FIKOM

MODUL PRAKTIKUM

BASIS DATA II

20
24
/
20
25

Tim Penyusun

- Amaliah Faradibah, S.Kom.,M.Kom.,MTA., MCF
- Ir. Dedy Atmajaya, S.Kom.,M.Eng.,MTA.
- Andi Ulfah Tenripada, S.Kom.,M.Kom.,MTA.
- Tim Asisten Laboratorium

Laboratorium Terpadu
Fakultas Ilmu Komputer
Universitas Muslim Indonesia

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Kuasa, yang telah memberikan rahmat-Nya sehingga Modul Praktikum **Basis Data 2** untuk mahasiswa/i Program Studi Teknik Informatika dan Sistem Informasi Fakultas Ilmu Komputer Universitas Muslim Indonesia ini dapat diselesaikan dengan sebaik-baiknya.

Modul praktikum ini dibuat sebagai pedoman dalam melakukan kegiatan praktikum **Basis Data 2** yang merupakan kegiatan penunjang mata kuliah pada Program Studi Teknik Informatika dan Sistem Informasi. Modul praktikum ini diharapkan dapat membantu mahasiswa/i dalam mempersiapkan dan melaksanakan praktikum dengan lebih baik, terarah, dan terencana. Pada setiap topik telah ditetapkan capaian pembelajaran mata kuliah pelaksanaan praktikum dan semua kegiatan yang harus dilakukan oleh mahasiswa/i serta teori singkat untuk memperdalam pemahaman mahasiswa/i mengenai materi yang dibahas.

Penyusun menyakini bahwa dalam pembuatan Modul Praktikum **Basis Data 2** ini masih jauh dari sempurna. Oleh karena itu penyusun mengharapkan kritik dan saran yang membangun guna penyempurnaan modul praktikum ini dimasa yang akan datang.

Akhir kata, penyusun mengucapkan banyak terima kasih kepada semua pihak yang telah membantu baik secara langsung maupun tidak langsung.

Makassar, September 2024

Tim Penyusun

TATA TERTIB PELAKSANAAN PRAKTIKUM

Tata Tertib Pelaksanaan Praktikum pada Laboratorium Terpadu Fakultas Ilmu Komputer UMI adalah sebagai berikut:

1. Seluruh Pengguna laboratorium harus dalam keadaan sehat tidak menunjukkan gejala sakit (batuk, hidung tersumbat, dan suhu badan diatas 37°C).
2. Praktikan hanya diizinkan melaksanakan praktikum apabila :
   a. Pria
      - Berpakaian rapi memakai kemeja putih polos;
      - Menggunakan celana kain berwarna hitam bukan dari bahan jeans/semi jeans;
      - Rambut rapi dan tidak panjang;

   b. Wanita
      - Berpakaian rapi memakai kemeja tunik putih polos (tidak transparan)
      - Memakai Jilbab Segitiga Hitam (bukan pasmina) dan menutupi dada.
      - Menggunakan Rok Panjang berwarna hitam yang tidak terbelah dan tidak span serta bukan dari bahan jeans/semi jeans;
      - Memakai kaos kaki dengan tinggi minimal 10 cm di atas mata kaki;
3. Ketika memasuki dan selama berada dalam ruangan, praktikan diwajibkan :
   - Tenang, tertib, dan sopan;
   - Tidak mengganggu praktikan lain yang sedang melaksanakan praktikum;
   - Tidak diperbolehkan merokok, membawa makanan / minuman senjata tajam dan senjata api ke dalam ruangan praktikum;
   - Tidak diperbolehkan membawa *handphone* ke meja praktikum dan *handphone* dalam mode senyap;
   - Tidak diperbolehkan membawa media penyimpanan eksternal atau *flashdisk* ke meja praktikum tanpa seizin Dosen Pengampu atau Asisten;
4. Dilarang membawa, mengambil, serta memindahkan perangkat yang digunakan pada saat praktikum tanpa instruksi dari Dosen Pengampu atau Asisten.
5. Toleransi keterlambatan praktikan maksimal 5 menit.
6. Praktikan berada diarea laboratorium dengan mengikuti jadwal yang telah ditentukan oleh Kepala Laboratorium.

7. Penggunaan fasilitas Laboratorium menyesuaikan dengan kapasitas ruang Laboratorium.
8. Segala pelanggaran yang dilakukan oleh praktikan akan berakibat pada penutupan dan penghentian penggunaan seluruh fasilitas laboratorium dan ditindak sesuai dengan aturan yang berlaku.

SANKSI-SANKSI

Sanksi terhadap pelanggaran **TATA TERTIB**:

Dosen Pengampu dan Asisten laboratorium berhak menjatuhkan sanksi, sesuai dengan aturan yang berlaku di Laboratorium Terpadu Fakultas Ilmu Komputer UMI apabila :

1. Praktikan merusak peralatan praktikum (*Personal Computer*) secara sengaja, maka praktikan bertanggung jawab untuk mengganti kerusakan tersebut.
2. Praktikan tidak mematuhi dan mentaati aturan praktikum maka tidak diperkenankan mengikuti praktikum.

Pelanggaran point lainnya dikenakan sanksi teguran, dikeluarkan/dicoret namanya dalam kegiatan praktikum (mengulang mata kuliah sesuai dengan semester berjalan) sampai sanksi akademik.

Kepala Laboratorium Terpadu,

Ir. Abdul Rachman Manga', S.Kom., M.T., MTA., MCF

DAFTAR ISI

MODUL 1 – Relationship (ERD), DDL

A. Capaian Pembelajaran Mata Kuliah (CPMK)

1. Mampu menjelaskan konsep basis data, data dan informasi, kegunaan basis data, dan perkembangan basis data, Jenis Bahasa Query Formal.
2. Mampu mengimplementasikan rancangan Database, mengelola Database, Menggabungkan Beberapa Tabel, Mengelola View, Mengelola Trigger, Mengelola Procedure, Menggunakan Fasilitas Backup dan Sekuritas

B. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

1. Mahasiswa dapat merancang basis data dan mentranformasikan rancangan logical database ke phisical database.
2. Mahasiswa dapat menggunakan SQL dasar untuk DDL (Data Definition Language) dan DML (Data Manipulation Language)

C. Teori Dasar

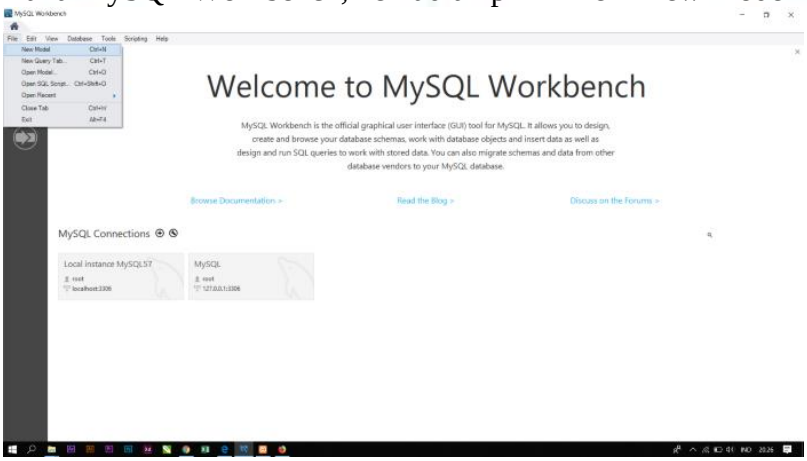
1. ERD (Entity Relation Database)

Model E-R adalah suatu model yang digunakan untuk menggambarkan data dalam bentuk entitas, atribut dan hubungan antaraentitas. Entitas adalah sesuatu dalam dunia nyata yang keberadaannya tidak bergantung pada yang lain (Elmasri dan Navathe, 1994). Sebagai contoh, setiap pegawai dalam sebuah organisasi adalah sebuah entitas. Entitas dapat berupa sesuatu yang nyata ataupun abstrak (berupa suatu konsep). Secara lebih rinci, hoffer, dkk, (2005) menjelaskan bahwa entitas dapat berupa seseorang, sebuah tempat, sebuah objek, sebuah kejadian, atau suatu konsep. Setiap entitas dinyatakan oleh sejumlah atribut. Atribut adalah property atau karakteristik yang terdapat pada setiap entitas. Setiap atribut dinyatakan dengan kata benda. Supaya konsisten, (Hoffer, dkk, 2005) menggunakan huruf kapital untuk setiap awal kata dan huruf kecil untuk yang lain.

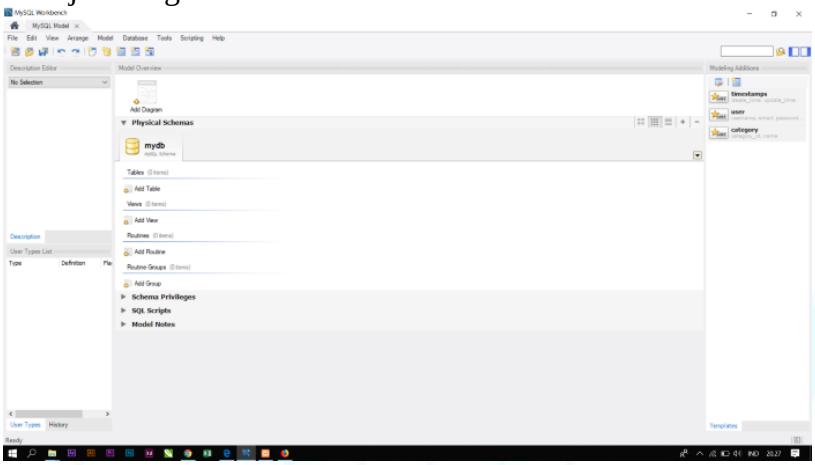
2. MySQL Workbench

Berikut langkah-langkah membuat ERD di MySQL Workbench:

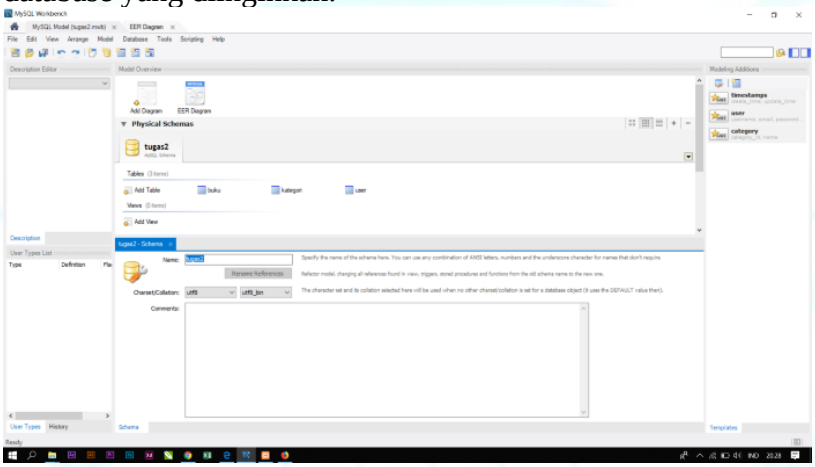
1. Buka MySQL Workbench, kemudian pilih File > New Model



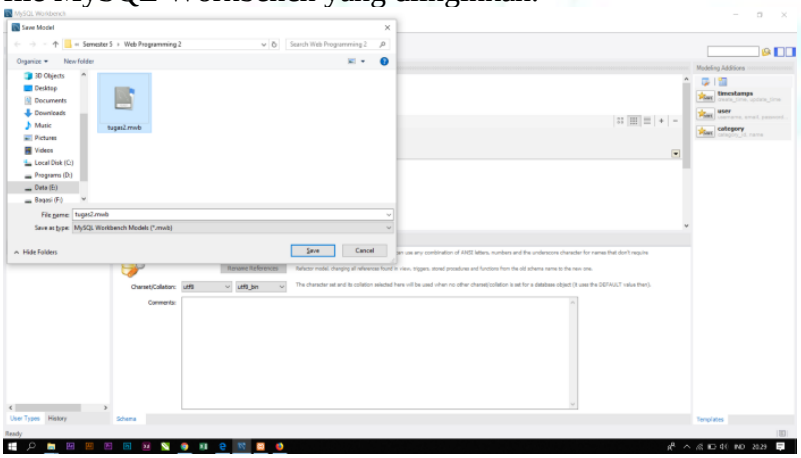
2. Kemudian akan tampil lembar kerja dengan schema otomatis bernama mydb. Schema dalam MySQL sama artinya dengan database. Jadi, ketika user meng-klik new model, akan muncul database baru dengan nama mydb seoerti ditunjukkan gambar berikut.



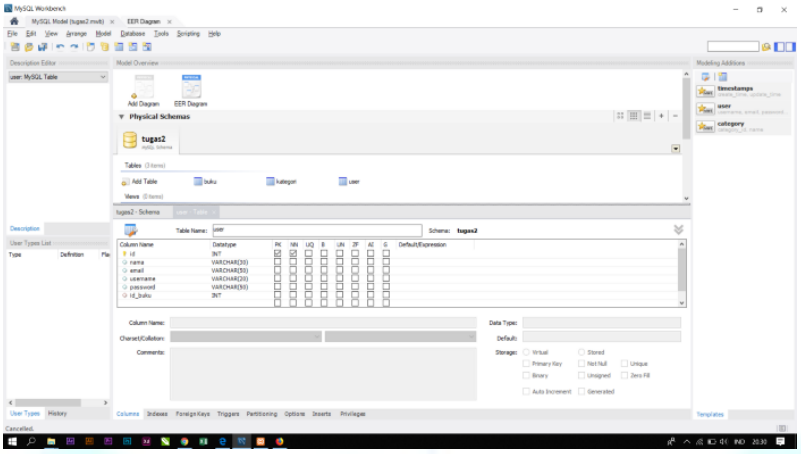
3. Kemudian klik 2x pada nama mydb, kemudian rename schema menjadi nama database yang diinginkan.



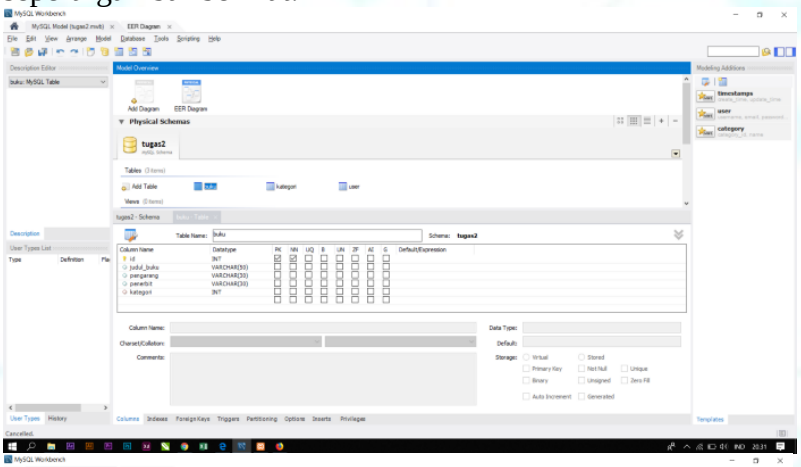
4. Klik tombol save di toolbar, kemudian simpan di destinasi folder dengan nama file MySQL Workbench yang diinginkan.

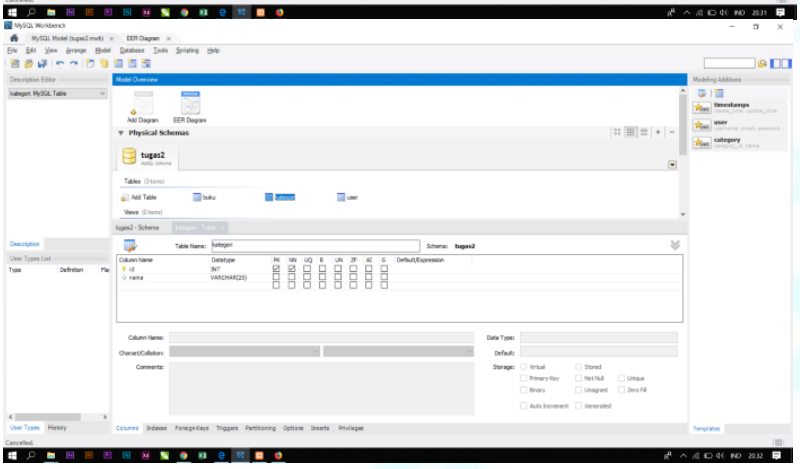


5. Kemudian klik 2x tombol pada tombol Add Table di bawah nama schema pada lembar kerja. Kemudian beri nama tabel dan isi atribut yang diperlukan untuk tabel tersebut.

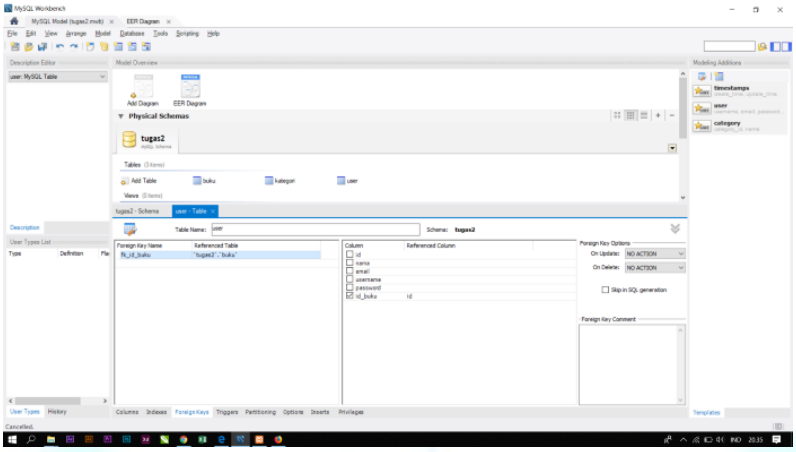


6. Lakukan langkah nomor 5 untuk membuat table buku dan kategori, hasilnya seperti gambar berikut.

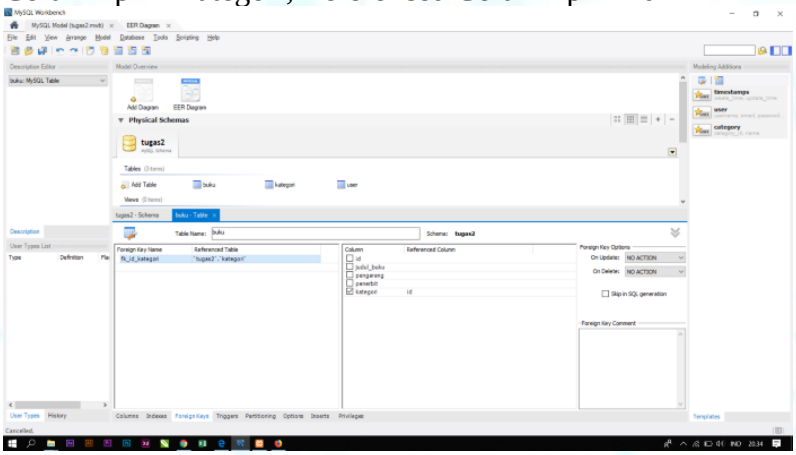




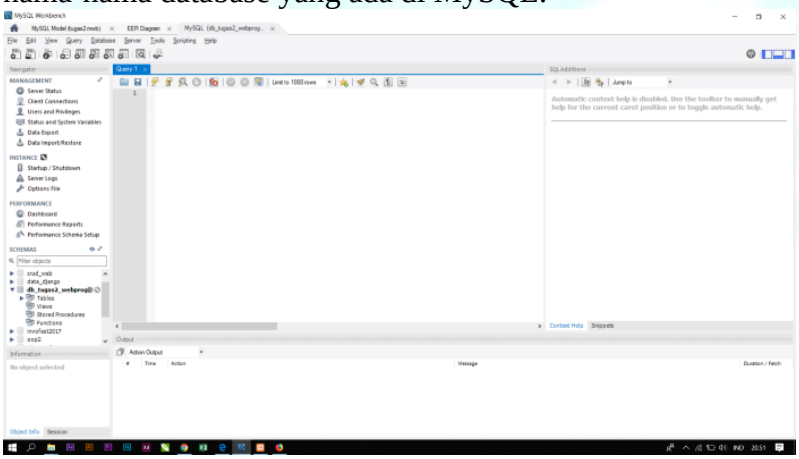
7. Pada table user, klik tab foreign key untuk membuat sebuah relasi databasenya. Foreign Key Name diisi fk_id_buku, Referenced Table ‘tugas2’.’buku’, Column pilih id_buku, Referenced Column pilih id



8. Pada table buku, klik tab foreign key untuk membuat sebuah relasi databasenya. Foreign Key Name diisi fk_id_kategori, Referenced Table ‘tugas2’.’kategori’, Column pilih kategori, Referenced Column pilih id



9. Buka halaman “Home”, kemudian pilih “local instance mysql”. Kemudian akan tampil halaman sebagai berikut. Perhatikan pada panel navigator, akan muncul nama-nama database yang ada di MySQL.



3. Tipe Data MySQL

Beberapa tipe data yang disediakan MySQL antara lain :

Tipe Data	Keterangan	Range	Format
Int	Angka	-2147483648 – 2147483648	
Float	Angka Desimal		
Date	Tanggal		YYYY-MM-DD
DateTime	Tanggal & Waktu		YYYY-MM-DD HH:MM:SS
Char	String	1 – 255 Char	
VarChar	String	1 – 255 Char	
Blob	String	<= 65535 Char	
LongBlob	String	<= 4294967295 Char	

4. Database Relational

Database Relational atau kita sering kita sebut database, merupakan kumpulan dari tabel-tabel. Sedangkan tabel merupakan kumpulan dari beberapa Field/ baris atau column. Untuk membuat suatu tabel maka seorang user harus membuat database terlebih dahulu. Kemudian mengaktifkan database yang dibuat tersebut.

5. Data Definition Language (DDL)

Berikut ini adalah beberapa perintah DDL:

Perintah	Bentuk Umum	Implementasi
Database		
Create	create database **nama_database;**	create database praktrikum;
Show	show databases;	show databases;
Use	use **nama_database;**	use praktikum;
Drop	drop **nama_database;**	drop praktikum;
Table		
Create	CREATE TABLE **[nama tabel] ([definisi kolom]) [parameter tabel];**	create table mahasiswa (stambuk varchar(12), nama_lengkap varchar(50));
Alter	ALTER TABLE **table_name** ADD **column_name datatype;** ALTER TABLE **table_name** DROP **column_name datatype;** RENAME TABLE **table_name** TO **new_table name**	ALTER TABLE Jurnal ADD Tanggal_lahir Date;
Drop	DROP TABLE **nama_tabel;**	Drop table mahasiswa;
Create dengan Key	CREATE TABLE **namatabel (Field1 TipeData1, Field2 TipeData2,** FOREIGN	

	KEY (Field2) REFERENCES namatabelinduk (namakolominduk)	
Constraint	ALTER TABLE namatabel ADD CONSTRAINT namaconstraint FOREIGN KEY (namakolom) REFERENCES namatabelinduk (namakolominduk)	

D. Kegiatan Praktikum

1. Instrument

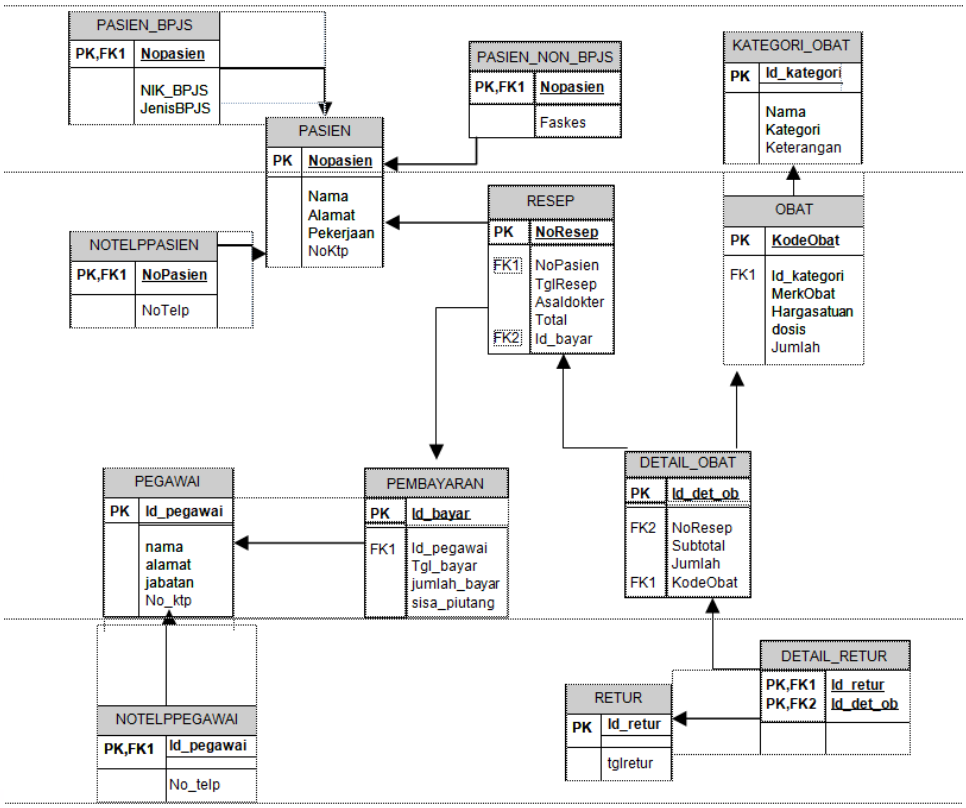
a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MySQL versi 5.0 atau di atasnya (sebagai engine basis data)

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

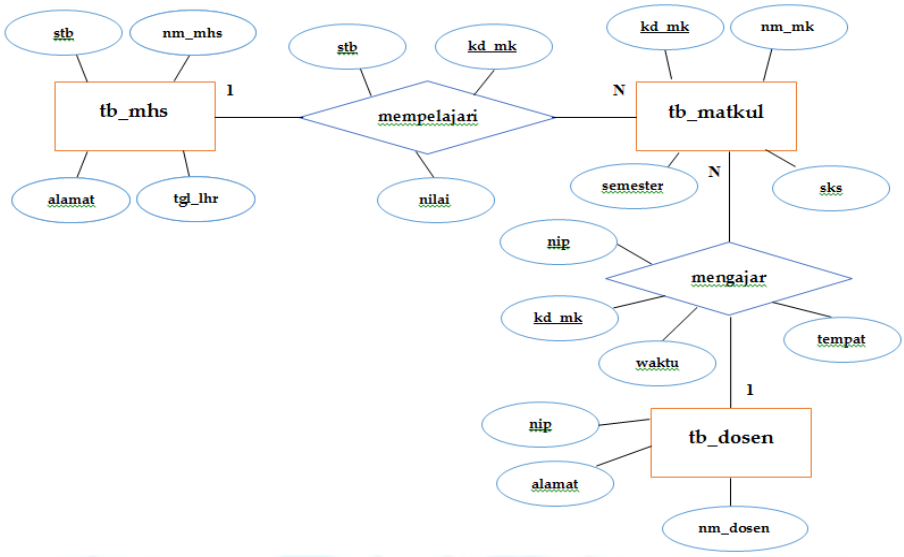
3. Studi Kasus

a) Buatlah Database dengan nama **apotek**
b) Buatlah tabel menggunakan skema database **apotek** pada gambar berikut.
Perhatikan field yang memiliki tanda PK (Primary Key).
c) Tambahkan reference tabel / Foreign Key
Perhatikan penulisan tanda panah pada garis menunjukan dari mana asal dari (reference) atribut key berasal. Misal Id_pegawai pada tabel pembayaran merupakan FK (Foreign key) berasal dari tabel pegawai. Pada tabel pegawai kolom id_pegawai adalah primary key nya. Perhatikan bentuk umum dari penggunaan contrains reference.
d) Masukkan data pada **tabel pasien** dan **tabel pegawai** minimal 5 record
Sintaks/Bentuk Umum:
INSERT INTO nama_tabel VALUES ('datafield1', ' datafield2', datafieldn');



LEMBAR EVALUASI PRAKTIKUM

1. Tulis semua perintah SQL percobaan di atas dan Screan
2. Jelaskan cara menambahkan field Alamat sebagai field awal pada tabel karyawan
3. Jelaskan cara menginput field total_transaksi tampa pengimputan langsung isi fieldnya.
4. Buat sebuah database dengan nama db_stambukAnda
5. Buat tabel sesuai dengan Entity Relationship Diagram (ERD) berikut ini:
6. Isikan data pada setiap tabel minimal 5-10 record.
7. Tambahkan field Inisial char(10) UNIQUE KEY pada tabel dosen setelah field primary key



Evaluasi Praktikum 1:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Mentransformasikan Skema ke Table Mysql					
2.	Pemahaman Perintah DDL					
3.	Pemahaman Perintah DDL (Primary dan Foreign Key)					
4.	Pemahaman Perintah DML (Insert dan Select)					

Catatan Asisten:

Dosen : _____

Asisten 1 : _____

Asisten 2 : _____

TINGKAT KEBERHASILAN

MODUL 2 – DML, SQL JOIN, VIEW

A. Capaian Pembelajaran Mata Kuliah (CPMK)

Mampu mengimplementasikan rancangan Database, mengelola Database, Menggabungkan Beberapa Tabel, Mengelola View, Mengelola Trigger, Mengelola Procedure, Menggunakan Fasilitas Backup dan Sekuritas

B. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

1. Mahasiswa dapat menggunakan SQL dasar untuk DDL (Data Definition Language) dan DML (Data Manipulation Language)
2. Mahasiswa dapat menggunakan SQL JOIN untuk merelasikan beberapa tabel

C. Teori Dasar

DML (Data Manipulation Language)

Merupakan perintah SQL yang berkaitan dengan manipulasi atau pengolahan data atau record dalam table. Perintah DML antara lain: SELECT, INSERT, UPDATE, DELETE.

1. Insert Tabel

Insert merupakan perintah yang dapat digunakan untuk melakukan input data ke dalam tabel yang sudah ada.

Perintahnya :

```
Insert Into Nama_Table Values (
            Isi_Field_1, Isi_Field_2, ⋯ , Isi_Field_N) ;

Insert Into Nama_Table (
            Nama_Field_1, Nama_Field_2, ⋯ , Nama_Field_N)
       Values
            (Isi_Field_1, Isi_Field_2, ⋯ , Isi_Field_N) ;
```

2. Query Sederhana

Select merupakan perintah yang dpat digunakan untuk :

- Menampilkan data secara kesuluruhan yang terdapat di dalam tabel
- Menampilkan data tertentu yang terdapat di dalam tabel
- Menampilkan dan mengurutan data secara *ascending* dan *descending*

Menampilkan Data Secara Keseluruhan

Perintahnya : `Select * From Nama_Table;`

Menampilkan Kolom Data Tertentu

Jika hanya ingin menampilkan beberapa field tertentu dalam suatu table. Misalkan dari data yang terdapat pada tabel Mahasiswa yang mempunyai Field (NIM, Nama_Mhs, Alamat) dan hanya akan menapilkan NIM dan Nama_Mhs.

Perintahnya :

```
Select Nama_Field_1, … , Nama_Field_N From Nama_Table;
```

Menampilkan Baris Data Tertentu

Jika hanya ingin menampilkan beberapa baris tertentu dalam suatu table. Misalkan dari data yang terdapat pada tabel Mahasiswa ingin menampilkan baris tertentu maka akan ditambahkan kondisi pada clause setelah where.

Perintahnya :

```
Select * from Nama_Table where Kondisi;
```

3. **Update Tabel**

Update merupakan perintah yang dapat digunakan untuk melakukan perubahan terhadap datayang sudah ada/dibuat. Latihan setelah sub bab ini kita gunakan **skema order entry.**

Perintahnya :

```
Update Nama_Table Set Nama_Field = 'Data_Baru'
                  Where Nama_Field_Key = 'Data_Key';
```

4. **Delete Data**

Delete merupakan perintah yang dapat digunakan untuk menghapus data yang terdapat di dalam tabel.

Perintahnya :

```
Delete From Nama_Table Where Nama_Field_Key;
```

5. **Query dengan Kondisi**

Query dengan perbandingan kondisi bentuk umumnya adalah sbb :

```
Select * from Nama_Table where Kondisi;
```

Pada bagian kondisi bisa diberikan berbagai value misalnya salah satunya dengan beberapa operator relasional.

Operator Relasional

Operator relasional merupakan operator yang digunakan untuk membandingkan antara dua buah nilai dalam suatu table.

Operator	Keterangan
=	Sama dengan
>	Lebih besar dari
<	Lebih kecil dari
>=	Lebih besar dari sama dengan
<=	Lebih kecil dari sama dengan
<>	Lebih kurang

Perintahnya :

```
Select * From Nama_Table
        Where Nama_Field[Operator Relasional] Ketentuan;
```

6. Urutan Data (ACS, DESC, Order By)

Untuk mengurutkan tampilan data dari suatu table, digunakan klausa Order By.Klausa Order By, dapat digunakan untuk mengurutkan data :

- **Asc (*Ascending*)** : Untuk mengurutkan data dari kecil ke besar
- **Desc (*Descending*)** : Untuk mengurutkan data dari besar ke kecil

Perintahnya :

```
Select * From Nama_Table Order By Nama_Field_Key Asc/Desc;
```

b) Agregrate Function

Fungsi agregar dapat digunakan untuk mencari jumlah, rata-rata, nilai maksimal dan nilai minimal dalam field yang terdapat pada table.

Beberapa fungsi agregat :

Agregat	Keterangan
Count	Menghitung cacah data
Sum	Penjumlahan data
Avg	Mencari Rata-rata data
Max	Mencari nilai maksimal
Min	Mencari nilai minimal

c) Operator Between, In, Like

Operator Beetween

Operator Between merupakan operator yang digunakan untuk menangani operasi jangkauan.

Perintahnya :

```
Select * From Nama_Table Where Nama_Field_ketentuan Between 'Ketentuan_1' And
'Ketentuan_2';
```

Operator In

Operator In merupakan operator yang digunakan untuk mencocokan suatu nilai.

Perintahnya :

```
Select Nama_Field From Nama_Table
Where Nama_Field_Pencocok In   ('Isi_Field_1','Isi_Field_2');
```

Operator Like

Operator Like merupakan operator yang digunakan untuk mencari suatu data (search).

Perintahnya :

```
Select * From Nama_Table Where Nama_Field_Dicari Like '%Key';
```

d) Ekspresi Query

Ekspresi Query dapat digunakan untuk melakukan perubahan terhadap field kolom keluaran, menambah baris teks field keluaran.

Mengganti Nama Field Keluaran

Perintahnya

```
Select Nama_Field_Asal As 'Nama_Field_Pengganti' From Nama_Table
```

Menambahkan baris teks field keluaran

Perintahnya

```
Select 'Nama Field Tambahan', Nama_Field_Asal From Nama_Table;
```

Ekspresi kondisi

Perintahnya:

```
Select Nama_Field_1 Case Nama_Field_2 When 'Nilai_field_2'
        Then 'Keterangan_1' Else 'Keterangan_2'
        End As Nilai_field_2 From Nama_Table;
```

7. Selection Data Dengan Join Table

Join digunakan untuk menampilkan data dari gabungan dua tabel atau lebih. Ada dua jenis join inner join dan outer join. Inner join dibahasi dibab ini, outer join. Pada INNER JOIN atau CROSS JOIN output/hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat dimana baris yang tampil hanya yang memiliki kondisi kesamaan data. Kesamaan data berdasarkan relasinya (kesamaan data foreign key dengan primary key tabel yang diacu). Berikut adalah bentuk umum INNER JOIN yang umumnya hanya disebut sebagai JOIN:

```
SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2,
        nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2
FROM    tabel1, tabel2
WHERE   tabel1.nama_kolom1 (primary key)=tabel2.nama_kolom1(foreign key yg
mengacu ke tabel1)
```

Clausa Join On Alias

```
SELECT  a.nm_kolom1, b.nm_kolom2, a.nm_kolom3
FROM    tabel1 a
JOIN    tabel2 b
ON      a.nama_kolom1(primary key)=b.nama_kolom1(foreign key yg mengacu ke
tabel1)
WHERE   kondisi;
```

Join 3 Table atau Lebih

Pada prinsipnya sama, hanya jumlah tabel ditambah dan sintaks disesuaikan.

Contoh: penerapan join dua tabel atau lebih untuk menampilkan nama customer, tgl order dan total jumlah order.

```
select a.cust_name,b.order_date,c.quantity
from customers a join orders b on a.cust_id=b.cust_id
join orderitems c on b.order_num=c.order_num;
```

D. Kegiatan Praktikum

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MySQL versi 5.0 atau di atasnya (sebagai engine basis data)

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

3. Studi Kasus

a. Buat sebuah database dengan nama db_StambukAnda
b. Buatlah sebuah table ‘tb_mahasiswa’ seperti deskripsi tabel dibawah ini:

Nama Field	Type	Panjang	Keterangan
Stb	VARCHAR	15	Primary Key
Nama	VARCHAR	30	
JK	VARCHAR	ENUM(‘Pria’,’Wanita)	
Alamat	VARCHAR	50	

c. Buatlah sebuah table ‘tb_matkul’ seperti deskripsi tabel dibawah ini:

Nama Field	Type	Panjang	Keterangan
Kode_mk	VARCHAR	20	Primary Key
Nama_mk	VARCHAR	50	
SKS	INT	1	
Semester	VARCHAR	ENUM(‘Ganjil’,’Genap’)	

d. Buatlah sebuah table ‘tb_nilai’ seperti deskripsi tabel dibawah ini:

Nama Field	Type	Panjang	Keterangan
Stb	VARCHAR	15	Foreign Key
Kode_mk	VARCHAR	20	Foreign Key
Nilai	INT	3	

e. Tambahkan 1 field baru pada tabel tb_nilai yaitu **id_nilai int otomatis bertambah 1 setiap ada data baru dan merupakan primary key**
f. Isikan data pada masing-masing tabel minimal 5 record pada tabel tb_mahasiswa dan tb_matkul
g. Untuk tb_nilai diisi dengan 5 record kode_mk dari tabel tb_matkul dengan stambuk pertama dari tb_mahasiswa, dan 5 record kode_mk dari tabel tb_matkul dengan stambuk kedua dari tb_mahasiswa.

h. Pada Tabel tb_nilai lakukan perintah select berikut:
   - Select + where + operator (= => <= < > ==)
   - Select + where + operator Logika (AND, OR)
   - Select + where + LIKE (a_, %a, %a%)
   - Select + where + IN
   - Select + where + BETWEEN
   - Select + ORDER BY (ASC/ DESC)
   - Select + GROUP BY

i. Gunakan Perintah Join untuk menampilkan semua data seperti tabel berikut ini:

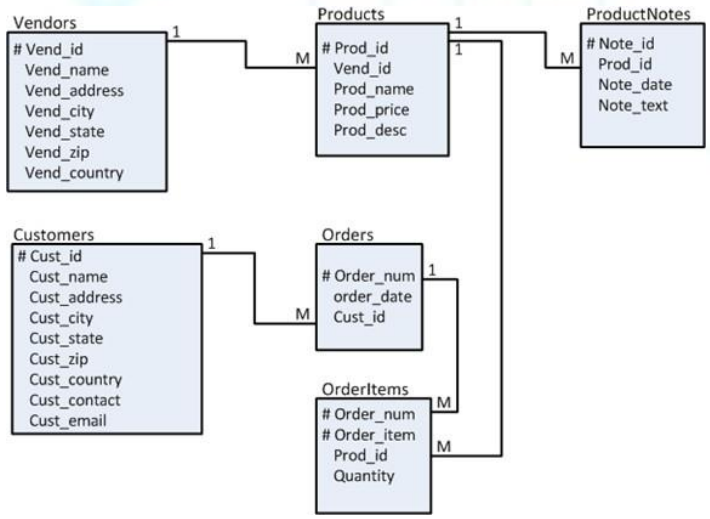
Stb	Nama	Kode_mk	Nama_mk	SKS	Nilai

j. Buat sebuah View dengan nama vw_nilai dengan menggabungkan field-field dari beberapa tabel yang telah Anda buat. Tampilannya sebagai berikut:
   Sintax Umum View:
   CREATE VIEW nama_view AS
   SELECT kolom_1, kolom_2, kolom_n
   FROM nama_table
   WHERE kondisi;

Stb	Nama	Kode_mk	Nama_mk	SKS	Nilai

k. Update salah satu record pada tabel tb_mahasiswa dan tb_nilai, setelah itu tampilkan isi tabel view vw_nilai. Amati perubahan yang terjadi pada tabel view.

LEMBAR EVALUASI PRAKTIKUM



1. Buatlah Tabel sesuai Skema diatas
   a. Masukkan data minimal 10 record memiliki hubungan dengan tabel yang berlasi.

   b. Menampilkan prod_name, vend_name dari tabel vendors dan product
   c. Buat query join 3 table menggunakan alias
2. Tuliskan perintah sql untuk mengupdate record di tabel View?
3. Buatlah database dengan 3 table yang saling berhubungan, kemudian buatlah View dengan 3 tabel tersebut!
4. Buat sebuah database terdiri dari 2 tabel yang berelasi.

Evaluasi Praktikum 2:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat Menginput Data pada tabel					
2.	Dapat Menggunakan Printah DML Select					
3.	Dapat Menggunakan Perintah JOIN					
4.	Dapat Membuat Tabel View					

Catatan Asisten:

Dosen : _____

Asisten 1 : _____

Asisten 2 : _____

TINGKAT KEBERHASILAN

MODUL 3 – Stored Procedure, Function

A. Capaian Pembelajaran Mata Kuliah (CPMK)

Mampu mengimplementasikan rancangan Database, mengelola Database, Menggabungkan Beberapa Tabel, Mengelola View, Mengelola Trigger, Mengelola Procedure, Menggunakan Fasilitas Backup dan Sekuritas

B. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat membuat Stored Procedure dan Stored Function

C. Teori Dasar

Stored Procedure

Stored Procedure adalah sebuah prosedur layaknya subprogram (subrutin) di dalam bahasa pemrograman reguler yang tersimpan di dalam katalog basis data. Beberapa kelebihan yang ditawarkan stored procedure antara lain: mengingkatakan performa, mereduksi trafik jaringan, reusable, dan meningkatkan kontrol sekuriti.

Di balik kelebihan tersebut, stored procedure juga memiliki kekurangan. Di antaranya adalah berpotensi meningkatkan beban server dan penulisnaya tidak mudah (memerlukan pengetahuan yang spesifik).

Contoh sintaks stored procedure:
Untuk memanggil sotred preocedure, digunakan perintah CALL (beberapa DBMS ada yang menggunakan EXECUTE). Dalam Implementasinya, penggunaan stored procedure sering melibatkan parameter. Di MySQL, parameter stored procedure dibedakan menjadi tiga mode: IN, OUT, dan INOUT.

IN

Parameter yang merupakan mode default ini mengindikasikan bahwa sebuah parameter dapat di-pass ke dalam stored procedure tetapi nilainya tidak dapat diubah (dari dalam stored procedure

OUT

Mode ini mengindikasikan bahwa stored procedure dapat mengubah parameter dan mengirimkan kembali ke program pemanggil

INOUT

Mode ini pada dasarnya merupakan kombinasi dari mode IN dan OUT.

Sintaks pendefinisan parameter diperlihatkan sebagai berikut :

```
MODE param_name param_type(param_size)
```

Stored procedure dapat mencerminkan beragam operasi data, misalnya seleksi, penambahan, pengubahan, penghapusan, dan juga operasi – oprasi DDL.
Seperti halnya procedure di dalam bahasa pemrograman, stored procedure juga dapat melibatkan variabel, pernyataan kondisional, dan pengulangan.

D. Kegiatan Praktikum

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MySQL versi 5.0 atau di atasnya (sebagai engine basis data)

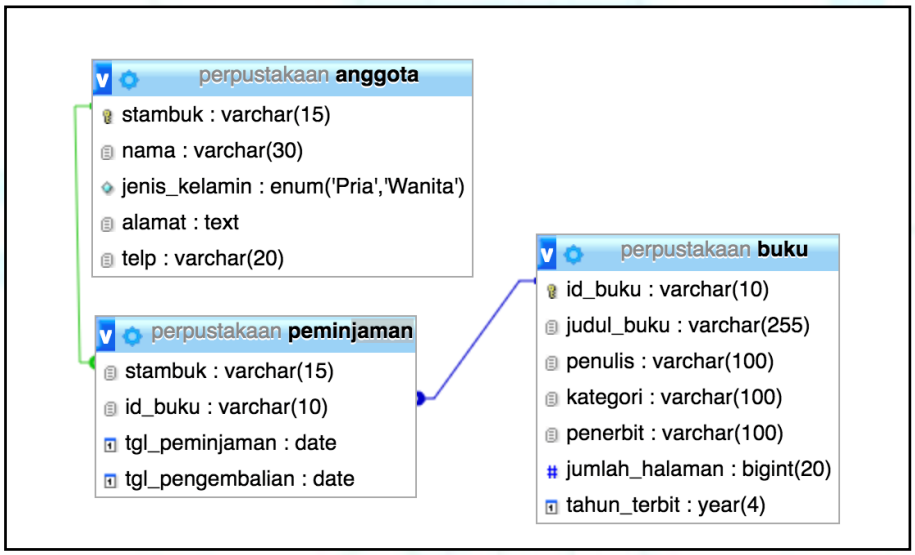
2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

3. Studi Kasus

Store Procedure

a) Buat sebuah database dengan nama ***perpustakaan***
b) Buatlah tabel sesuai gambar relasi tabel dibawah ini:



c) Isikan data pada masing-masing tabel minimal 5-10 record

d) Lakukan Percobaan berikut :

a. Membuat procedure tanpa parameter

```
create procedure tampil_anggota()
select * from anggota;
```

```
call tampil_anggota();
```

b. Membuat procedure menggunakan parameter 'IN'

```
create procedure isi_anggota (in stambuk varchar(15),in nama varchar(30),
in jenis_kelamin enum('Pria','Wanita'), in alamat varchar(50),
in telp varchar(20))
insert into anggota values (stambuk,nama,jenis kelamin,alamat,telp);
```

```
call isi_anggota('13020170156','Rhia','Wanita','Sudiang','082333456789');
```

c. Membuat procedure menggunakan parameter 'OUT'

```
create procedure cek_buku (out x varchar(255), out y varchar(10),
in z varchar(30))
select judul_buku,jumlah_halaman into x,y from buku where id_buku=z;
```

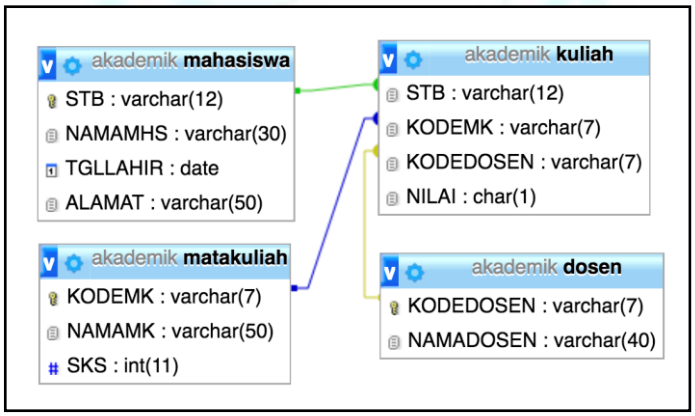
```
MariaDB [perpustakaan]> call cek_buku (@judul_buku,@jumlah_halaman,'B001');
Query OK, 1 row affected (0.05 sec)

MariaDB [perpustakaan]> select @judul_buku,@jumlah_halaman;
+----------------------+-----------------+
| @judul_buku          | @jumlah_halaman |
+----------------------+-----------------+
| Pengantar Pemrograman | 200             |
+----------------------+-----------------+
1 row in set (0.00 sec)
```

Dari contoh yang ketiga ini terlihat bahwa parameter "z" (sebagai IN) digunakan sebagai jalur untuk masukan *routine* dan parameter "x" dan "y" digunakan untuk menampung hasil dari perintah *routine_body*. Pernyataan "into x, y", inilah yang mengakibatkan "x" dan "y" menyimpan informasi judul buku dan jumlah halaman (sebagai kolom yang ter-select).
Pernyataan "call pBuku(@judul_buku, @jumlah_halaman)" menghasilkan informasi yang kemudian disimpan pada parameter @judul_buku dan @jumlah_halaman, sedangkan parameter "z" digunakan untuk menampung string 'B001' yang kemudian digunakan untuk memproses *routine_body*. Kemudian untuk menampilkan informasi ke layar digunakan pernyataan "Select @judul_buku, @jumlah_halaman".

Function

a) Buat sebuah database dengan nama ***akademik***
b) Buatlah tabel sesuai gambar relasi tabel dibawah ini:



c) Isikan data pada masing-masing tabel sesuai gambar dibawah ini:

✓ Tabel Mahasiswa

STB	NAMAMHS	TGLLAHIR	ALAMAT
13020170001	ABI	1998-07-23	PAMPANG
13020170080	RAHMAN	1997-08-08	BTP
13020170071	DITA	1999-01-29	SUDIANG
13020170156	NISA	1998-05-25	SUDIANG
13020170063	AYU	1999-06-29	ANTANG
13020170004	EKO PRANAJAYA	1997-06-18	BTP

✓ Tabel Dosen

KODEDOSEN	NAMADOSEN
D001	ISKANDAR
D002	MARIATI
D003	ZUKIFLI
D004	ABDURAUF

✓ Tabel Matakuliah

KODEMK	NAMAMK	SKS
MK01	KECERDASAN BUATAN	3
MK02	SISTEM DIGITAL	2
MK03	ILMU DAKWAH	2
MK04	PRAKTIKUM BASIS DATA 2	1
MK05	SISTEM TERDISTRIBUSI	2
MK06	JAVA FUNDAMENTAL	2

✓ Tabel Kuliah

STB	KODEMK	KODEDOSEN	NILAI
13020170001	MK01	D002	A
13020170080	MK02	D003	C
13020170071	MK03	D001	B
13020170156	MK04	D002	A
13020170063	MK05	D002	C
13020170004	MK06	D003	E
13020170071	MK04	D004	D
13020170156	MK05	D001	A
13020170063	MK06	D002	B
13020170004	MK01	D003	C
13020170001	MK02	D004	D

d) Lakukan Percobaan Berikut :

1. Membuat fungsi untuk menampilkan nama mahasiswa dengan parameter inputan nim

✓ Pembuatan *function*

```
delimiter $$
CREATE FUNCTION TAMPILNAMA (stambuk varchar(12))
RETURNS VARCHAR(30)
BEGIN
DECLARE NAMA VARCHAR(30);
SELECT NAMAMHS FROM mahasiswa WHERE STB=stambuk INTO NAMA;
RETURN NAMA;
end $$
```

✓ Memanggil *function*

```
MariaDB [akadamik]> SELECT TAMPILNAMA('13020170156');
+---------------------------+
| TAMPILNAMA('13020170156') |
+---------------------------+
| NISA                      |
+---------------------------+
MariaDB [akadamik]> SELECT STB,TAMPILNAMA(STB) FROM mahasiswa;
+-------------+-----------------+
| STB         | TAMPILNAMA(STB) |
+-------------+-----------------+
| 13020170001 | ABI             |
| 13020170004 | EKO PRANAJAYA   |
| 13020170063 | AYU             |
| 13020170071 | DITA            |
| 13020170080 | RAHMAN          |
| 13020170156 | NISA            |
+-------------+-----------------+
```

2. Membuat fungsi untuk menghitung bobot nilai

✓ Pembuatan *function*

```
DELIMITER $$
CREATE FUNCTION HITUNGBOBOT (NIL CHAR(1),JSKS INT)
RETURNS INT(11)
BEGIN
    IF NIL='A' THEN
        RETURN 4*JSKS;
    END IF;
    IF NIL='B' THEN
        RETURN 3*JSKS;
    END IF;
    IF NIL='C' THEN
        RETURN 2*JSKS;
    END IF;
    IF NIL='D' THEN
        RETURN 1*JSKS;
    END IF;
    IF NIL='E' THEN
        RETURN 0*JSKS;
    END IF;
END$$
```

✓ Memanggil *function*

```
MariaDB [akadamik]> SELECT HITUNGBOBOT('A','3');
+-----------------------+
| HITUNGBOBOT('A','3') |
+-----------------------+
|                    12 |
+-----------------------+
```

```
MariaDB [akadamik]> SELECT mahasiswa.STB, kuliah.NILAI, MATAKULIAH.SKS,
    -> HITUNGBOBOT(kuliah.NILAI, matakuliah.SKS) as 'Total Nilai'
    -> FROM kuliah, mahasiswa, matakuliah WHERE
    -> mahasiswa.STB=kuliah.STB and kuliah.KODEMK=matakuliah.KODEMK;
+-------------+-------+-----+-------------+
| STB         | NILAI | SKS | Total Nilai |
+-------------+-------+-----+-------------+
| 13020170001 | A     |   3 |          12 |
| 13020170004 | C     |   3 |           6 |
| 13020170080 | C     |   2 |           4 |
| 13020170001 | D     |   2 |           2 |
| 13020170001 | D     |   2 |           2 |
| 13020170071 | B     |   2 |           6 |
| 13020170156 | A     |   1 |           4 |
| 13020170071 | D     |   1 |           1 |
| 13020170063 | C     |   2 |           4 |
| 13020170156 | A     |   2 |           8 |
| 13020170004 | E     |   2 |           0 |
| 13020170063 | B     |   2 |           6 |
+-------------+-------+-----+-------------+
```

LEMBAR EVALUASI PRAKTIKUM

1. Buatlah sebuah procedure dengan ketentuan sebagai berikut:
   a. Menggunakan parameter IN
   b. Menggunakan parameter OUT
   c. Menggunakan parameter INOUT
2. Jelaskan proses procedure yang terjadi pada setiap parameter
3. Menurut anda untuk apa tujuan penggunaan tanda '@' pada variable
4. Buatlah sebuah *function* menghitung IPK mahasiswa
5. Jelaskan proses perhitungan ipk dengan penggunaan function\Apa perbedaan return dan returns

Evaluasi Praktikum 3:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat membuat Stored Procedure Sederhana tanpa Parameter					
2.	Dapat membuat Stored Function Sederhana tanpa Parameter					

Catatan Asisten:

Dosen : _____

Asisten 1 : _____

Asisten 2 : _____

TINGKAT KEBERHASILAN

MODUL 4 – TRIGGER, REPLIKASI NATIF

A. Capaian Pembelajaran Mata Kuliah (CPMK)

Mampu mengimplementasikan rancangan Database, mengelola Database, Menggabungkan Beberapa Tabel, Mengelola View, Mengelola Trigger, Mengelola Procedure, Menggunakan Fasilitas Backup dan Sekuritas

B. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

1. Mahasiswa dapat membuat Trigger pada sebuah tabel
2. Mahasiswa dapat mengimplementasikan pendistribusian basis data melalui pendekatan replikasi (one-way)

C. Teori Dasar

Trigger adalah suatu objek database yang merupakan aksi atau prosedur yang terjadi jika terjadi perubahan pada suatu row.

Berikut ini adalah cara membuat trigger.

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

Misalnya kita akan membuat trigger after delete pada tabel produk. Jika kita hapus salah satu data produk maka TRIGGER akan secara otomatis memindahkan data yang terhapus ke tabel produk_hapus. Terlebih dahulu kita buat tabel produk_hapus.

```
MariaDB [(none)]> use orderentry;
Database changed
MariaDB [orderentry]> create table produk_hapus as select * from products where 1=2;
Query OK, 0 rows affected (0.43 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [orderentry]> desc produk_hapus;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| prod_id    | varchar(10)  | NO   |     | NULL    |       |
| vend_id    | char(4)      | NO   |     | NULL    |       |
| prod_name  | varchar(25)  | NO   |     | NULL    |       |
| prod_price | int(11)      | NO   |     | NULL    |       |
| prod_desc  | varchar(255) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

Tambahkan kolom tgl hapus dan user untuk merekam kapan data dihapus dan siapa yang menghapus.

```
MariaDB [orderentry]> DESC PRODUK_HAPUS;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| prod_id    | varchar(10)  | NO   |     | NULL    |       |
| vend_id    | char(4)      | NO   |     | NULL    |       |
| prod_name  | varchar(25)  | NO   |     | NULL    |       |
| prod_price | int(11)      | NO   |     | NULL    |       |
| prod_desc  | varchar(255) | YES  |     | NULL    |       |
| tgl_hapus  | date         | YES  |     | NULL    |       |
| user       | varchar(30)  | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
7 rows in set (0.03 sec)
```

Setelah itu kita buat trigger untuk eksekusi jika terjadi penghapusan pada tabel produk

```
MariaDB [orderentry]> CREATE TRIGGER hapus_barang AFTER DELETE
    ->     ON products FOR EACH ROW
    -> BEGIN
    ->   INSERT INTO produk_hapus
    ->         (       prod_id,
    ->                 vend_id,
    ->                 prod_name,
    ->                 prod_price,
    ->                 prod_desc,
    ->                 tgl_hapus,
    ->                 nama_user
    ->         )
    ->   VALUES
    ->         (       OLD.prod_id,
    ->                 OLD.vend_id,
    ->                 OLD.prod_name,
    ->                 OLD.prod_price,
    ->                 OLD.prod_desc,
    ->                 SYSDATE(),
    ->                 CURRENT_USER
    ->         );
    -> end;
    ->
Query OK, 0 rows affected (0.22 sec)
```

Setelah trigger kita buat, lakukan pengujian dengan cara

1. Hapus salah satu row dari table products

2. Buka table produk_hapus, perhatikan tabel tersebut terisi data dari products yang dihapus.

D. Kegiatan Praktikum

1. Instrument

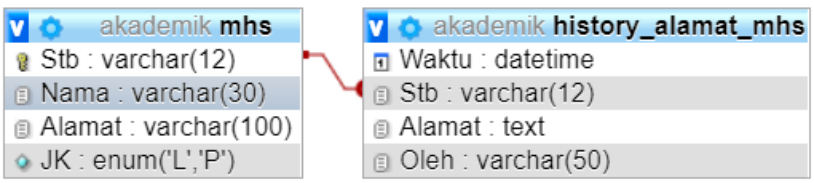
a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MySQL versi 5.0 atau di atasnya (sebagai engine basis data)

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

3. Studi Kasus

a) Buat sebuah database dengan nama **Akademik**



b) Isikan data pada tabel mahasiswa sesuai gambar dibawah ini:

Stb	Nama	Alamat	JK
13020170001	Septario Sagita	Jl. aa No.1	L
13020170002	Taurio Taufik N	Jl. aa No.2	L
13020170003	Irfan Fajarudy	Jl. aa No.3	L

c) Lakukan percobaan berikut.

1. Membuat sebuah trigger yang akan menyimpan history alamat. Jika sebuah alamat berubah, maka alamat lama harus disimpan ke tabel history_alamat_mhs.

- Pembuatan trigger

```
delimiter //
create trigger trig_update_mhs
after update on mhs
for each row
begin
if old.alamat <> new.alamat THEN
insert into history_alamat_mhs
values (now(),old.Stb, old.Alamat, user());
end if;
end//
```

- Penggunaan trigger

```
update mhs set Alamat='BTP' WHERE Stb='13020170001';

update mhs set Alamat='Antang' WHERE Stb='13020170002';

update mhs set Alamat='Baruga' WHERE Stb='13020170003';
```

```
MariaDB [Akademik]> select * from mhs;
+-------------+-----------------+--------+------+
| Stb         | Nama            | Alamat | JK   |
+-------------+-----------------+--------+------+
| 13020170001 | Septario Sagita | BTP    | L    |
| 13020170002 | Taurio Taufik N | Antang | L    |
| 13020170003 | Irfan Fajarudy  | Baruga | L    |
+-------------+-----------------+--------+------+
3 rows in set (0.00 sec)

MariaDB [Akademik]> select * from history_alamat_mhs;
+---------------------+-------------+------------+----------------+
| Waktu               | Stb         | Alamat     | Oleh           |
+---------------------+-------------+------------+----------------+
| 2018-09-10 21:52:45 | 13020170001 | Jl.aa No.1 | root@localhost |
| 2018-09-10 21:52:45 | 13020170002 | Jl.aa No.2 | root@localhost |
| 2018-09-10 21:52:45 | 13020170003 | Jl.aa No.3 | root@localhost |
+---------------------+-------------+------------+----------------+
3 rows in set (0.00 sec)
```

2. Membuat sebuah trigger yang akan menghapus data pada table history_alamat_mhs ketika ada penghapus pada tabel mhs.

- Pembuatan trigger

```
delimiter //
create trigger trig_delete_mhs
before delete on mhs
for each row
begin
delete from history_alamat_mhs where Stb=old.Stb;
end//
```

- Penggunaan trigger

```
DELETE FROM Mhs WHERE Stb='13020170003';
```

```
MariaDB [Akademik]> select * from mhs;
+-------------+-----------------+--------+------+
| Stb         | Nama            | Alamat | JK   |
+-------------+-----------------+--------+------+
| 13020170001 | Septario Sagita | BTP    | L    |
| 13020170002 | Taurio Taufik N | Antang | L    |
+-------------+-----------------+--------+------+
2 rows in set (0.00 sec)

MariaDB [Akademik]> select * from history_alamat_mhs;
+---------------------+-------------+------------+----------------+
| Waktu               | Stb         | Alamat     | Oleh           |
+---------------------+-------------+------------+----------------+
| 2018-09-10 21:52:45 | 13020170001 | Jl.aa No.1 | root@localhost |
| 2018-09-10 21:52:45 | 13020170002 | Jl.aa No.2 | root@localhost |
+---------------------+-------------+------------+----------------+
2 rows in set (0.00 sec)
```

a) Siapkan PC/LAPTOP (Min.2 Buah), 1 dijadikan sebagai MASTER dan 1 laptop berikutnya jadikan SLAVE.

b) Hubungkan antar Laptop dengan menggunakan Kabel LAN atau jaringan internet. Setting Masing-masing IP. PASTIKAN Masing-masing Laptop terhubung dengan PING antar IP.

IP MASTER :192.168.1.1

IP SLAVE :192.168.1.2

c) Membuat Database

Database yang akan di replikasi antar master dan slave haruslah sama dalam nama dan strukturnya tabelnya:

```
mysql> create database test_replication;
mysql> use test_replication;
mysql> create table mahasiswa  (nim varchar (15) primary key not null, nama varchar (40), alamat text);
```

d) Pada Server Master :

1. Matikan firewall (Start | Settings | Control Panel | Windows Firewall | Off
2. Edit File My.Ini (C:\Xammp\Mysql\Bin\My.Ini) Menggunakan Notepad++
3. Pada [MYSQLD] Tambahkan :

```
# The MySQL server
[mysqld]
server-id=1
log-bin=mysql-bin
binlog-do-db= test_replication
```

Cari server-id=1, jika ada selain yang diatas diberi tanda # (#server-id=1). pastikan server-id=1 hanya ada satu yang aktif dibawah [mysqld]. Setelah itu simpan dan RESTART MYSQL di XAMPP CONTROL PANEL. jika error ulangi langkah yang diatas.

4. Masuk Ke Mysql (Start|Run|Cmd|Ok)
   C:\cd xammp\mysql\bin (Command Prompt)

mysql -u root –p
password : root *bila tidak memakai password langsung tekan enter

```
mysql> CREATE USER 'replicator_user' identified by 'replicator_user';
mysql> Grant replication slave on *.* to 'replicator_user'@'%' identified by 'replicator_user';
mysql> FLUSH privileges;
mysql> SHOW MASTER STATUS;
```

SHOW MASTER STATUS;

```
+--------------------+----------+------------------+------------------+
| File               | Position | Binlog_Do_DB     | Binlog_Ignore_DB |
+--------------------+----------+------------------+------------------+
| andrey-bin.000001 |      107 | test_replication |                  |
+--------------------+----------+------------------+------------------+
1 row in set (0.00 sec)
```

e) Pada Server Slave :

1. Matikan firewall (Start | Settings | Control Panel | Windows Firewall | Off)
2. Edit File My.Ini (C:\Xammp\Mysql\Bin\My.Ini) Menggunakan Notepad++
3. Pada [MYSQLD] Tambahkan :

```
# The MySQL server
[mysqld]
server-id=2
```

Cari server-id=2, jika ada diberi tanda # (#server-id=2). Setelah itu simpan dan RESTART MYSQL di XAMPP CONTROL PANEL. jika error ulangi langkah yang diatas.

4. Masuk ke Mysql (Start|Run|Cmd|Ok)

C:\cd xammp\mysql\bin (Command Prompt)
mysql -u root –p
password : root *bila tidak memakai password langsung tekan enter

```
mysql> stop slave;
mysql> CHANGE MASTER TO MASTER_HOST='192.168.1.1',
       MASTER_USER='replicator_user',
         MASTER_PASSWORD='replicator_user',
         MASTER_PORT=3306,
         MASTER_LOG_FILE='andrey-bin.000001',
         MASTER_LOG_POS=107;
```

Pada tahap ini ialah melihat status yang ada pada komputer Client, jika konfigurasi berhasil maka akan tampil dua status yes, seperti yang ada pada gambar disamping ini :

Untuk mencoba replikasi, lakukan perubahan data di salah satu tabel pada database yang ada di MASTER. Lalu Refresh database di SLAVE, maka data yang ada di tabel SLAVE juga otomatis akan berubah sama dengan yang di MASTER.

```
mysql> start slave;
Query OK, 0 rows affected (0.01 sec)

mysql> show slave status\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.1.2
                  Master_User: replicator_user
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: andrey-bin.000006
          Read_Master_Log_Pos: 107
               Relay_Log_File: mysql-relay-bin.000002
                Relay_Log_Pos: 254
        Relay_Master_Log_File: andrey-bin.000006
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
```

LEMBAR EVALUASI PRAKTIKUM

Evaluasi Praktikum 4:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat membuat Trigger dengan event insert, Update, Delete					
2.	Dapat membuat server Master, Server Slave					

Catatan Asisten:

Dosen : _____

Asisten 1 : _____

Asisten 2 : _____

TINGKAT KEBERHASILAN