbox"/>	

Isian Data Tabel Matakuliah

id	nama	sks	jurusan_id
1	Algoritma Pemrograman	3	5
2	Sistem Basis Data	3	5
3	Agama	2	5
4	Bahasa Indonesia	2	5
5	Praktek Sistem Basis Data	3	5
6	Pemrograman Dasar	3	5
7	Praktek Pemrograman Dasar	2	5
8	Aplikasi Perkantoran	2	5
9	Sistem Operasi	2	5
10	Pemrograma Visual	3	5
11	Pemrograman Web	3	5
12	BAHASA INGGRIS 1	3	5

Semua tabel di atas kita create, kemudian kita insert datanya sesuai dengan isian data di masing-masing tabel.

1. Function untuk cari nama berdasarkan nimnya:

```
USE kampusku;
DROP function IF EXISTS TAMPILNAMA_MHS;

DELIMITER $$
USE kampusku$$
CREATE DEFINER=root@localhost FUNCTION
TAMPILNAMA_MHS(NOMHS VARCHAR(30)) RETURNS
varchar(30) CHARSET latin1
BEGIN
    DECLARE NMAA VARCHAR(30);
    SELECT nama FROM mahasiswa WHERE nim = NOMHS
LIMIT 1 INTO NMAA;
    RETURN NMAA;

END$$
DELIMITER;

Memanggil function TAMPILNAMA_MHS:

select TAMPILNAMA_MHS('1703130064');
```

2. Function ip untuk mahasiswa:

```
USE kampusKU;
DROP function IF EXISTS IP_MHS;

DELIMITER $$
USE kampusku$$
CREATE DEFINER=root@localhost FUNCTION IP_MHS(NOMHS
CHAR(9)) RETURNS decimal(5,2)
BEGIN
    DECLARE IP FLOAT;
SELECT
SUM(CASE
    WHEN NILAI = 'A' THEN 4 * b.sks
    WHEN NILAI = 'A-' THEN 3.7 * b.sks
    WHEN NILAI = 'B+' THEN 3.4 * b.sks
```

```

        WHEN NILAI = 'B' THEN 3 * b.sks
        WHEN NILAI = 'B-' THEN 2.7 * b.sks
        WHEN NILAI = 'C+' THEN 2.4 * b.sks
        WHEN NILAI = 'C' THEN 2 * b.sks
        WHEN NILAI = 'C-' THEN 1.7 * b.sks
        WHEN NILAI = 'D+' THEN 1.4 * b.sks
        WHEN NILAI = 'D' THEN 1 * b.sks
        ELSE 0 * b.sks
    END) / SUM(b.sks) AS POINT
FROM
    kuliah a,
    matakuliah b,
    mahasiswa c
WHERE
    a.matakuliah_id = b.id AND a.mahasiswa_id = c.id AND c.nim =
NOMHS
GROUP BY c.nim INTO IP;
RETURN IP;
    END$$
DELIMITER ;
Memanggil function IP_MHS:
    select IP_MHS('1703130064');

```

3. Function BOBOT_NILAI_MHS:

```

USE kampusku;
DROP function IF EXISTS BOBOT_NILAI_MHS;
DELIMITER $$
USE kampusku$$
CREATE DEFINER=root@localhost FUNCTION BOBOT_NILAI_MHS
(NIL CHAR(2), JSKS INT) RETURNS decimal(5,2)
BEGIN
    IF NIL='A' THEN
        RETURN 4*JSKS;

```

```
END IF;
    IF NIL='A-' THEN
RETURN 3.4*JSKS;
END IF;
IF NIL='B+' THEN
RETURN 3.7*JSKS;
END IF;
IF NIL='B' THEN
RETURN 3*JSKS;
END IF;
IF NIL='B-' THEN
RETURN 2.7*JSKS;
END IF;
IF NIL='C+' THEN
RETURN 2.4*JSKS;
END IF;
IF NIL='C' THEN
RETURN 2*JSKS;
END IF;
IF NIL='C-' THEN
RETURN 1.7*JSKS;
END IF;
IF NIL='D+' THEN
RETURN 1.4*JSKS;
END IF;
IF NIL='D' THEN
RETURN 1*JSKS;
END IF;
IF NIL='E' THEN
RETURN 0*JSKS;
END IF;
END$$
DELIMITER ;
```

Memanggil function BOBOT_NILAI_MHS:

```
select BOBOT_NILAI_MHS ('A',2);
```

A adalah nilai huruf dan 2 merupakan SKS mata kuliahnya, setelah itu kita gabungkan semua function ke dalam store procedure:

```
USE kampusku;
DROP procedure IF EXISTS NILAI_PERKULIAHAN;
DELIMITER $$
USE kampusku$$
CREATE DEFINER=root@localhost PROCEDURE
NILAI_PERKULIAHAN (in NIMNYA int(11))
BEGIN
select a.* from (
SELECT nama matakuliah_and_nama, nim nim_and_sks, '-' nilainya, '-'
rata_rata
from mahasiswa where nim = NIMNYA
UNION ALL
select "matakuliah_and_nama,"nim_and_sks,"nilainya, "rata_rata
UNION ALL
SELECT
    b.nama matakuliah_and_nama,
    b.sks nim_and_sks,
    a.nilai nilainya,
BOBOT_NILAI_MHS (a.nilai, b.sks) rata_rata
FROM
    kuliah a,
    matakuliah b,
    mahasiswa c
WHERE
    a.matakuliah_id = b.id
        AND a.mahasiswa_id = c.id
        AND c.nim = NIMNYA
UNION ALL
select "matakuliah_and_nama,"nim_and_sks,"nilainya, "rata_rata
```



```
UNION ALL
select "matakuliah_and_nama,"nim_and_sks,'IP ='nilainya, IP_MHS
(NIMNYA)rata_rata
) a;
END$$
DELIMITER ;
```

Cara memanggil store procedure **NILAI_PERKULIAHAN** yang menggabungkan function **BOBOT_NILAI_MHS** dan **IP_MHS**

```
call NILAI_PERKULIAHAN('1703130064');
```

8.3 Soal Latihan

1. Pada kasus dosen memiliki karya penelitian, maka pada kesempatan kali ini anda diminta untuk membuat function yang bertujuan untuk dapat menghitung jumlah karya penelitian dari seorang dosen!
2. Yang kedua anda diminta untuk memberikan pemeringkatan dari setiap dosen yang memiliki karya penelitian dengan kondisi, jika penelitiannya berjumlah lebih dari sama dengan 2, maka diberi status "Baik". Jika penelitiannya berjumlah kurang dari 2, maka diberi status "Cukup"!

BAB 9

TRANSACT SQL

Capaian Pembelajaran:

1. Mampu memahami lebih lanjut tentang perintah SQL transact sql pada DBMS MySQL
2. Mampu mengimplementasikan pembuatan transact SQL yang diperlukan pada DBMS MySQL

9.1 Transact SQL

SQL merupakan bahasa pemrograman non prosedural, yang artinya alur program tidak seperti bahasa pemrograman biasa, melainkan melalui "request" dan "response". *Transact SQL* mengembangkan kemampuan SQL, sehingga *Transact SQL* melengkapi SQL dengan instruksi logic (*procedural logic*). Sehingga hasil proses SQL Server (ResultSet) dapat diolah lebih lanjut menggunakan logic pemrograman procedural, antara lain seperti Fungsi, Prosedur, Loop, Case, If-Then-Else dan lainnya

Transact SQL merupakan operasi sekuensial untuk memanipulasi data dalam database, yang dieksekusi sebagai satu kesatuan perintah tunggal. *Transact SQL* akan berhasil jika setiap operasi kode program sql individu dalam grup juga berhasil. Jika salah satu operasi kode program sql dalam *Transact SQL* gagal, maka seluruh *Transact SQL* akan gagal. Untuk menggunakan feature *Transact SQL*, harus dimulai dengan **START TRANSACTION** di akhir dengan **COMMIT** atau **ROLLBACK**.

MySQL Transaction, jadi jika terjadi error pada sebuah query dalam blok transaksi, semua perubahan yg terjadi di query

sebelumnya akan diabaikan dan query selanjutnya akan diabaikan juga atau istilah nya **ROLLBACK**. Akan tetapi jika semua query dari setiap operasi berhasil, maka diakhiri **COMMIT**. **COMMIT** merupakan sebuah perintah menyimpan perubahan secara fisik ke database, sedangkan **ROLLBACK** merupakan perintah untuk mengabaikan semua perubahan dan **store engine tabel** yang digunakan yaitu **innnoDB**.

Aturan Penulisan transact sql:

```
START TRANSACTION;
    [BLOK QUERY]
COMMIT/ROOLBACK;
```

9.2 Contoh Studi Kasus

Pada kesempatan kali ini akan diberikan contoh penerapan Transact SQL pada kasus saldo customer

1. Struktur Tabel Customer

Fields	Indexes	Foreign Keys	Triggers	Options	Comment	SQL Preview
Name						
id						1
name						
saldo						

Create table customer (id int(11) not null primary key Auto_increment, name varchar(40), saldo int(11));

2. Isian data Customer

Insert into customer

(id,name,saldo)

Values

("","John Doe",100),

("","Will Smith",100);

id	name	saldo
1	John Doe	100
2	Will Smith	100
3	Sania	250
4	Willy	200

Data di atas merupakan data awal saldo customer sebelum dilakukan perubahan dengan menggunakan perintah Update.

3. Operasi perubahan data dengan Update dan dilakukan Rollback

```
mysql> BEGIN;
Query OK, 0 ROWS affected (0.01 sec)

mysql> UPDATE customer SET saldo = saldo - 20 WHERE id = 1;
Query OK, 1 ROW affected (0.00 sec)
ROWS matched: 1 Changed: 1 Warnings: 0

mysql> SELECT * FROM customer;
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
| 1  | John Doe  | 80    |
| 2  | Will Smith | 100   |
+----+-----+-----+
2 ROWS IN SET (0.00 sec)

mysql> ROLLBACK;
Query OK, 0 ROWS affected (0.10 sec)
```

4. Perubahan data tidak tersimpan karena dilakukan Rollback

```
mysql> SELECT * FROM customer;
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |   100 |
|  2 | Will Smith |   100 |
+----+-----+-----+
2 ROWS IN SET (0.00 sec)
```

5. Operasi perubahan data dengan Update dan dilakukan Commit

```
mysql> BEGIN;
Query OK, 0 ROWS affected (0.00 sec)

mysql> UPDATE customer SET saldo = saldo - 30 WHERE id = 1;
Query OK, 1 ROW affected (0.00 sec)
ROWS matched: 1 Changed: 1 Warnings: 0
```

Operasi update saldo customer yang bernama "John Doe"

```
mysql> SELECT * FROM customer;
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |    70 |
|  2 | Will Smith|   100 |
+----+-----+-----+
2 ROWS IN SET (0.00 sec)

mysql> COMMIT;
Query OK, 0 ROWS affected (0.05 sec)

mysql> SELECT * FROM customer;
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |    70 |
|  2 | Will Smith|   100 |
+----+-----+-----+
2 ROWS IN SET (0.00 sec)
```

Operasi update saldo customer yang bernama “John Doe”, yang semula 100 berubah menjadi 70 dan tersimpan di database setelah **COMMIT**

6. Operasi Transact SQL menggunakan store procedure

```
mysql> delimiter //
mysql> create procedure update_saldo_customer
-> (p_id int)
-> begin
-> start transaction;
-> if exists (select * from customer
-> where id=p_id)
-> then update customer set saldo=saldo-10 where id=p_id;
-> commit;
```

```
-> else
-> rollback;
-> end if;
-> end//
Query OK, 0 rows affected (0.06 sec)
```

Operasi memanggil store procedure yang di dalamnya terdapat operasi transact SQL

```
cmd Select C:\Windows\System32\cmd.exe - mysql -urc
3 rows in set (0.00 sec)

mysql> select * from customer;//
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |   100 |
|  2 | Will Smith|   100 |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql> call update_saldo_customer
-> ("1");//
Query OK, 0 rows affected (0.18 sec)

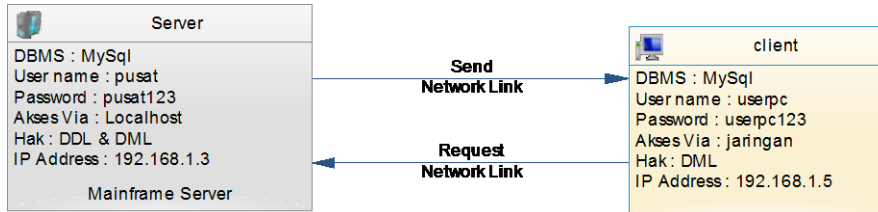
mysql> select * from customer;//
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |    90 |
|  2 | Will Smith|   100 |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql> call update_saldo_customer
-> ("3");//
Query OK, 0 rows affected (0.00 sec)

mysql> select * from customer;//
+----+-----+-----+
| id | name      | saldo |
+----+-----+-----+
|  1 | John Doe  |    90 |
|  2 | Will Smith|   100 |
+----+-----+-----+
```

9.3 Soal Latihan

SKEMA JARINGAN DAN DBMS CONNECTION



Struktur Tabel “Tbl_Koleksi”

	Column Name	Data Type	Allow Nulls
🔔	No_ISBN	varchar(20)	<input type="checkbox"/>
	Judul	varchar(255)	<input checked="" type="checkbox"/>
	Pengarang	varchar(255)	<input checked="" type="checkbox"/>
	Thn_Terbit	int	<input checked="" type="checkbox"/>
	Penerbit	varchar(255)	<input checked="" type="checkbox"/>
	Stok_Awal	int	<input checked="" type="checkbox"/>
	Harga_Jual	float	<input checked="" type="checkbox"/>

Baris Data Tabel “Tbl_Koleksi”

No_ISBN	Judul	Pengarang	Thn_Terbit	Penerbit	Stok_Awal	Harga_Jual
9786027680012	Aplikasi Resto ...	Muhammad Sa...	2010	Maxikom	100	65000.0000
9786027680015	Sistem Informa...	Rahimi Fitri, S.K...	2012	Elex Media	100	45000.0000
9786027680017	Pemrograman ...	Rahimi Fitri, S.K...	2012	Elex Media	100	55000.0000
9786027680020	Sistem Komput...	Indra Selambang	2012	Elex Media	100	65000.0000

Struktur Tabel “Tbl_Log_Transaksi”

Column Name	Data Type	Allow Nulls
Kejadian	Varchar (50) <input checked="" type="checkbox"/>	<input type="checkbox"/>
Waktu	Datetime	<input checked="" type="checkbox"/>

Struktur Tabel “Tbl_Transaksi_Detail”

Column Name	Data Type	Allow Nulls
No_Jual	int	<input checked="" type="checkbox"/>
No_Isbn	varchar(20)	<input checked="" type="checkbox"/>
Jumlah	int	<input checked="" type="checkbox"/>
Harga	float	<input checked="" type="checkbox"/>
Sub_Total	float	<input checked="" type="checkbox"/>

Soal:

1. Atur konfigurasi jaringan sesuai skema yang ada!
2. Buat Database **TB_Gramedia** dengan Tabel-tabel yang diperlukan sesuai dengan struktur tabel yang ada (**Tbl_Koleksi**, **Tbl_Log_Transaksi**, **Tbl_Transaksi_Detail**)
3. Insert data pada **Tbl_Koleksi** sesuai dengan **Baris data tabel Tbl_Koleksi** di atas!
4. Buat Transact SQL di dalam store procedure **insert_transaksi_detail** yang dapat menambah data/insert di **Tbl_Transaksi_Detail**, kemudian melakukan pencarian/select dan update data **stok_awal** di **Tbl_Koleksi**.
5. Buat Trigger **Log_Transaksi** yang berfungsi untuk menambah data/insert di **Tbl_Log_Transaksi** yang dilakukan/dieksekusi setelah insert **Tbl_Transaksi_Detail** (**Call insert_transaksi_detail**)

HASIL YANG DIHARAPKAN!

Call insert_transaksi_detail

```
mysql> call insert_penjualan_detail
-> ("1001","9786027680015","2","45000","90000");//
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> select * from I_penjualan_detail;//
```

No_Jual	No_ISBN	jumlah	harga	sub_total
1001	9786027680012	2	65000	130000
1001	9786027680015	2	45000	90000

2 rows in set (0.00 sec)

```
mysql> select * from t_Buku;
```

```
-> //
```

No_ISBN	Judul	Pengarang	Thn_Terbit	Penerbit
9786027680012	Aplikasi Resto	Muhammad S	2010	Maxikom
9786027680015	Sistem Informasi	Rahimi Fitri	2012	Elex media
9786027680017	Penrograman	Rahimi Fitri	2012	Elex media
9786027680020	Sistem Komputer	Indra Selambang	2012	Elex media

4 rows in set (0.00 sec)

```
mysql> select * from Tbl_Log_Transaksi ;
```

```
-> //
```

kejadian	waktu
Penjualan Buku	2014-06-15 21:22:58

1 row in set (0.00 sec)

Daftar Pustaka

1. Agus Mulyanto, 2009, Teori Client Server, Universitas Komputer Indonesia, Bandung.
2. C. J. Date. 2006. An Introduction to Database Systems 8th. Pearson Education
3. Henry F. Korth, Abraham Silberschatz. 2011. Database system concepts 6th Edition. McGraw-Hill
4. Jeffrey Ullman, Jennifer Widom, and Hector Garcia-Molina. 2013. Database Systems: Pearson New International Edition: The Complete Book.
5. Raghu Ramakrishnan and Johannes Gehrke. 2003. Database Management Systems Third Edition. McGraw-Hill
6. Raharjo Budi, 2015 Belajar Otodidak Teknik Pembuatan dan Pengelolaan Database, Informatika, Bandung
7. Solichin Achmad, 2010, Mysql 5 dari Pemula Hingga Mahir, Universitas Budi Luhur, Jakarta

SISTEM BASIS DATA MYSQL

PADA KONSEP JARINGAN KLIEN SERVER

ABDUL ROZAQ

Kesempurnaan hanyalah Allah SWT yang memilikinya, dan kita hanyalah manusia biasa yang tak luput dari salah dan dosa. Kiranya para pembaca dalam mencermati buku ajar ini bisa memberikan sumbang saran dan kritik yang nantinya bisa digunakan dalam mengoreksi serta mengevaluasi bahan perkuliahan ini. Atas kritik dan saran yang diberikan, diucapkan banyak terima kasih, semoga Allah membalas kebaikan yang pembaca sampaikan dengan berlipat ganda amin.

Akhirnya, semoga apa yang disampaikan dapat bermanfaat bagi penulis khususnya dan semua pihak yang berkepentingan pada umumnya.
Wassalamuallaikum Wr. Wb.



Penerbit Poliban Press
Redaksi :
Politeknik Negeri Banjarmasin, Jl. Brigjen H. Hasan Basry,
Pangeran, Komp. Kampus ULM, Banjarmasin Utara
Telp : (0511)3305052
Email : press@poliban.ac.id

