

TIM PENYUSUN

MODUL PRAKTIKUM BASIS DATA

MY SQL



**LABORATORIUM BASIS DATA DAN JARINGAN
PRODI TEKNIK INFORMATIKA DAN PRODI SISTEM INFORMASI
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS MERCU BUANA YOGYAKARTA
2016**

Identitas Praktikan

No. Presensi Praktikan :

Nama :

NIM :

Alamat :

.....

.....

Telp/HP :

Hari Praktikum :

Jam Praktikum :

Mengetahui, Yogyakarta, 20....
Asisten Praktikan.

KATA PENGANTAR

Bismilahirrahmanirakhim

Syukur kehadirat Allah SWT atas pertolongan dan kasih sayang Nya sehingga penulis bisa merampungkan penyusunan 'Modul Praktikum Basis Data' ini. Mengutip salah satu kata bijak imam bukhori 'Bersiaplah berlelah lelah dalam belajar jika engkau tidak ingin terhina dalam kebodohan'. Maka atas semangat saling belajar lah modul ini bisa terpenuhi. Terima kasih untuk beliau Bapak A.Sidiq Purnomo selaku penyusun modul praktikum versi pertama sekaligus supervisor modul ini atas saran dan masukannya.

Modul praktikum ini bertujuan untuk pendukung perkuliahan Basis Data dan Sistem Basis Data. Dicitak terbatas untuk Fakultas Teknologi Informasi Universitas Mercu Buana Yogyakarta. Modul ini adalah edisi kedua dengan penambahan beberapa bagian diantaranya, transformasi ER ke diagram relationship, single row function, advance query, trigger, view.

Fokus pembelajaran adalah best practise sharing dengan memperbanyak contoh dan studi kasus. Dengan harapan bisa lebih mudah dimengerti. Serta harapanya bisa memenuhi prasyarat mata kuliah pemrograman web dengan skill penguasaan perancangan basis data.

Kesempurnaan hanya milik Allah SWT, tak ada gading yang tak retak; Penulis sangat terbuka untuk segala masukan demi perbaikan perkuliahan praktikum basis data ke depan. Masukan bisa langsung atau via email arita@mercubuana-yogya.ac.id. Terakhir semoga karya ini bisa memberi kemanfaatan untuk civitas akademika FTI pada khususnya. Terima kasih

Wassalamu 'alaykum Wr.Wb

Yogyakarta, 5 Februari 2016

Arita Witanti, S.T., M.T

DAFTAR ISI

TABLE OF CONTENTS

- KATA PENGANTAR..... iii
- DAFTAR ISI..... iv
- Tata Tertib Praktikum viii
- 1 BAB 1 – REVIEW KONVERSI ENTITY RELATIONSHIP (ER) DIAGRAM KE SKEMA RELASI ix
 - 1.1 Identitas ix
 - 1.2 Test Awal..... 9
 - 1.3 Konversi ER Ke Skema Relasi Dan Diagram Relationship..... 10
 - 1.4 Studi Kasus Skema Pembayaran Apotik..... 17
 - 1.5 Test Akhir 19
- 2 BAB 2 – PENGANTAR BASIS DATA-DDL 20
 - 2.1 Identitas 20
 - 2.2 Test Awal..... 20
 - 2.3 Dbms My SQL..... 20
 - 2.4 Instalasi My SQL 21
 - 2.5 Aplikasi Server My SQL..... 21
 - 2.6 Mengakses My SQL..... 22
 - 2.7 Client My SQL..... 23
 - 2.8 Type Data My SQL..... 24
 - 2.9 Database Relational 24
 - 2.10 Data definition language (ddl) 24
 - 2.10.1 Membuat Database 24
 - 2.10.2 Melihat seluruh Database..... 25
 - 2.10.3 Mengakses Database 25
 - 2.10.4 Menghapus Database 25
 - 2.11 Test Akhir 26
- 3 BAB 3 – DATA DEFINITION LANGUANGE (DDL) 27
 - 3.1 Identitas 27
 - 3.2 Test Awal..... 27
 - 3.3 SQL 28

3.4	Membuat table	28
3.4.1	Create Table	28
3.5	Constraint Table	29
3.5.1	Penerapan constraint pada Skema Order Entry	30
3.6	Nilai otomatis dan nilai default	32
3.6.1	Nilai otomatis / Auto Increment	32
3.6.2	Nilai default	33
3.7	Type : innodb dan xtradb	33
3.8	Test Akhir	34
4	BAB 4 – ALTER, MODIFY , DROP , RENAME (DDL)	35
4.1	Identitas	35
4.2	Test Awal	35
4.3	Alter Table	35
4.3.1	Merubah Struktur Table	35
4.4	Drop	38
4.5	Test Akhir	39
5	BAB 5 – DATA MANIPULATION LANGUAGE (DML)	40
5.1	Identitas	40
5.2	Test Awal	40
5.3	Insert table	40
5.4	Query Sederhana	41
5.4.1	Menampilkan Data Secara Keseluruhan	41
5.5	Update table	43
5.6	Delete Data	43
5.7	Query Dengan Kondisi	44
5.8	Test Akhir	45
6	BAB 6 – SINGLE ROW FUNCTION (DML)	47
6.1	Identitas	47
6.2	Test Awal	47
6.3	Urutan Data (acs, desc, Order by)	47
6.3.1	Mengurutkan Data	47
6.4	Agregrate Function	48
6.5	Operator Between, In, like	49

6.6	Ekspresi query	50
6.7	Fungsi waktu	51
6.8	Test Akhir	52
7	BAB 7 – JOIN DAN SUBQUERY (DML).....	54
7.1	Identitas	54
7.2	Test Awal.....	54
7.3	Selecting data dengan join table.....	54
7.4	Clausa Join On Alias.....	56
7.5	Join 3 table atau lebih	56
7.6	Test Akhir	56
8	BAB 8 – ADVANCE JOIN, TRIGGER, VIEW (DML)	58
8.1	Identitas	58
8.2	Test Awal.....	58
8.3	Outer Join.....	58
8.4	Left Join	58
8.5	Right Join.....	59
8.6	Self Join	60
8.7	Trigger	60
8.8	View	62
8.9	Test Akhir	62
9	BAB 8 – DATA CONTROL LANGUAGE (DCL).....	64
9.1	Identitas	64
9.2	Test Awal.....	64
9.3	User	64
9.4	Hak akses user.....	65
9.5	Grant	66
9.6	Revoke.....	67
9.7	Test Akhir	68
10	BAB 10 – PHP MY ADMIN.....	69
10.1	Identitas	69
10.2	Test Awal.....	69
10.3	SQL	69
10.4	Relasi Table	71

10.5	Test Akhir	72
11	BAB 11 – STUDI KASUS SQL.....	73
11.1	Identitas	73
11.2	Studi Kasus	73
11.3	Test Akhir	74
12	BAB 12 – STUDI KASUS NORMALISASI	75
12.1	Identitas	75
12.2	Studi Kasus	lxxv
12.3	Test Akhir	lxxv
13	Bibliography	lxxvii

Tata Tertib Praktikum

Tata Tertib berikut untuk kebaikan bersama

1. Praktikan hadir 5 menit sebelum praktikum dimulai.
2. Toleransi keterlambatan adalah 15 menit. Bila praktikan terlambat lebih dari 15 menit tidak diperkenankan mengikuti praktikum.
3. Praktikan berpakaian bebas rapi dan sopan. Praktikan tidak diperkenankan memakai kaos oblong, topi, sandal. Bagi cowok dilarang memakai anting – anting.
4. Praktikan Mengisi daftar hadir kemudian mengumpul kartu praktikum untuk diparaf asisten jaga.
5. Praktikan duduk sesuai dengan yang ditentukan dan menyiapkan modul praktikum serta buku penunjang praktikum.
6. Selama praktikum berlangsung, semua praktikan wajib menjaga ketertiban praktikum.
7. Praktikan tidak diperkenankan melakukan koneksi dengan internet selama praktikum berlangsung.
8. Perangkat komunikasi dalam kondisi silent dimasukkan ke dalam tas.
9. Praktikan mengerjakan tugas di setiap akhir praktikum.
10. Praktikan Menjaga kebersihan peralatan laboratorium dan laboratorium.
11. Praktikan Tidak diperkenankan melakukan konfigurasi ulang terhadap hardware dan software di laboratorium.
12. Selesai praktikum, praktikan mematikan komputer dan merapikan peralatan.
13. Diberikan kesempatan inhal 1 kali pada akhir praktikum (praktikan berhalangan hadir dan memberi keterangan yang sah).
14. Pada akhir praktikum, praktikan diwajibkan mengumpulkan laporan praktikum.
15. Pada akhir praktikum akan diadakan responsi (di sesuaikan dengan jadwal ujian dan praktikan harus hadir 100% untuk dapat mengikuti responsi).
16. Bagi yang melanggar tata tertib akan dikenakan sangsi.

1 BAB 1 – REVIEW KONVERSI ENTITY RELATIONSHIP (ER) DIAGRAM KE SKEMA RELASI

1.1 IDENTITAS

Kompetensi

1. Memantapkan pemahaman praktikan tentang cara mengkonversi ER ke skema relasi dan dari skema relasi ke table.
2. Kedua memudahkan proses transformasi table level view ke level fisik pada basis data

Topik

1. Konversi ER ke Skema Relasi dan Diagram Relationship
2. Studi Kasus Skema Order Entry.

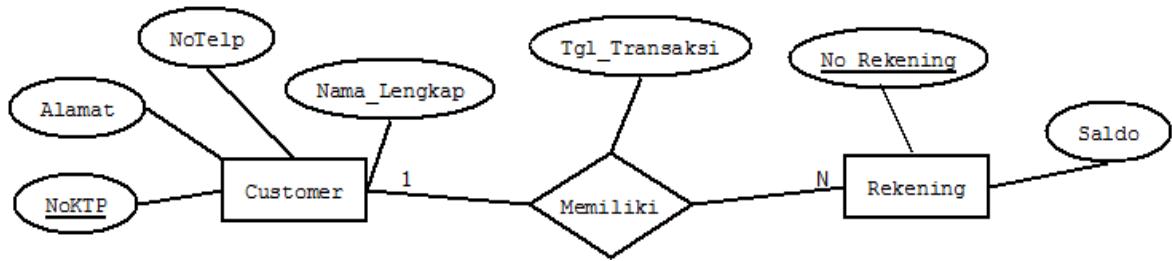
1.2 TEST AWAL

Kerjakan test awal dibawah ini dengan bolpoint/spidol/ pensil 5 warna

1. Menurut anda apakah entitas, atribut, primary key itu, bagaimana simbolnya ?

2. Menurut anda apakah relasi dan kardinalitas itu, bagaimana simbolnya ?

3. Perhatikan gambar potongan ER diagram dibawah ini !
Lalu tuliskan mana sajakah yang termasuk entitas, atribut, primary key, relasi, dan kardinalitas ! Gunakan warna 5 warna berbeda untuk menandai . Beri Keterangan



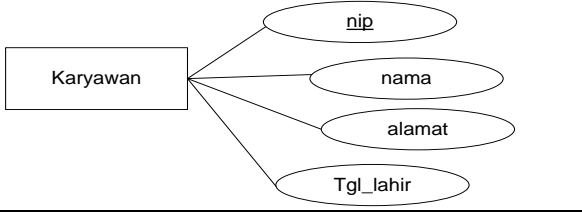
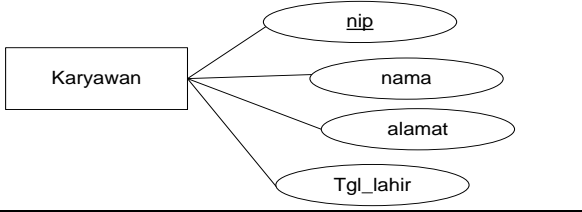
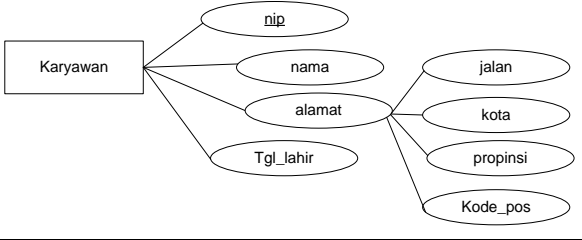
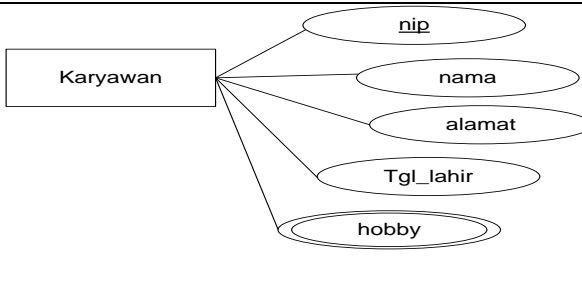
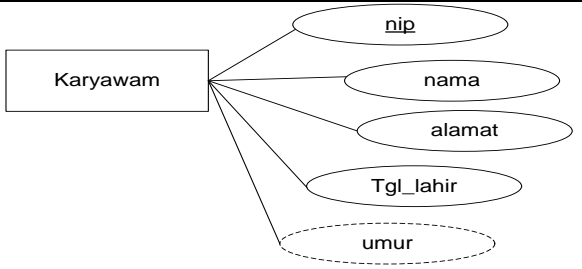
4. Perhatikan gambar potongan ER no 3 pada gambar 3! Konversikan menjadi skema relasi !

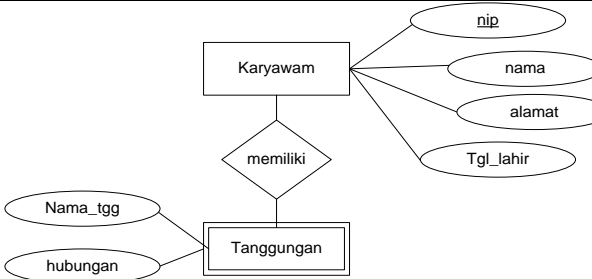
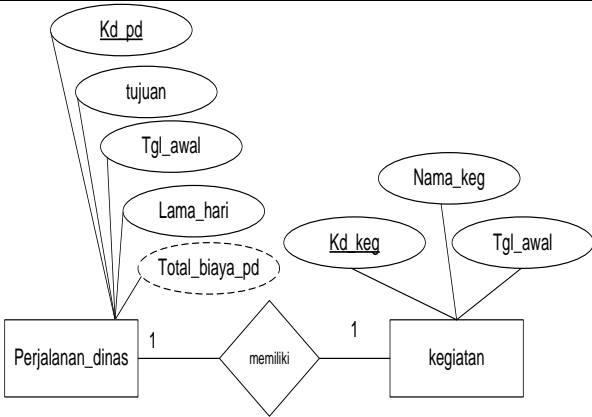
5. Konversikan skema relasi dari ER no 3 menjadi diagram relationship ! Amati hasil diagram relationship anda, ?

1.3 KONVERSI ER KE SKEMA RELASI DAN DIAGRAM RELATIONSHIP

Sebelum memulai membangun database di level fisik di komputer dalam bentuk file, terlebih dahulu kita harus memahami dalam mentransformasikan level view dari diagram ER menjadi diagram relationship dan tabel.

Berikut ini adalah aturan konversi ER diagram ke diagram relationship dan table

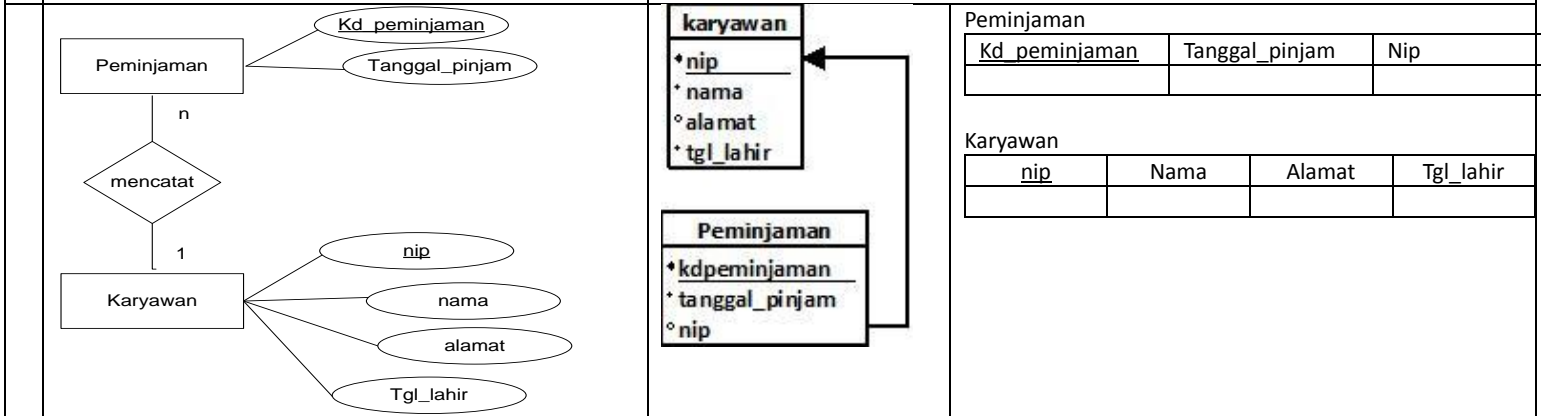
NO	ERD → DIAGRAM RELATIONSHIP	ATURAN CONTOH ER → DIAGRAM RELATIONSHIP → TABLE														
1		<p>Setiap entitas kuat (<i>strong entity</i>) menjadi satu tabel dan setiap simple atribut menjadi kolom [1, p. 42, 2, p. 244]. Nama entitas menjadi nama tabel. Nama atribut menjadi nama kolom. Atribut kunci menjadi Primary Key.</p>														
		<div style="display: flex; justify-content: space-between;"> <div data-bbox="781 468 915 632"> <p>Karyawan</p> <ul style="list-style-type: none"> *nip *nama oalamat otgl_lahir </div> <div data-bbox="1013 468 1490 632"> <p>Tabel_Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Nip</th> <th style="width: 25%;">Nama</th> <th style="width: 25%;">Alamat</th> <th style="width: 25%;">Tgl_lahir</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	Nip	Nama	Alamat	Tgl_lahir										
Nip	Nama	Alamat	Tgl_lahir													
2	<p>COMPOSIT ATRIBUT</p>	<p>Setiap entitas kuat (<i>strong entity</i>) menjadi satu tabel dan setiap simple atribut menjadi kolom [1, p. 42, 2, p. 244]. Nama entitas menjadi nama tabel. Nama atribut menjadi nama kolom. Atribut kunci menjadi Primary Key Atribut alamat tidak menjadi kolom , perhatikan !</p>														
		<div style="display: flex; justify-content: space-between;"> <div data-bbox="781 915 915 1152"> <p>Karyawan</p> <ul style="list-style-type: none"> *nip *nama *tgl_lahir *jalan *kota opropinsi okodepos </div> <div data-bbox="1013 915 1580 1035"> <p>Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 12.5%;">nip</th> <th style="width: 12.5%;">Nama</th> <th style="width: 12.5%;">Jalan</th> <th style="width: 12.5%;">Kota</th> <th style="width: 12.5%;">Prop insi</th> <th style="width: 12.5%;">Kode_pos</th> <th style="width: 12.5%;">Tgl_l ahir</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	nip	Nama	Jalan	Kota	Prop insi	Kode_pos	Tgl_l ahir							
nip	Nama	Jalan	Kota	Prop insi	Kode_pos	Tgl_l ahir										
3	<p>MULTIVALUE ATRIBUT</p>	<p>Multivalued attribute menjadi tabel tersendiri [1, p. 58, 2, pp. 224-225], sehingga entitas kuat yang memiliki multivalued attribute menjadi 2 tabel</p>														
		<div style="display: flex; justify-content: space-between;"> <div data-bbox="724 1268 971 1551"> <p>karyawan</p> <ul style="list-style-type: none"> *nip *nama oalamat *tgl_lahir <p>Karyawan_hobby</p> <ul style="list-style-type: none"> *nip *hobby </div> <div data-bbox="1013 1268 1580 1507"> <p>Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Nip</th> <th style="width: 25%;">Nama</th> <th style="width: 25%;">Alamat</th> <th style="width: 25%;">Tgl_lahir</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> <p>Hobby_Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Nip</th> <th style="width: 50%;">Hobby</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	Nip	Nama	Alamat	Tgl_lahir					Nip	Hobby				
Nip	Nama	Alamat	Tgl_lahir													
Nip	Hobby															
4	<p>DERIVATE ATRIBUT</p>	<p>Setiap derivate atribut menjadi kolom</p>														
		<div style="display: flex; justify-content: space-between;"> <div data-bbox="776 1593 915 1797"> <p>karyawan</p> <ul style="list-style-type: none"> *nip *nama oalamat *tgl_lahir *umur </div> <div data-bbox="1013 1593 1515 1686"> <p>Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 16.6%;">nip</th> <th style="width: 16.6%;">Nama</th> <th style="width: 16.6%;">Alamat</th> <th style="width: 16.6%;">Tgl_lahir</th> <th style="width: 16.6%;">Umur</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	nip	Nama	Alamat	Tgl_lahir	Umur									
nip	Nama	Alamat	Tgl_lahir	Umur												
5	<p>ENTITAS LEMAH</p>	<p>Setiap entitas lemah menjadi tabel</p>														

	<p>setiap simple atribut menjadi kolom. Atribut kunci pada entitas kuat yang berelasi dengan entitas menjadi kolom foreign key [1, 2, p. 245].</p> <p>Perhatikan atribut nip di tabel karyawan, menjadi foreign key (FK) di tabel tanggungan.</p>																																				
	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>karyawan</p> <ul style="list-style-type: none"> *nip *nama ◦alamat *tgl_lahir <p>Tanggungan</p> <ul style="list-style-type: none"> *nip *Nama_tanggungan ◦Hubungan </div> <div style="width: 50%;"> <p>Karyawan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Nip</th> <th>Nama</th> <th>Alamat</th> <th>Tgl_lahir</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> <p>Tanggungan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Nip</th> <th>Nama_tgg</th> <th>hubungan</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	Nip	Nama	Alamat	Tgl_lahir					Nip	Nama_tgg	hubungan																									
Nip	Nama	Alamat	Tgl_lahir																																		
Nip	Nama_tgg	hubungan																																			
<p>5 RELASI SATU KE SATU</p>	<p>Full participant – Full participant</p> <p>Setiap entitas kuat (<i>strong entity</i>) menjadi satu tabel dan simple attributnya menjadi kolom. Tabel yang terbentuk ada dua (2) buah.</p> <p>Atribut kunci pada salah satu entitas, menjadi kolom foreign key pada entitas lain. [2, pp. 245-246]</p>																																				
	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Alternatif 1:</p> <p>Perjalanan Dinas</p> <ul style="list-style-type: none"> *kd_pd *tujuan ◦tgl_awal *lama_hari ◦total_biaya_pd ◦kd_keg <p>Kegiatan</p> <ul style="list-style-type: none"> *kd_keg *nama_keg ◦tgl_awal <p>Alternatif 2:</p> <p>Perjalanan Dinas</p> <ul style="list-style-type: none"> *kd_pd *tujuan ◦tgl_awal *lama_hari ◦total_biaya_pd ◦kd_keg <p>Kegiatan</p> <ul style="list-style-type: none"> *kd_keg *nama_keg ◦tgl_awal ◦kd_pd </div> <div style="width: 50%;"> <p>Alternatif 1:</p> <p>Perjalanan_dinas</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Kd_pd</th> <th>Tujuan</th> <th>Tgl_awal</th> <th>Lama_hari</th> <th>Total_biaya_pd</th> <th>Kd_keg</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> <p>Kegiatan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Kd_keg</th> <th>Nama_keg</th> <th>Tgl_awal</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> <p>Alternatif 2:</p> <p>Perjalanan_dinas</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Kd_pd</th> <th>Tujuan</th> <th>Tgl_awal</th> <th>Lama_hari</th> <th>Total_biaya_pd</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> <p>Kegiatan</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Kd_keg</th> <th>Nama_keg</th> <th>Tgl_awal</th> <th>Kd_pd</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table> </div> </div>	Kd_pd	Tujuan	Tgl_awal	Lama_hari	Total_biaya_pd	Kd_keg							Kd_keg	Nama_keg	Tgl_awal				Kd_pd	Tujuan	Tgl_awal	Lama_hari	Total_biaya_pd						Kd_keg	Nama_keg	Tgl_awal	Kd_pd				
Kd_pd	Tujuan	Tgl_awal	Lama_hari	Total_biaya_pd	Kd_keg																																
Kd_keg	Nama_keg	Tgl_awal																																			
Kd_pd	Tujuan	Tgl_awal	Lama_hari	Total_biaya_pd																																	
Kd_keg	Nama_keg	Tgl_awal	Kd_pd																																		
<p>6 RELASI ONE TO MANY (1 ke N)</p>	<p>Setiap entitas kuat (<i>strong entity</i>) menjadi satu tabel dan simple</p>																																				

atributnya menjadi kolom. Tabel yang terbentuk ada dua (2) buah.

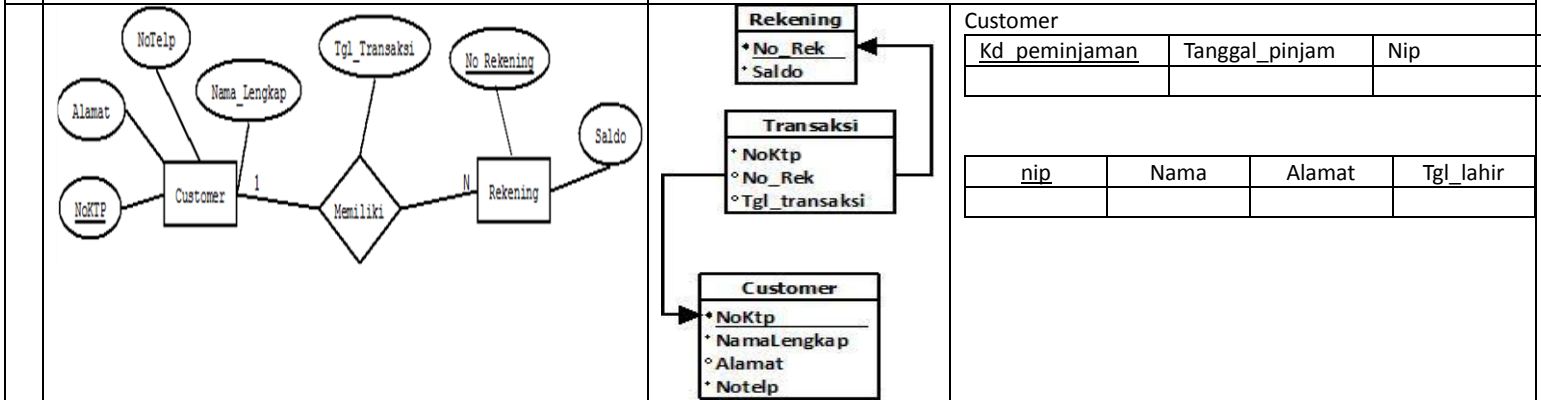
Atribut kunci pada entitas yang memiliki kardinalitas satu (*one*), menjadi kolom **foreign key** pada entitas yang memiliki kardinalitas banyak (*many*). [2, p. 246] (sedikit ikut ke yg banyak)

Perhatikan atribut **nip**, menjadi **foreign key (FK)** di tabel Peminjaman



7 RELASI ONE TO MANY (1 KE N) DENGAN ATRIBUT PADA RELASI

One-To-Many (ada atribut di relasi)
Menjadi 3 table, tabel dari entitas 1, entitas 2 dan hasil relasinya, pada tabel relasi terdapat atribut foreign key sebagai hasil relationship [2, p. 251].



8 RELASI MANY TO MANY (N KE N)

#Setiap entitas kuat (*strong entity*) menjadi satu tabel dan simple atributnya menjadi kolom.

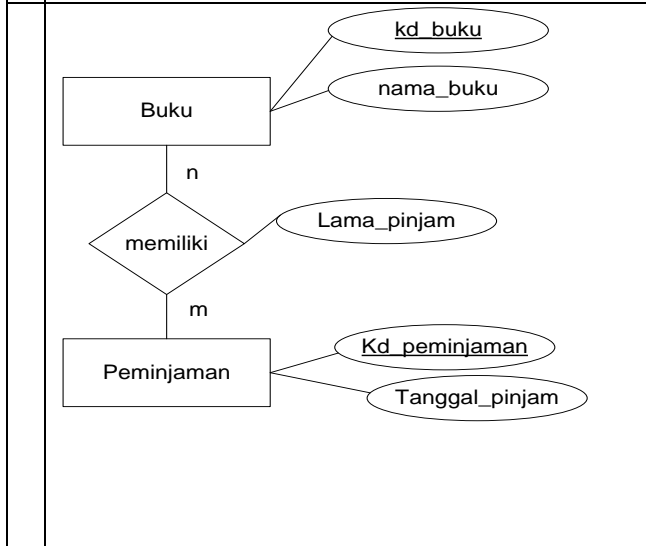
#Relasi dari kardinalitas *many-to-many* menjadi tabel, dan simple atribut pada relasi menjadi kolom.

#Atribut kunci pada entitas yang memiliki kardinalitas satu (*many*), menjadi kolom **foreign key** pada relasi.

#Jika diperlukan dapat ditambahkan kolom baru sebagai **primary**

key pada tabel dari relasi ini (lihat kolom id_det_pin).

#Tabel yang terbentuk ada tiga (3) buah.
Perhatikan atribut **kd_buku** dan **kd_peminjaman** menjadi **foreign key (FK)** di tabel detail_peminjaman.



Peminjaman

<u>Kd_peminjaman</u>	Tanggal_pinjam

Buku

<u>Kd_buku</u>	Nama_buku

Detail_peminjaman

<u>Kd_peminjaman</u>	<u>Kd_buku</u>	Lama_pinjam

Atau

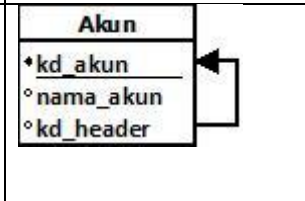
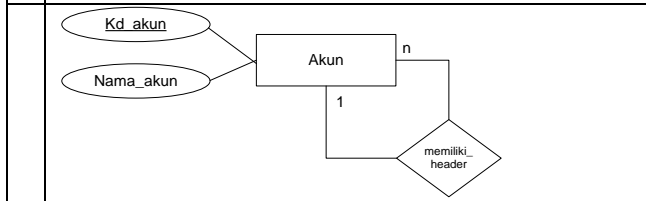
Detail_peminjaman

<u>Id_det_pin</u>	<u>Kd_peminjaman</u>	<u>Kd_buku</u>	Lama_pinjam

9 RELASI UNARY

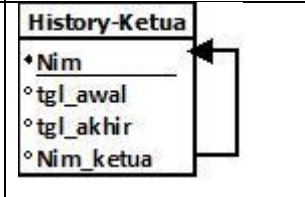
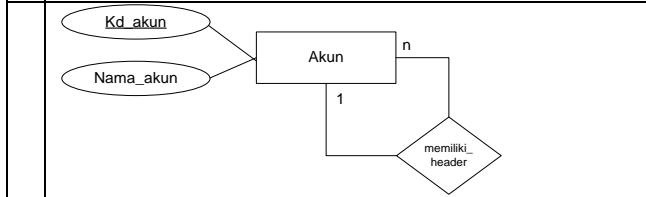
Satu ke satu (1 to 1)
Setiap entitas kuat (*strong entity*) menjadi satu tabel dan simple attributnya menjadi kolom.
Atribut kunci menjadi kolom **primary key** dengan nama kolom seperti nama atribut kunci.
Atribut kunci menambahkan kolom **foreign key** dengan nama kolom seperti nama sesuai relasi.
Tabel yang terbentuk ada satu buah.

Many-To-Many (N ke N)
Relasi dari kardinalitas *many-to-many* menjadi tabel, dan simple atribut menjadi kolom.



Akun

<u>Kd_akun</u>	Nama_Akun	Kd_header



History_ketua

<u>Nim</u>	Tgl_awal	Tgl_akhir	Nim_ketua

1 RELASI TERNARY

Setiap entitas kuat (*strong entity*) menjadi satu tabel dan simple

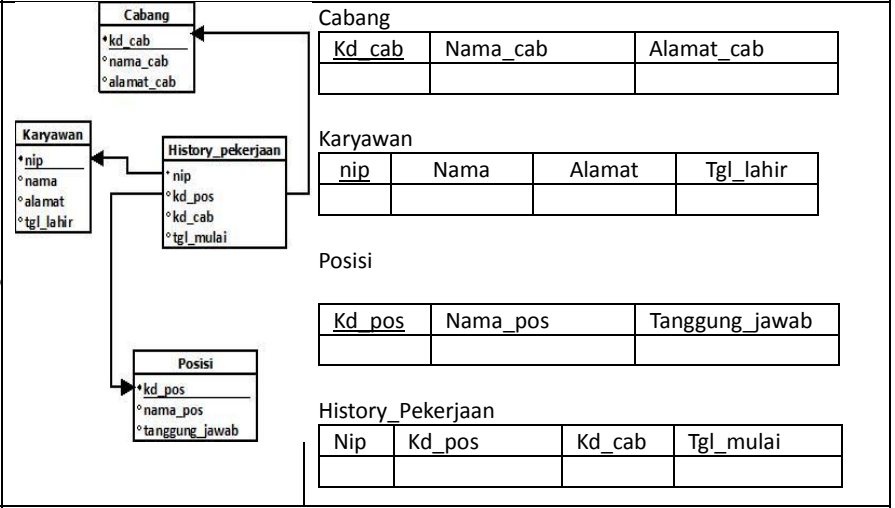
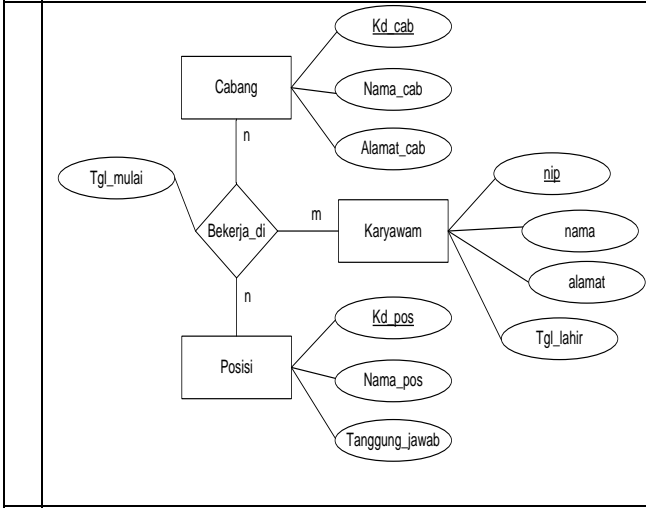
0

atributnya menjadi kolom. Atribut kunci menjadi kolom **primary key** dengan nama kolom seperti nama atribut kunci.

Relasi dari ternary relationship menjadi tabel, dan simple atribut menjadi kolom.

Atribut kunci pada entitas yang berelasi menjadi kolom **foreign key** pada relasi.

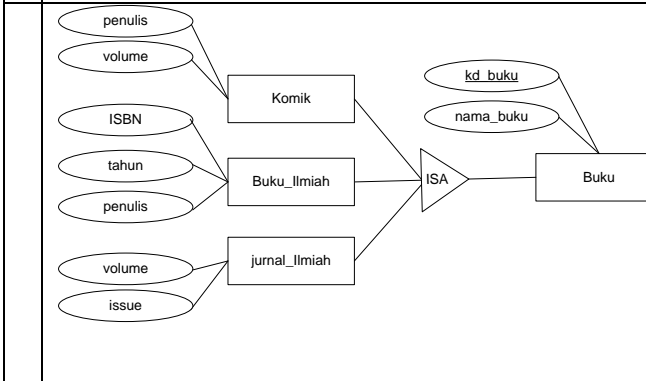
Tabel yang terbentuk ada empat (4) buah.



1 GENSPEC (GENERALISASI DAN SPESIALIASI)
1 Metoda 1

Entitas **superclass** menjadi tabel dengan simple atribut menjadi kolom. Atribut kunci menjadi **primary key**.

Entitas **subclass** menjadi tabel dengan simple atribut menjadi kolom dan atribut kunci dari superclass menjadi kolom **primary key** pada subclass.



<p>Metoda 2</p> <p>Entitas subclass menjadi tabel dengan simple atribut menjadi kolom dan atribut dari entitas superclass menjadi kolom pada subclass. Atribut kunci dari superclass menjadi kolom primary key pada subclass</p>	<p>Buku_Ilmiah</p> <ul style="list-style-type: none"> *kd_buku °ISBN °Tahun °Penulis <p>Komik</p> <ul style="list-style-type: none"> *kd_buku °penulis °volume <p>Jurnal_Ilmiah</p> <ul style="list-style-type: none"> *kd_buku °volume *Issue 	<p>Komik</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th>Kd_buku</th> <th>Nama_buku</th> <th>Penulis</th> <th>Volume</th> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </table> <p>Buku_ilmiah</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th>Kd_buku</th> <th>Nama_buku</th> <th>ISBN</th> <th>Tahun</th> <th>Penulis</th> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </table> <p>Jurnal_ilmiah</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th>Kd_buku</th> <th>Nama_buku</th> <th>Volume</th> <th>Issue</th> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </table>	Kd_buku	Nama_buku	Penulis	Volume					Kd_buku	Nama_buku	ISBN	Tahun	Penulis						Kd_buku	Nama_buku	Volume	Issue				
Kd_buku	Nama_buku	Penulis	Volume																									
Kd_buku	Nama_buku	ISBN	Tahun	Penulis																								
Kd_buku	Nama_buku	Volume	Issue																									

1 AGREGASI

2

Setiap entitas kuat (*strong entity*) menjadi satu tabel dan simple attributnya menjadi kolom. Atribut kunci menjadi kolom **primary key** dengan nama kolom seperti nama atribut kunci.

Relasi antara entitas dengan entitas yang beragregasi, memiliki **foreign key** dari semua entitas yang berhubungan.

Buku

- *kd_buku
- °nama_buku

detail_peminjaman

- °kd_buku
- °kd_peminjaman
- *lama_pinjam

Peminjaman

- *kd_peminjaman
- *Tanggal_pinjam

Detail_pengembalian

- *kd_pengembalian
- °kd_peminjaman
- °kd_buku

Pengembalian

- *kd_pengembalian
- °tot_bayar_denda
- °tgl_kembali

Peminjaman

Kd_peminjaman	Tanggal_pinjam

Buku

Kd_buku	Nama_buku

Detail_peminjaman

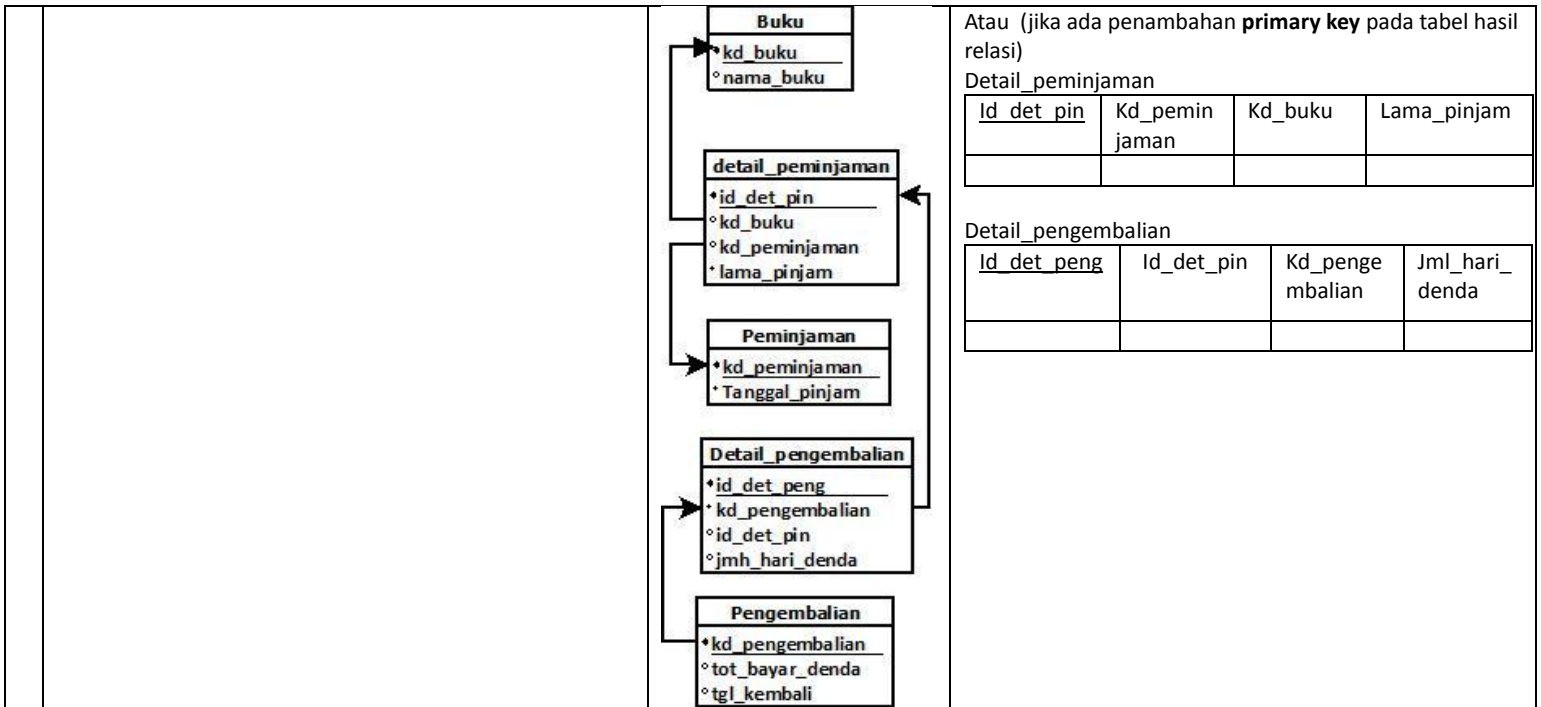
Kd_peminjaman	Kd_buku	Lama_pinjam

Pengembalian

Kd_pengembalian	Tot_bayar_denda	Tgl_kembali

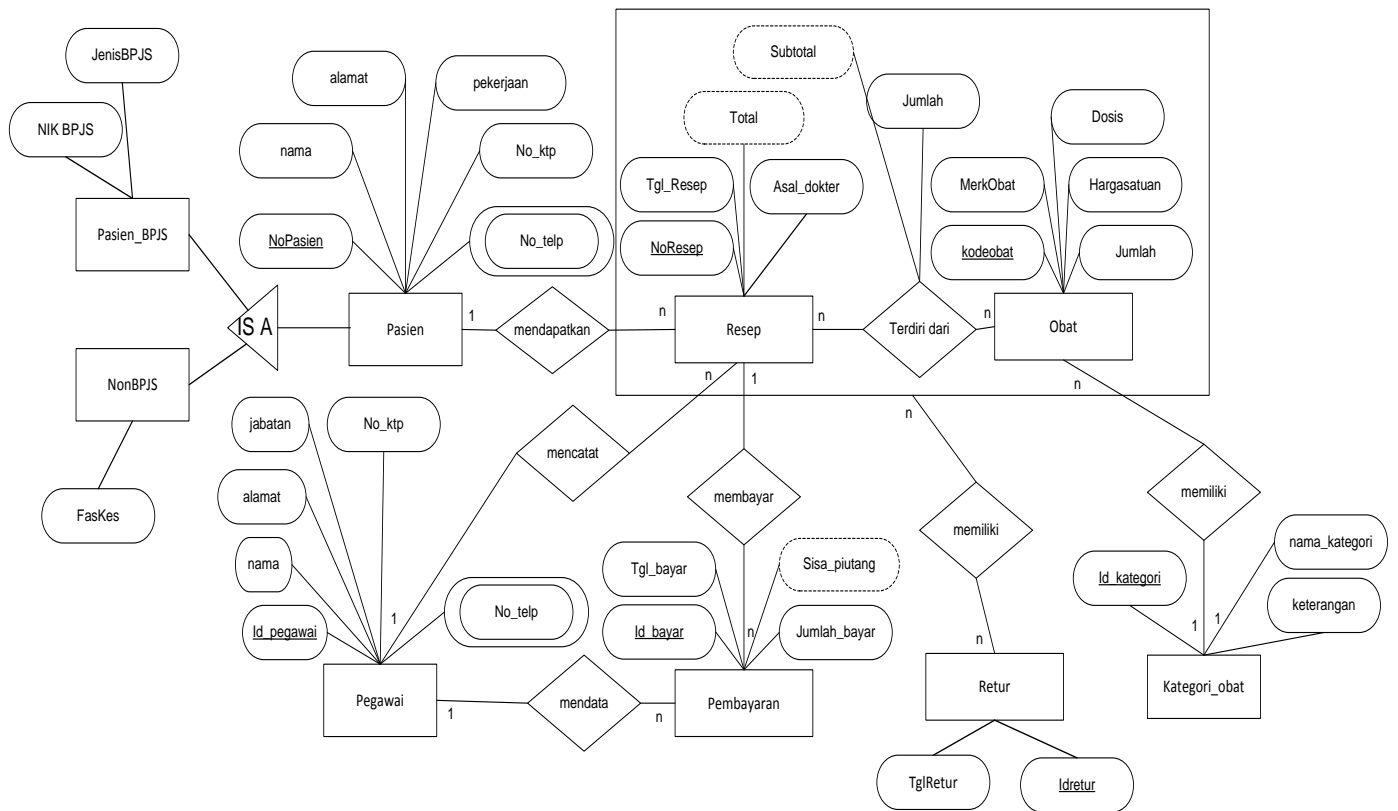
Detail_pengembalian

Kd_pengembalian	Kd_peminjaman	Kd_buku	Jml_hari_denda

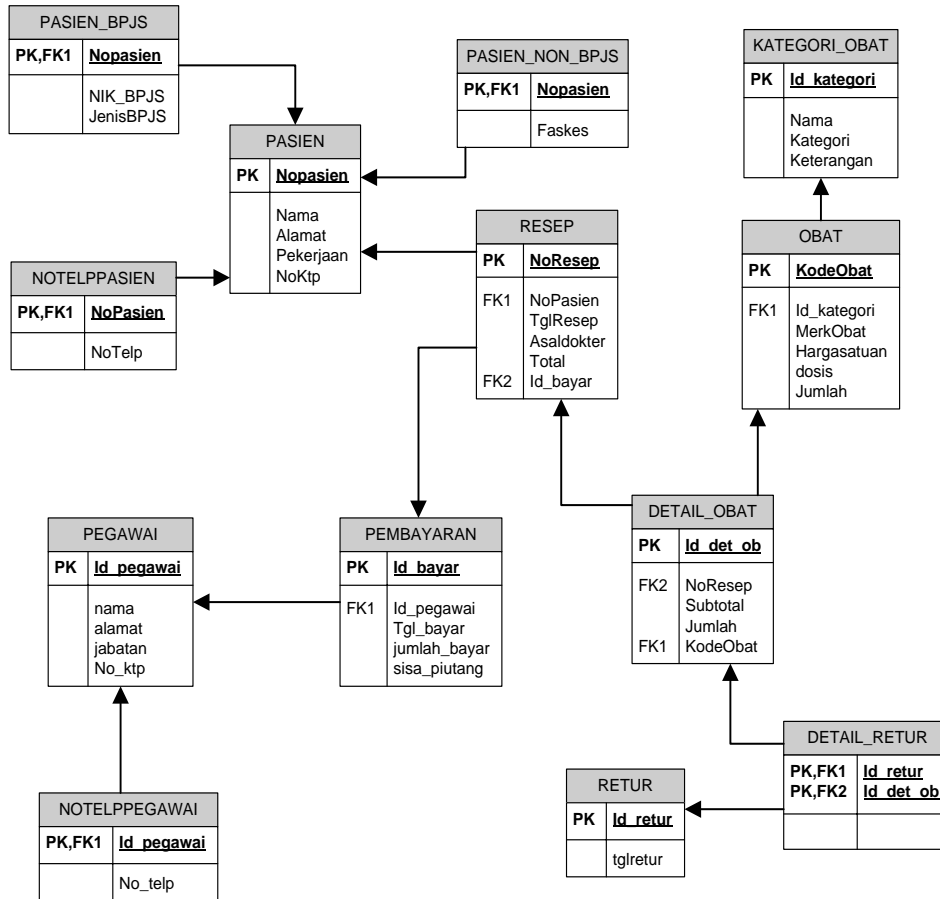


1.4 STUDI KASUS SKEMA PEMBAYARAN APOTIK

Berikut ini contoh Entity Relationship diagram studi kasus skema apotik



Dengan menggunakan aturan konversi ER diagram menjadi diagram relationshipnya seperti pada bagian 1.3 , maka berikut ini adalah hasil dari diagram relationship skema apotik. Diagram relationship = Relasi antar tabel



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**. Di tabel pegawai kolom **id_pegawai** adalah **primary key** nya

TABEL HASIL RELASI

Tabel diperoleh dari hasil relasi antar tabel diatas. Jika dilihat setiap kotak akan menjadi tabel, total 13 Tabel.

Tabel PASIEN

(#Nopasien,nama,alamat,pekerjaan,no_ktp)

Tabel NOTELPPASIEAN

(#Nopasien,Notelp)

Tabel PASIEN_BPJS

(#Nopasien,NIK_BPJS,JenisBPJS)

Tabel PASIEN_NON_BPJS

(#Nopasien,Faskes)

Tabel RESEP

(#NoResep,@Nopasien,tglresep,asaldokter,total)

Tabel OBAT

(#KodeObat,@id_kategori,merkObat,hargasatuan,dosis,jumlah)

Tabel DETAIL_OBAT

(#id_det_ob,@NoResep,@kodeobat,subtotal,jumlah)

Tabel PEGAWAI

(#Id_pegawai,nama,alamat,jabatan,no_ktp)

Tabel NOTELPPEGAWAI

(#Id_pegawai,no_telp)

Tabel PEMBAYARAN

(#Id_bayar,@id_pegawai,tgl_bayar,jumlah_bayar, sisa_piutang)

Tabel RETUR

(#Id_retur,tglretur)

Tabel DETAIL_RETUR

(@id_retur,@id_det_obat)

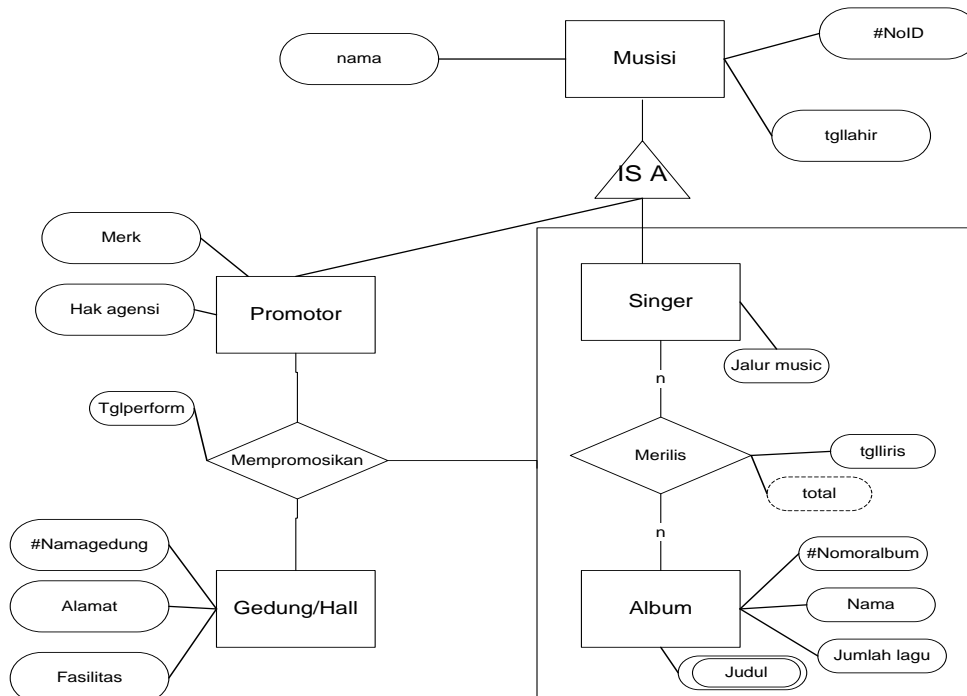
Tabel KATEGORI_OBAT

(#id_kategori,nama_kategori,keterangan)

1.5 TEST AKHIR

Perhatikan ER Digram dibawah ini, jawablah pada kertas yang telah disediakan isi dengan nama dan nim

1. Konversikan menjadi tabel tabel hasil relasi
2. Berapa jumlah tabel yang dihasilkan ?



2 BAB 2 – PENGANTAR BASIS DATA-DDL

2.1 IDENTITAS

Kompetensi

1. Praktikan mengetahui & memahami memiliki pengetahuan dasar basis data
2. Praktikan mengetahui berbagai aplikasi pendukung basis data meliputi editor dan server My SQL.
3. Praktikan dapat menjalankan aplikasi server MySQL, mengakses MySQL, Client MySQL, dan memahami tipe data pada mysql.
4. Praktikan dapat mengcreate database.

Topik

1. MySQL
2. Instalasi Aplikasi
3. Aplikasi Server MySQL
4. Mengakses MySQL
5. Client MySQL
6. Tipe Data MySQL
7. Database
8. DDL Awal

2.2 TEST AWAL

1. Apa yang ada ketahui tentang Basisdata atau database ?
2. Apakah DBMS (Data Base Management System) itu?
3. Berikan contoh 3 DBMS yang anda ketahui !

2.3 DBMS MY SQL

MySQL (*My Structure Query Language*) merupakan salah satu DBMS dari sekian banyak DBMS lain seperti Oracle, MS SQL, PostgreSQL dan banyak lagi. Semuanya mempunyai fungsi dan manfaat yang hampir sama namun dalam kelebihan dan kekurangan masing masing.

MySQL menggunakan bahasa SQL dan dapat dikatakan sebagai DBMS. DBMS (*Database Manajemen System*) merupakan salah satu sistem dalam mengakses database dengan menggunakan bahasa SQL.

MySQL juga merupakan aplikasi *Open source* artinya memungkinkan untuk

semua orang untuk menggunakan dan memodifikasi aplikasi tersebut. Sehingga siapapun bisa mendapatkan aplikasi MySQL secara gratis dan bebas digunakan.

Alasan menggunakan MySQL ?

- Database MySQL mempunyai performance sangat cepat, dapat dipercaya
- Reliable,
- MySQL telah banyak menangani pembuatan software besar,
- Bersifat *open source*,
- Mudah digunakan,
- Dapat dijalankan diberbagai sistem operasi (*multiplatform*) → Linux, Windows, Mac OS,
- Server *multithread* (dapat menangani beberapa permintaan (*request*) secara bersamaan),
- Metode enkripsi-nya bagus,
- Menggunakan autentikasi *user & password*,

2.4 INSTALASI MY SQL

Untuk instalasi MySQL lebih mudah bila anda sekalian melakukan instalasi my sql dan php my admin sekaligus dalam paket xampp . xampp bisa di download di

<https://www.apachefriends.org/index.html>. Sejak XAMPP 5.5.30 dan 5.6.14, XAMPP menggunakan MariaDB bukan MySQL. Perintah dan tool nya sama untuk keduanya.

2.5 APLIKASI SERVER MY SQL

Di dalam MySQL terdapat sebuah *database* yang ada sejak awal setelah anda menginstal MySQL, nama *database* tersebut adalah **mysql**. Pada *database* tersebut tersimpan nama-nama pengguna yang dapat mengakses MySQL secara lengkap dengan opsi otoritas yang dapat dilakukan oleh pengguna tersebut. Secara default *user* dengan nama **root** adalah pengguna yang menguasai secara utuh dan dapat membuat *user* lainnya (termasuk membatasi *user-user* lain). Untuk dapat menggunakan MySQL anda harus memasukkan *user* dan *password* yang sama dengan apa yang dideklarasikan dalam *database* (MySQL).

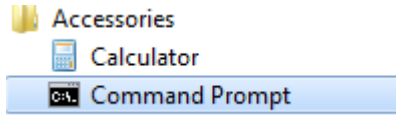
Dalam penerapan sistem sebenarnya, MySQL Server dan MySQL Client biasanya dijalankan pada komputer yang berbeda. Komputer Server berada pada sebuah ruangan tersendiri dan terhubung melalui jaringan dengan beberapa komputer Client. Namun kali ini kita akan menjalankan keduanya di dalam sebuah komputer saja. Untuk MySQL Server, kita telah menjalankannya baik sebagai service dari control **panel xampp**, atau **manual dengan mysqld.exe** .

Dengan MySQL server yang telah berjalan, kita akan mengaksesnya menggunakan MySQL Client dari Command Prompt Windows (selanjutnya akan kita singkat dengan cmd), menggunakan aplikasi mysql.exe dari folder bin MySQL.

2.6 MENGAKSES MY SQL

Untuk mengakses MySQL dengan mode *text* dari *Console* (*Command Prompt* = Microsoft Windows), dapat dilakukan dengan cara :

1. Buka Command Promp dan berikan perintah (jika MySQL terinstall di direktori C:\) :



2. Login. Berikan perintah (jika MySQL terinstall di direktori C:\)

```
cd c:\xampp\mysql\bin
```

login, ketikkan

```
mysql -u root -h localhost root -p
```

lalu enter, akan muncul perintah enter password: , biarkan kosong lalu tekan enter

username : root

password : (maksudnya password kosong)

Keterangan :

- **-u = root**
Opsi ini menunjukkan nama User yang digunakan
- **-h = localhost**
Opsi ini menunjukkan nama Host/IP (localhost/127.0.0.1)
- **-p =**
 - Opsi ini menunjukkan *password* yang digunakan (jika opsi ini digunakan), jika tidak menggunakan *password* maka opsi ini tidak perlu dicantumkan.
 - Opsi ini juga dapat digunakan untuk mendeklarasikan nama database yang digunakan.

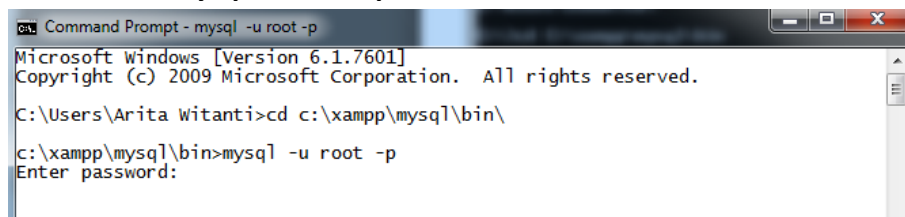
Jika menggunakan XAMPP di **Linux** dapat diakses dengan :

```
dnd@riyaniezt:~$ /opt/lampp/bin/mysql -u root
```

Jika menggunakan XAMPP di **Microsoft Windows** dapat diakses dengan :

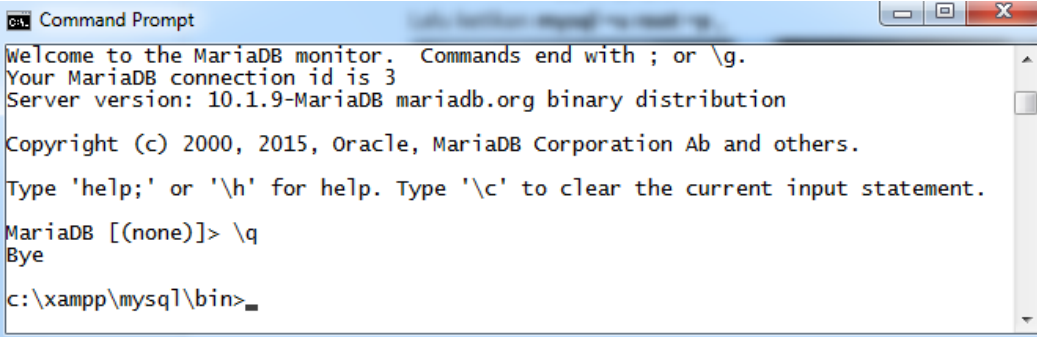
```
C:\xampp\mysql\bin\
```

Lalu ketikkan **mysql -u root -p** ,



3. Keluar dari mysql

```
mysql> \q  
bye
```



```
Command Prompt  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 3  
Server version: 10.1.9-MariaDB mariadb.org binary distribution  
Copyright (c) 2000, 2015, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
MariaDB [(none)]> \q  
Bye  
c:\xampp\mysql\bin>
```

File-file database, tabel dalam MySQL dengan menggunakan XAMPP disimpan di directory :

OS	Database MySQL	PHP
Linux	/opt/lampp/var/mysql/	/opt/lampp/htdocs
Windows	C:\xampp\mysql\data	c:\xampp\htdocs

2.7 CLIENT MY SQL

mysql.exe merupakan aplikasi yang digunakan untuk melakukan interaksi dengan server MySQL atau sering juga disebut sebagai *client* MySQL. *Client* MySQL biasanya diletakkan di *directory* :

Linux :

Menggunakan LAMPP (XAMPP versi LINUX).

dnd@riyaniezt:~\$ /opt/lampp/bin/mysql

Microsoft Windows :

Menggunakan XAMPP.

C:\xampp\mysql\bin\mysql.exe atau mysql

Perintah-perintah yang dituliskan dengan *command line* pada *prompt* MySQL harus diakhiri dengan titik koma(;).

```

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Arita Witanti>cd c:\xampp\mysql\bin

c:\xampp\mysql\bin>mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 5
Server version: 10.1.9-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2015, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> _

```

2.8 TYPE DATA MY SQL

Beberapa tipe data yang disediakan oleh MySQL antara lain :

Type 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	

2.9 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.

2.10 DATA DEFINITION LANGUAGE (DDL)

Berikut ini adalah beberapa perintah DDL untuk membuat database

2.10.1 Membuat Database

Untuk membuat *database* dalam server MySQL,

- Perintah :

```
create database nama_database;
```

Contoh :

```
mysql > create database praktikum;
```

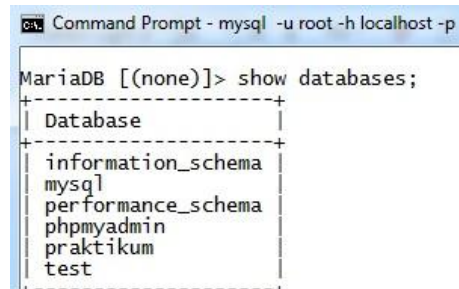


```
MariaDB [(none)]> create database praktikum;
Query OK, 1 row affected (0.03 sec)
MariaDB [(none)]>
```

2.10.2 Melihat seluruh Database

Untuk melihat seluruh *database* yang telah dibuat,

- Perintah :
show databases;
- Contoh :

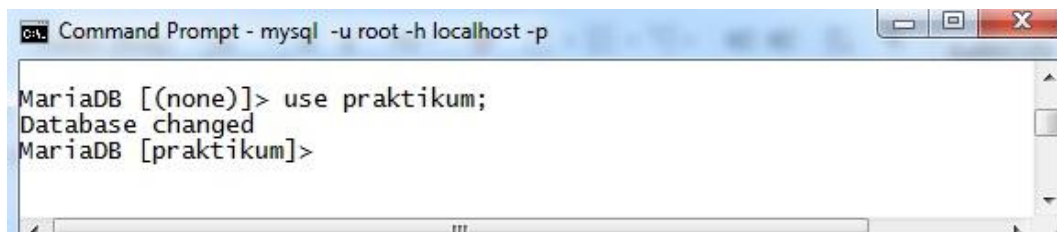


```
Command Prompt - mysql -u root -h localhost -p
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| praktikum |
| test |
+-----+
```

2.10.3 Mengakses Database

Untuk mengakses *database* yang telah dibuat. Perintah ini diperlukan saat kita akan menggunakan database tersebut.

- Perintah :
use nama_database;
- Contoh :
mysql > **use praktikum;**



```
Command Prompt - mysql -u root -h localhost -p
MariaDB [(none)]> use praktikum;
Database changed
MariaDB [praktikum]>
```

2.10.4 Menghapus Database

Untuk menghapus *database* yang telah dibuat,

- Perintah :

drop database nama_database;

- Contoh :

mysql > **drop database** praktikum;

```
MariaDB [praktikum]> drop database praktikum;
Query OK, 0 rows affected (0.19 sec)

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| test |
+-----+
5 rows in set (0.00 sec)
```

2.11 TEST AKHIR

1. Apakah instalasi My SQL anda berhasil ? cobalah login ke mysql pada komputer anda !
2. Buatlah sebuah database dengan nama “Prak_NIM”
3. Operasikan perintah SQL untuk :
 - a. Membuat database
 - b. Melihat seluruh database pada mysql server
 - c. Mengakses database/ menggunakan database
 - d. Menghapus database
4. Pilih tema dalam perancangan database (ditentukan oleh asisten),
5. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab1-NIM.odt
 - b. Isi file laporan :
 - i. Source SQL
 - ii. Screenshot CMD
 - iii. Tema yang ditentukan
 - c. Simpan di directory “PrakDB-NIM” yang telah dibuat tadi

Jawaban

3 BAB 3 – DATA DEFINITION LANGUAGE (DDL)

3.1 IDENTITAS

Kompetensi

1. Praktikan memahami SQL dan perintah DDL pada SQL.
2. Praktikan dapat membuat table dengan benar beserta relationshipnya
3. Praktikan memahami type table InnoDB.

Topik

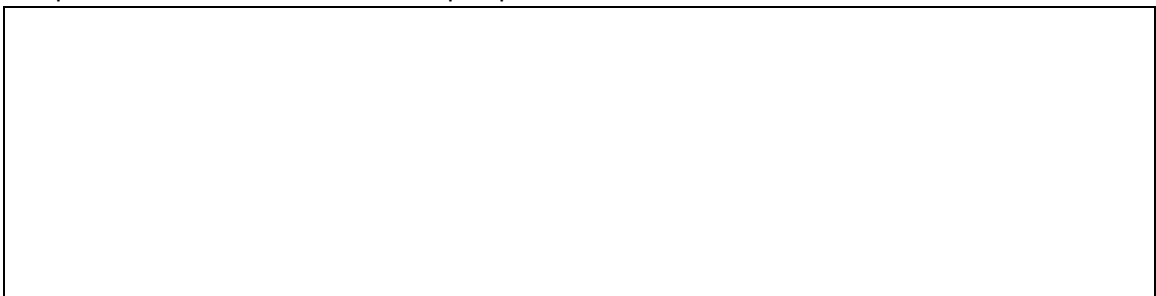
1. SQL
2. Membuat Table
3. Constraint Relasi
4. Nilai otomatis dan nilai default

3.2 TEST AWAL

1. Operasikan perintah SQL untuk :
 - a. Membuat database
 - b. Melihat seluruh database pada mysql server
 - c. Mengakses database/ menggunakan database
 - d. Menghapus database



2. Tampilkan hasil screen shoot dan simpan pada folder PrakDB-NIM



3.3 SQL

Secara umum perintah-perintah yang terdapat di dalam SQL, diklasifikasikan menjadi tiga bagian, antara lain yaitu :

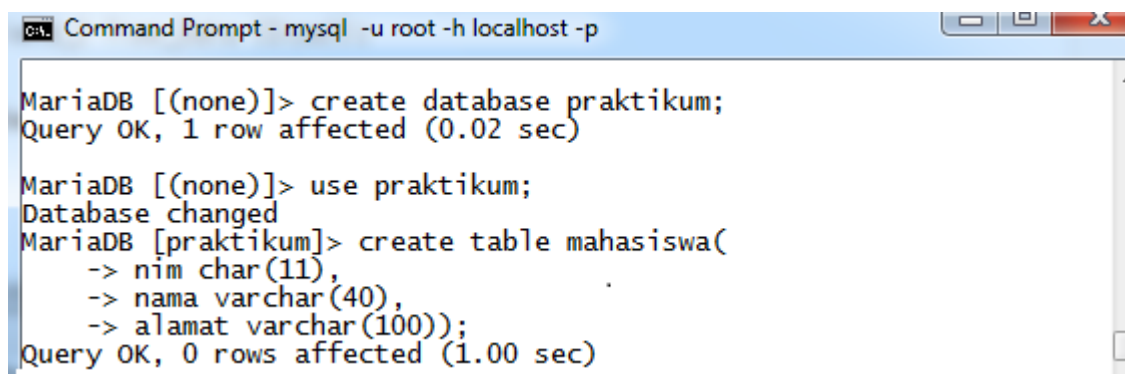
1. DDL (*Data Definition Language*)
 - Merupakan perintah SQL yang berkaitan dengan pendefinisian suatu struktur database, dalam hal ini database dan table.
 - Perintah DDL adalah: CREATE, ALTER, RENAME, DROP.
2. DML (*Data Manipulation Language*)
 - Merupakan perintah SQL yang berkaitan dengan manipulasi atau pengolahan data atau record dalam table.
 - Perintah DML antara lain: SELECT, INSERT, UPDATE, DELETE.
3. DCL (*Data Control Language*)
 - Merupakan perintah SQL yang berkaitan dengan manipulasi user dan hak akses (priviledges).
 - Perintah SQL yang termasuk dalam DCL antara lain: GRANT, REVOKE.

3.4 MEMBUAT TABLE

Setelah menciptakan suatu database dan mengaktifkan database tersebut maka dapat dilakukan perintah pembuatan tabel.

3.4.1 Create Table

- Perintahnya :
Create Table Nama_Table (Nama_Field_1 Tipe_Data (Size),
Nama_Field_2 Tipe_Data (Size));
- Contoh :
mysql > **Create Table** Mahasiswa (NIM char(11),
Nama varchar(40),
Alamat varchar(100));



```
Command Prompt - mysql -u root -h localhost -p
MariaDB [(none)]> create database praktikum;
Query OK, 1 row affected (0.02 sec)

MariaDB [(none)]> use praktikum;
Database changed
MariaDB [praktikum]> create table mahasiswa(
-> nim char(11),
-> nama varchar(40),
-> alamat varchar(100));
Query OK, 0 rows affected (1.00 sec)
```

2.4.2 Melihat Table dan Struktur Table

Untuk melihat seluruh table yang telah dibuat sebelumnya, (Dengan syarat : sudah berada di database yang mempunyai table tersebut).

- perintahnya :
mysql > **Show Tables;**

```
MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| mahasiswa           |
+-----+
1 row in set (0.01 sec)
```

Sedangkan untuk melihat struktur dari masing-masing tabel,

- perintahnya :
Desc/Describe Nama_Table ;
- Contoh :
mysql > **Desc Mahasiswa;**

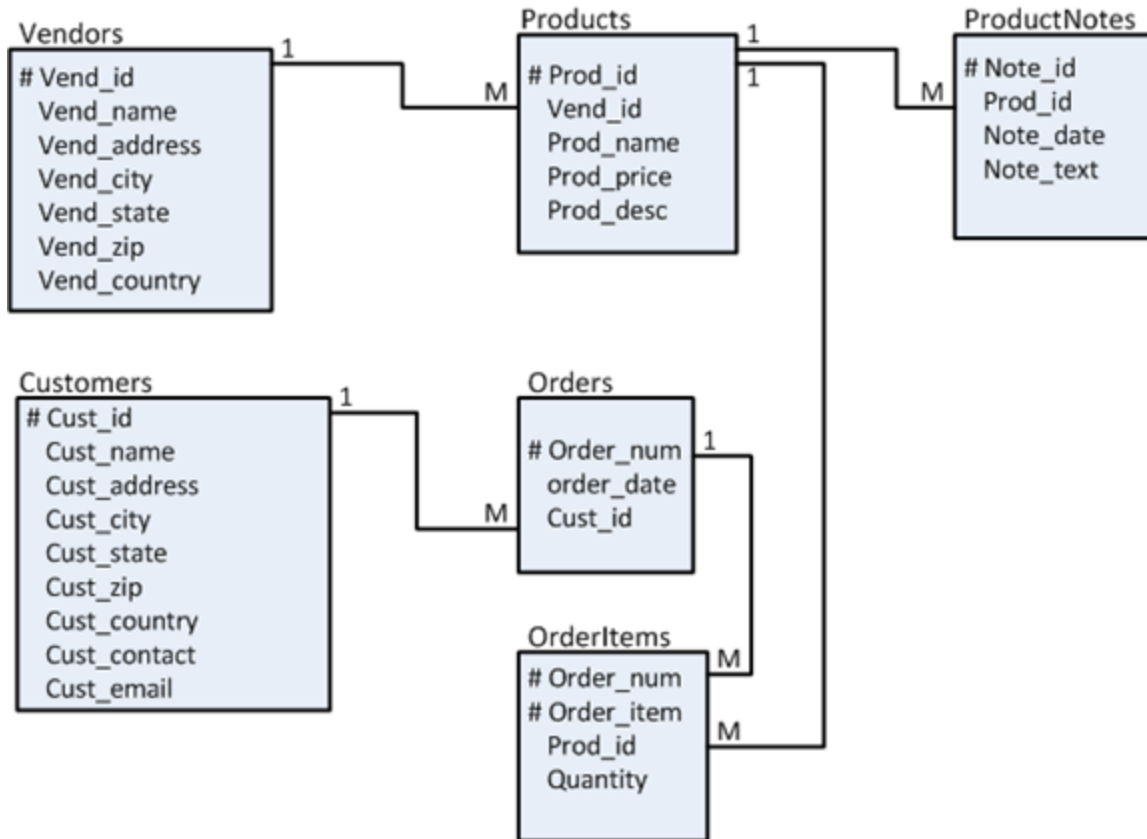
```
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |       |
| nama  | varchar(40)   | YES  |     | NULL    |       |
| alamat | varchar(100) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.03 sec)
```

3.5 CONSTRAINT TABLE

Constraint adalah aturan atau batasan yang sengaja kita terapkan pada table untuk menjaga integritas dan konsistensi data. Ada 5 aturan constraint yang biasanya diterapkan pada table. Constraint ini biasanya diterapkan saat melakukan **create table** atau bisa juga saat **alter table** (dibahas di bab 4).

Berikut ini 5 aturan constraint pada mysql yaitu *primary key, foreign key, unique, not null dan check*.

Untuk lebih memahami penggunaan constraint tersebut kita coba terapkan pada skema order entry. Berikut ini adalah diagram relationship atau relasi antar table dari skema order entry.



Gambar Skema Order Entry

Untuk satu skema kita buat satu database. Untuk menerapkan skema tersebut dalam sebuah database berikut langkah – langkah yang kita lakukan :

1. Create database *order entry*
2. Buat table skema *order entry* dengan memperhatikan urutan, mulailah dari table kuat yaitu table hasil dari entitas tunggal dan tanpa foreign key kemudian dilanjutkan table hasil relasi. Dalam kasus *order entry* urutannya sbb :
 - Vendors, customers
 - Products, orders
 - ProductNotes, orderItems

3.5.1 Penerapan constraint pada Skema Order Entry

1. Terlebih dahulu kita buat database nya dan use databasenya

```

MariaDB [praktikum]> create database OrderEntry;
Query OK, 1 row affected (0.01 sec)

MariaDB [praktikum]> use orderEntry;
Database changed
MariaDB [orderEntry]>
    
```

2. Buat table vendor dan table customer

```

MariaDB [orderEntry]> CREATE TABLE vendors(
-> vend_id CHAR(4) NOT NULL PRIMARY KEY ,
-> vend_name VARCHAR(25) NOT NULL,
-> vend_address VARCHAR(30),
-> vend_city VARCHAR(20),
-> vend_state VARCHAR(5),
-> vend_zip VARCHAR(7),
-> vend_country VARCHAR(15));

```

Query OK, 0 rows affected (1.14 sec)

```

MariaDB [orderEntry]> CREATE TABLE customers(
-> cust_id CHAR(5) NOT NULL PRIMARY KEY,
-> cust_name VARCHAR(25) NOT NULL,
-> cust_address VARCHAR(30) NULL,
-> cust_city VARCHAR(25) NULL,
-> cust_state VARCHAR(5) NULL,
-> cust_zip VARCHAR(5) NULL,
-> cust_country VARCHAR(20) NULL,
-> cust_contact VARCHAR(25) NULL,
-> cust_email VARCHAR(30) NULL);

```

Query OK, 0 rows affected (0.49 sec)

```

MariaDB [orderEntry]> show tables;

```

```

+-----+
| Tables_in_orderentry |
+-----+
| customers             |
| vendors               |
+-----+
2 rows in set (0.27 sec)

```

3. Buat table products dan orders

```

MariaDB [orderEntry]> CREATE TABLE products(
-> prod_id VARCHAR(10) NOT NULL PRIMARY KEY,
-> vend_id CHAR(4) NOT NULL ,
-> prod_name VARCHAR(25) NOT NULL ,
-> prod_price INT NOT NULL ,
-> prod_desc VARCHAR(255) NULL);

```

Query OK, 0 rows affected (0.90 sec)

```

MariaDB [orderEntry]> CREATE TABLE orders(
-> order_num INT NOT NULL ,
-> order_date DATE NOT NULL,
-> cust_id CHAR(5) NOT NULL,
-> PRIMARY KEY(order_num));

```

Query OK, 0 rows affected (0.33 sec)

4. Buat table productnotes dan table orderItems

```

MariaDB [orderEntry]> CREATE TABLE productnotes(
-> note_id CHAR(3) NOT NULL,
-> prod_id VARCHAR(10) NOT NULL,
-> note_date DATE NOT NULL,
-> note_text VARCHAR(200) NULL,
-> PRIMARY KEY (note_id),
-> FOREIGN KEY (prod_id) REFERENCES products (prod_id));

```

Query OK, 0 rows affected (0.36 sec)

Perhatikan table orderItem memiliki containt primary key dua kolom sekaligus karena menerapkan surrogate/kunci pengganti .

```

MariaDB [orderentry]> CREATE TABLE orderitems(
-> order_num INT NOT NULL ,
-> order_item INT NOT NULL ,
-> prod_id VARCHAR(10) NOT NULL ,
-> quantity INT NOT NULL,
-> PRIMARY KEY (order_num, order_item));
Query OK, 0 rows affected (0.45 sec)

MariaDB [orderentry]> Show tables;
+-----+
| Tables_in_orderentry |
+-----+
| customers            |
| orderitems          |
| orders              |
| productnotes        |
| products            |
| vendors             |
+-----+
6 rows in set (0.00 sec)

```

3.6 NILAI OTOMATIS DAN NILAI DEFAULT

3.6.1 Nilai otomatis / Auto Increment

Suatu nilai otomatis merupakan suatu field yang diisi secara otomatis oleh sistem. Biasanya paling banyak digunakan pada *primary key*. Tipe data kolom yang akan diset nilai autoincrement adalah **int**.

Perintahnya :

Auto_Increment

Contoh :

```

mysql> Create Table Mahasiswa2 (
          ID int(5) not null primary key auto_increment,
          NIM char(8) not null,
          Nama_Mhs varchar(50),
          Jurusan varchar(200),
          Fakultas varchar(30));

```

```

MariaDB [praktikum]> Create Table Mahasiswa2 (
-> ID int(5) not null primary key auto_increment,
-> NIM char(8) not null,
-> Nama_Mhs varchar(50),
-> Jurusan varchar(200),
-> Fakultas varchar(30));
Query OK, 0 rows affected (1.00 sec)

MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| mahasiswa           |
| mahasiswa2          |
+-----+
2 rows in set (0.03 sec)

MariaDB [praktikum]> desc mahasiswa2;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID    | int(5)       | NO   | PRI | NULL    | auto_increment |
| NIM   | char(8)      | NO   |     | NULL    |                |
| Nama_Mhs | varchar(50) | YES  |     | NULL    |                |
| Jurusan | varchar(200) | YES  |     | NULL    |                |
| Fakultas | varchar(30) | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.03 sec)

```


3.6.2 Nilai default

Suatu nilai default merupakan pemberian nilai secara otomatis oleh system terhadap suatu field tertentu dengan nilai NULL.

Perintahnya :

```
Default Nilai_Default
```

Contoh :

```
mysql> Create Table Mtkul (  
        Kode_Mtkul int(5) not null primary key,  
        Nama_Mtkul Varchar (30),  
        Sks int(1) default 0,  
        Semester int(1) default 0);
```

```
MariaDB [praktikum]> Create Table Mtkul (  
-> Kode_Mtkul int(5) not null primary key,  
-> Nama_Mtkul Varchar(30),  
-> Sks int(1) default 0,  
-> Semester int(1) default 0);  
Query OK, 0 rows affected (0.23 sec)
```

```
MariaDB [praktikum]> desc mtkul;
```

Field	Type	Null	Key	Default	Extra
Kode_Mtkul	int(5)	NO	PRI	NULL	
Nama_Mtkul	varchar(30)	YES		NULL	
Sks	int(1)	YES		0	
Semester	int(1)	YES		0	

4 rows in set (0.03 sec)

3.7 TYPE : INNODB DAN XTRADB

InnoDB Merupakan storage engine yang sering dipakai di website. MySQL memberikan pilihan beberapa *table engine* untuk setiap tabel yang ada. Sebelum versi 5.5, table yang dibuat dengan CREATE TABLE tanpa menyertakan *table engine* yang akan dipakai secara otomatis akan menggunakan engine MyISAM. Pada versi 5.5, *default table engine* diganti dari MyISAM menjadi InnoDB Storage engine ini sering dikenal karena mempunyai fitur transaksi, seperti commit, rollback dan crash recovery layaknya oracle. Disamping itu juga mempunyai fitur tabel relasi dan integritas – foreignkey. Kekurangan InnoDB adalah membutuhkan resource memori yang besar.

Pada mariaDB InnoDB digantikan dengan XtraDB yang lebih stabil.

Apabila pada satu kondisi anda memerlukan type table dengan type engine InnoDB maka dapat menambahkan sintak berikut pada akhir create table.

```
Engine = InnoDB;
```

Contoh :

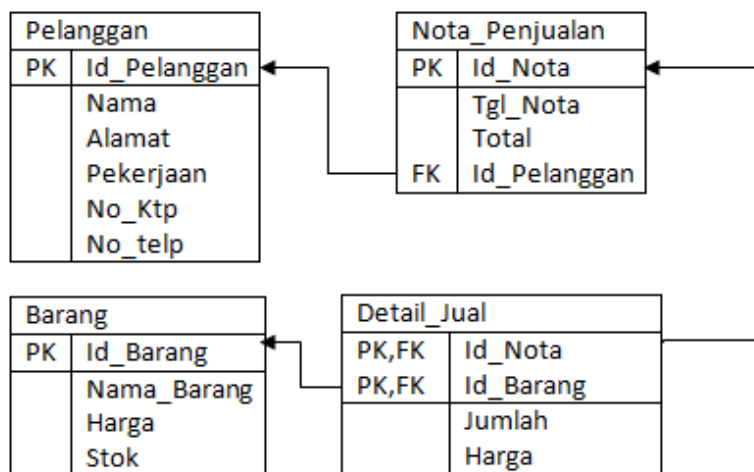
```

MariaDB [praktikum]> create table compress(
  -> x int primary key not null,
  -> y varchar(50) null) engine=InnoDB;
Query OK, 0 rows affected (0.33 sec)

```

3.8 TEST AKHIR

1. Buatlah sebuah database baru dengan nama Penjualan_Barang
2. Gunakan database penjualan_barang kemudian buatlah tabel tabcel hasil dari diagram relationship penjualan barang.
3. Buatlah laporan praktikum dengan ketentuan sbb :
 - a. Nama file laporan : PrakDB_Bab3_NIM.odt
 - b. Isi file laporan :
 - i. Souce sql
 - ii. Screenshot CMD
 - c. Simpan di direktory "PrakDB-NIM"



ERD Penjualan Barang

4 BAB 4 – ALTER, MODIFY , DROP , RENAME (DDL)

4.1 IDENTITAS

Kompetensi

1. Praktikan memahami penggunaan DDL antara lain alter , modify , drop dan rename

Topik

1. Alter , Modify
2. Drop, Rename

4.2 TEST AWAL

Jawablah pertanyaan berikut dengan tulisan tangan anda sendiri pada selembar kertas, cantumkan nama dan nim serta kelas.

Berdasarkan skema order entry pada bab 3 jawablah pertanyaan berikut ini.

1. Berikan 1 contoh perintah alter dan modify dalam satu sintaks
2. Berikan 1 contoh penggunaan drop table.
3. Berikan 1 contoh penerapan rename table

4.3 ALTER TABLE

4.3.1 Merubah Struktur Table

Ada empat macam perubahan yang dapat dilakukan terhadap struktur tabel, yaitu :

1. Perubahan terhadap nama field/kolom
 2. Perubahan terhadap tipe data
 3. Penambahan field
 4. Penghapusan field
1. **Merubah Nama Field**
Perubahan yang terjadi hanya pada nama field/kolom saja. Nama field/kolom lama diganti dengan nama field/kolom yang baru. Untuk merubah nama field tersebut dapat digunakan perintah **Change**.
 - Perintahnya :
`Alter Table Nama_Table Change Nama_Field_Lama
Nama_Field_Baru Tipe_Data (Size);`
 - Contoh :
`mysql > Alter Table Mahasiswa
Change Nama Nama_Mhs Char (40);`

```

Command Prompt - mysql -u root -p
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |       |
| nama  | varchar(40)   | YES  |     | NULL    |       |
| alamat| varchar(100)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.06 sec)

MariaDB [praktikum]> alter table mahasiswa change nama Nama_Mhs char(40);
Query OK, 0 rows affected (0.80 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim        | char(11)      | YES  |     | NULL    |       |
| Nama_Mhs   | char(40)      | YES  |     | NULL    |       |
| alamat     | varchar(100)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.02 sec)

```

2. Merubah Tipe Data

Perubahan yang terjadi hanya pada tipe data yang digunakan oleh field/kolom tertentu. Tipe data baru langsung disebutkan di belakang nama field/kolom, tanpa harus menyebutkan tipe data lama. Untuk merubah tipe data tersebut digunakan perintah **Modify**.

- Perintahnya :

Alter Table Nama_Table

Modify Nama_Field Tipe_Data_Baru (Size);

- Contoh :

mysql > **Alter Table** Mahasiswa **Modify** NIM Char (8);

```

Command Prompt - mysql -u root -p
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | int(11)       | YES  |     | NULL    |       |
| Nama_Mhs | char(40)     | YES  |     | NULL    |       |
| alamat| varchar(100)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.03 sec)

MariaDB [praktikum]> alter table mahasiswa modify nim char(11);
Query OK, 0 rows affected (1.01 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim        | char(11)      | YES  |     | NULL    |       |
| Nama_Mhs   | char(40)      | YES  |     | NULL    |       |
| alamat     | varchar(100)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.02 sec)

```

3. Menambah Field

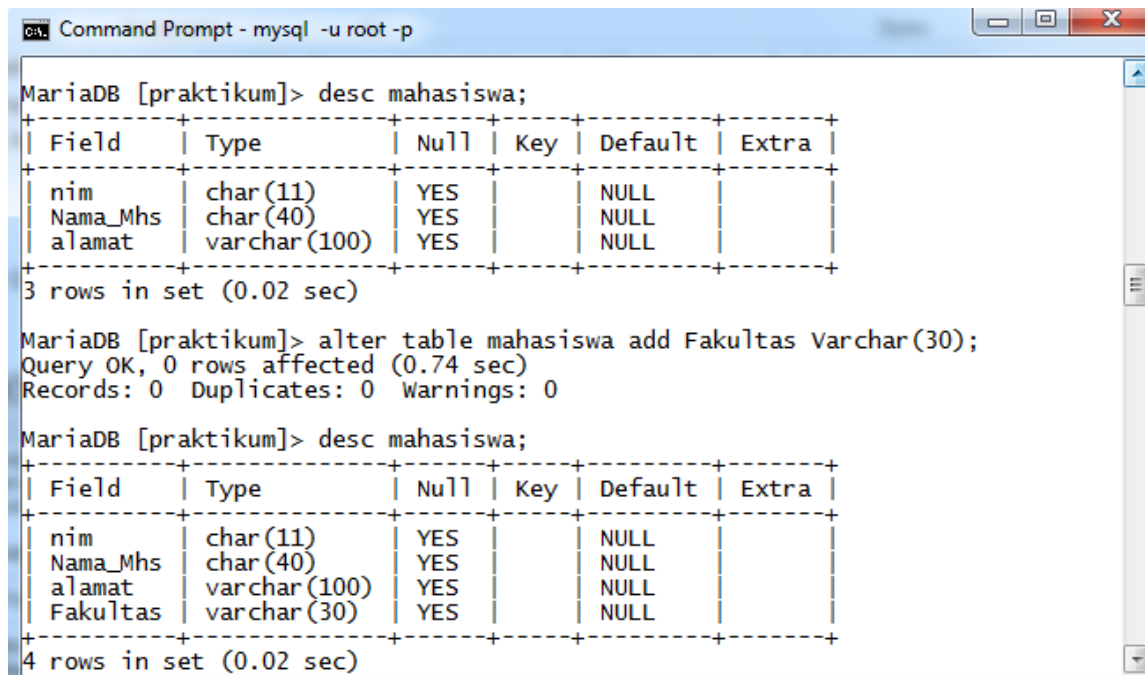
Struktur tabel akan berubah dengan bertambahnya field/kolom baru di dalamnya. Untuk menambahkan field baru dapat digunakan perintah **Add**.

- Perintahnya :

```
Alter Table Nama_Table  
Add Nama_Field_Baru Tipe_Data (Size);
```

- Contoh :

```
mysql > Alter Table Mahasiswa Add Fakultas Varchar (30);
```



```
Command Prompt - mysql -u root -p  
MariaDB [praktikum]> desc mahasiswa;  
+-----+-----+-----+-----+-----+-----+  
| Field | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| nim   | char(11)      | YES  |     | NULL    |       |  
| Nama_Mhs | char(40)     | YES  |     | NULL    |       |  
| alamat | varchar(100) | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
3 rows in set (0.02 sec)  
  
MariaDB [praktikum]> alter table mahasiswa add Fakultas Varchar(30);  
Query OK, 0 rows affected (0.74 sec)  
Records: 0 Duplicates: 0 Warnings: 0  
  
MariaDB [praktikum]> desc mahasiswa;  
+-----+-----+-----+-----+-----+-----+  
| Field | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| nim   | char(11)      | YES  |     | NULL    |       |  
| Nama_Mhs | char(40)     | YES  |     | NULL    |       |  
| alamat | varchar(100) | YES  |     | NULL    |       |  
| Fakultas | varchar(30) | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.02 sec)
```

4. Menghapus Field

Struktur tabel dapat mengalami perubahan karena berkurangnya field/kolom tertentu. Untuk menghapus file dalam suatu table tersebut dapat dilakukan dengan perintah **Drop Column**.

- Perintahnya :

```
Alter Table Nama_Table Drop Column Nama_Field;
```

- Contoh :

```
mysql > Alter Table Mahasiswa Drop Column Fakultas;
```

```

C:\> Command Prompt - mysql -u root -p
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |       |
| Nama_Mhs | char(40)     | YES  |     | NULL    |       |
| alamat | varchar(100) | YES  |     | NULL    |       |
| Fakultas | varchar(30) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.02 sec)

MariaDB [praktikum]> alter table mahasiswa drop column fakultas;
Query OK, 0 rows affected (0.59 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |       |
| Nama_Mhs | char(40)     | YES  |     | NULL    |       |
| alamat | varchar(100) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.02 sec)

```

4.4 DROP

Menghapus Table

Jika table yang dibuat tadi sudah tidak dibutuhkan, table bisa dihapus. Sehingga ingin menghapusnya maka dapat digunakan perintah **Drop Table**.

- Perintahnya :

Drop Table Nama_Table;

- Contoh :

mysql > **Drop Table** Mhs;

```

MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| compress            |
| mahasiswa           |
| mahasiswa2          |
| mtkul               |
+-----+
4 rows in set (0.00 sec)

MariaDB [praktikum]> drop table mahasiswa2;
Query OK, 0 rows affected (0.47 sec)

MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| compress            |
| mahasiswa           |
| mtkul               |
+-----+
3 rows in set (0.00 sec)

```

4.5 TEST AKHIR

1. Di dalam database "Order Entry" yang telah anda buat pada praktikum sebelumnya. Untuk nim ganjil gunakan tabel customer,orderitems,order Operasikan salah satu tabel dengan perintah SQL berikut ini :
 - i. Perubahan terhadap nama field/kolom
 - ii. Perubahan terhadap tipe data
 - iii. Penambahan field
 - iv. Penghapusan field
2. Buatlah tabel baru dengan nama contact_customer

Nama kolom	Type data dan ukuran
Id_cust	Varchar 5
HP1	Varchar 20
HP 2	Varchar 20

Kemudian hapus table tersebut jika telah berhasil anda buat, jangan lupa menyimpan snapshot cmd nya.

3. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab4-NIM.odt
 - b. Isi file laporan :
 - i. Source SSQL
 - ii. Screenshot CMD
 - c. Simpan di directory "PrakDB-NIM".

5 BAB 5 – DATA MANIPULATION LANGUAGE (DML)

5.1 IDENTITAS

Kompetensi

1. Praktikan dapat mengoperasikan perintah dml
2. Praktikan dapat memilih perintah dml yang tepat untuk memanipulasi table

Topik

1. Insert table
2. Query sederhana
3. Update table
4. Delete data
5. Query dengan kondisi

5.2 TEST AWAL

Tuliskan pada kertas jawaban pertanyaan dibawah ini. Dengan tulisan tangan, jangan lupa NIM, Nama , kelas.

1. Bagaimana cara memasukan data ke table, beri contoh !
2. Bagaimana cara menghapus data dari table, beri contoh !

5.3 INSERT TABLE

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) ;
```

- Atau dengan perintah :

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

- Contoh penggunaan pada skema order entry


```
mysql> Insert Into Mahasiswa Values ('05052652', 'Paijo',
'Jalan Wates km 11');
```

```
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |      |
| Nama_Mhs | char(40)     | YES  |     | NULL    |      |
| alamat | varchar(100) | YES  |     | NULL    |      |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

MariaDB [praktikum]> insert into Mahasiswa values ('05052652','Painem','Jalan Wates Km 11');
Query OK, 1 row affected (0.76 sec)

MariaDB [praktikum]> select * from mahasiswa;
+-----+-----+-----+
| nim   | Nama_Mhs | alamat |
+-----+-----+-----+
| 05052652 | Painem   | Jalan Wates Km 11 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Contoh dengan sintaks lain

```
MariaDB [praktikum]> insert into mahasiswa(nim,nama_mhs,alamat) values ('07080905040','Paijo','Jln Jembatan Merah no 23');
Query OK, 1 row affected (0.73 sec)

MariaDB [praktikum]> select * from mahasiswa;
+-----+-----+-----+
| nim   | Nama_Mhs | alamat |
+-----+-----+-----+
| 05052652 | Painem   | Jalan Wates Km 11 |
| 07080905040 | Paijo    | Jln Jembatan Merah no 23 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

Untuk memperlancar entry data , isikan table customer sehingga datanya terisi seperti dibawah ini

```
MariaDB [orderentry]> select * from customers;
+-----+-----+-----+-----+-----+-----+-----+-----+
| cust_id | cust_name | cust_address | cust_city | cust_state | cust_zip | cust_country | cust_contact | cust_email |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 10002 | Mouse House | 333 Fromage Lane | Columbus | OH | 43333 | USA | Jerry Mouse | NULL |
| 10003 | Wascals | 1 Sunny Place | Muncie | IN | 42222 | USA | Jim Jones | rabbit@wascally.com |
| 10004 | Yosemite Place | 829 Riverside Drive | Phoenix | AZ | 88888 | USA | Y Sam | sam@yosemite.com |
| 10005 | E Fudd | 4545 53rd Street | Chicago | IL | 54545 | USA | E Fudd | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

5.4 QUERY SEDERHANA

Select merupakan perintah yang dapat digunakan untuk :

- Menampilkan data secara keseluruhan yang terdapat di dalam table.
- Menampilkan data tertentu yang terdapat di dalam table.
- Menampilkan dan mengurutkan data secara ascending dan descending

5.4.1 Menampilkan Data Secara Keseluruhan

Jika ingin menampilkan data data secara keseluruhan yang terdapat di dalam table, misalnya table Mhs_2 secara keseluruhan.

Perintahnya :

```
Select * From Nama_Table;
```

Contoh :

```
mysql> Select * From Mahasiswa;
```

```
MariaDB [praktikum]> select * from mahasiswa;
+-----+-----+-----+
| nim      | Nama_Mhs | alamat                |
+-----+-----+-----+
| 05052652 | Painem   | Jalan Wates Km 11    |
| 07080905040 | Paijo   | Jln Jembatan Merah no 23 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

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 menampilkan NIM dan Nama_Mhs.

Perintahnya :

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

Contoh :

```
mysql> Select NIM, Nama_Mhs From Mahasiswa;
```

```
MariaDB [praktikum]> select nim,nama_mhs from mahasiswa;
+-----+-----+
| nim      | nama_mhs |
+-----+-----+
| 05052652 | Painem   |
| 07080905040 | Paijo   |
+-----+-----+
2 rows in set (0.14 sec)
```

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;
```

Contoh :

```
mysql> Select NIM, Nama_Mhs From Mahasiswa Where Nim=05052652;
```

```
MariaDB [praktikum]> select * from mahasiswa where nim=05052652;
+-----+-----+-----+
| nim      | Nama_Mhs | alamat                |
+-----+-----+-----+
| 05052652 | Painem   | Jalan Wates Km 11    |
+-----+-----+-----+
1 row in set (3.23 sec)
```

5.5 UPDATE TABLE

Update merupakan perintah yang dapat digunakan untuk melakukan perubahan terhadap data yang 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' ;
```

- Contoh :

```
mysql> Update customers Set cust_address = 'Gejayan Yogya', cust_country=' INA'  
Where cust_id=' 10002' ;
```

```
MariaDB [orderentry]> update customers set cust_address='Gejayan Yogya',cust_country='INA' where cust_id='10002';  
Query OK, 1 row affected (0.42 sec)  
Rows matched: 1 Changed: 1 Warnings: 0
```

```
MariaDB [orderentry]> select * from customers;
```

cust_id	cust_name	cust_address	cust_city	cust_state	cust_zip	cust_country	cust_contact	cust_email
10002	Mouse House	Gejayan Yogya	Columbus	OH	43333	INA	Jerry Mouse	NULL
10003	Wascols	1 Sunny Place	Muncie	IN	42222	USA	Jim Jones	rabbit@wascallly.com
10004	Yosemite Place	829 Riverside Drive	Phoenix	AZ	88888	USA	Y Sam	sam@yosemite.com
10005	E Fudd	4545 53rd Street	Chicago	IL	54545	USA	E Fudd	NULL

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

5.6 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;
```

- Contoh :

```
mysql> Delete From Mhs_2 Where ID=' 5' ;
```

```

MariaDB [orderentry]> select * from customers;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| cust_id | cust_name | cust_address | cust_city | cust_state | cust_zip | cust_country | cust_contact | cust_email |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 10002 | Mouse House | Gejayan Yogya | Columbus | OH | 43333 | INA | Jerry Mouse | NULL | |
| 10003 | WascaIs | 1 Sunny Place | Muncie | IN | 42222 | USA | Jim Jones | rabbit@wascaIly.com |
| 10004 | Yosemite Place | 829 Riverside Drive | Phoenix | AZ | 88888 | USA | Y Sam | sam@yosemite.com |
| 10005 | E Fudd | 4545 53rd Street | Chicago | IL | 54545 | USA | E Fudd | NULL |
| 10006 | Ponidi | Jln Wates km 10 | Yogyakarta | | | | | | |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

MariaDB [orderentry]> delete from customers where cust_id='10006';
Query OK, 1 row affected (0.24 sec)

MariaDB [orderentry]> select * from customers;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| cust_id | cust_name | cust_address | cust_city | cust_state | cust_zip | cust_country | cust_contact | cust_email |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 10002 | Mouse House | Gejayan Yogya | Columbus | OH | 43333 | INA | Jerry Mouse | NULL |
| 10003 | WascaIs | 1 Sunny Place | Muncie | IN | 42222 | USA | Jim Jones | rabbit@wascaIly.com |
| 10004 | Yosemite Place | 829 Riverside Drive | Phoenix | AZ | 88888 | USA | Y Sam | sam@yosemite.com |
| 10005 | E Fudd | 4545 53rd Street | Chicago | IL | 54545 | USA | E Fudd | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

5.7 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;

```

contoh :

```

select * from customer where cust_id = '10003' or cust_name = ' wascaIs;

```

```
MariaDB [orderentry]> select * from customers where cust_id='10003' or cust_name='Wascols';
```

cust_id	cust_name	cust_address	cust_city	cust_state	cust_zip	cust_country	cust_contact	cust_email
10003	Wascols	1 Sunny Place	Muncie	IN	42222	USA	Jim Jones	rabbit@wascallly.com

1 row in set (0.01 sec)

artinya menampilkan data customer yang mempunyai id 10003 atau namanya wascols.

```
select prod_name from products where prod_price >=20 ;
```

artinya menampilkan data barang barang yang harganya lebih dari 20.

```
MariaDB [orderentry]> select prod_name from products where prod_price>=20;
```

prod_name
JetPack 1000
JetPack 2000
Safe

3 rows in set (0.06 sec)

Bisa dicoba untuk query dengan berbagai kondisi yang lain. Silahkan tanya asisten untuk lebih lanjut.

5.8 TEST AKHIR

- Berdasarkan Database order entry dan tabel yang telah dibuat sebelumnya, Isikan tabel berikut ini sesuai dengan isian sbb
 - Tabel customers

```
MariaDB [orderentry]> select * from customers;
```

cust_id	cust_name	cust_address	cust_city	cust_state	cust_zip	cust_country	cust_contact	cust_email
10002	Mouse House	333 Fromage Lane	Columbus	OH	43333	USA	Jerry Mouse	NULL
10003	Wascols	1 Sunny Place	Muncie	IN	42222	USA	Jim Jones	rabbit@wascallly.com
10004	Yosemite Place	829 Riverside Drive	Phoenix	AZ	88888	USA	Y Sam	sam@yosemite.com
10005	E Fudd	4545 53rd Street	Chicago	IL	54545	USA	E Fudd	NULL

4 rows in set (0.00 sec)

- Tabel Products

```
MariaDB [orderentry]> select * from products;
```

prod_id	vend_id	prod_name	prod_price	prod_desc
ANV01	1001	.5 ton anvil	6	.5 ton anvil, black, complete with handy hook
ANV02	1001	1 ton anvil	10	1 ton anvil, black, complete with handy hook and carrying case
ANV03	1001	2 ton anvil	15	2 ton anvil, black, complete with handy hook and carrying case
DTNTR	1003	Detonator	13	Detonator (plunger powered), fuses not included
FB	1003	Bird seed	10	Large bag (suitable for road runners)
FC	1003	Carrots	3	Carrots (rabbit hunting season only)
FU1	1002	Fuses	3	1 dozen, extra long
JP1000	1005	JetPack 1000	35	JetPack 1000, intended for single use
JP2000	1005	JetPack 2000	55	JetPack 2000, multi-use
OL1	1002	Oil can	9	Oil can, red
SAFE	1003	Safe	50	Safe with combination lock
SLING	1003	Sling	4	Sling, one size fits all
TNT1	1003	TNT (1 stick)	3	TNT, red, single stick
TNT2	1003	TNT (5 sticks)	10	TNT, red, pack of 10 sticks

14 rows in set (0.00 sec)

- Tabel vendors

```
MariaDB [orderentry]> select * from vendors;
```

vend_id	vend_name	vend_address	vend_city	vend_state	vend_zip	vend_country
1001	Anvils R Us	123 Main Street	Southfield	MI	48075	USA
1002	LT Supplies	500 Park Street	Anytown	OH	44333	USA
1003	ACME	555 High Street	Los Angeles	CA	90046	USA
1004	Furball Inc.	1000 5th Avenue	New York	NY	11111	USA
1005	Jet Set	42 Galaxy Road	London	NULL	N16 6PS	England
1006	Jouets Et Ours	1 Rue Amusement	Paris	NULL	45678	France

6 rows in set (0.00 sec)

d. Tabel orderitems

```
MariaDB [orderentry]> select * from orderitems;
```

order_num	order_item	prod_id	quantity
20005	1	ANV01	10
20005	2	ANV02	3
20005	3	TNT2	5
20005	4	FB	1
20006	1	JP2000	1
20007	1	TNT2	100
20008	1	FC	50
20009	1	FB	1
20009	2	OL1	1
20009	3	SLING	1
20009	4	ANV03	1

11 rows in set (0.00 sec)

2. Setelah keempat table diisi tampilkan query berikut ini
 - a. Ubahlah data salah satu customer dengan perintah update
 - b. Hapuslah data salah satu vendor atau product dengan perintah delete
 - c. Tampilkan nomer order dan jumlah nya
 - d. Tampilkan data customer yang berasal dari 'USA'.
3. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab5-NIM.odt
 - b. Isi file laporan :
 - i. Source SQL
 - ii. Screenshot CMD
 - c. Simpan di directory "PrakDB-NIM" .
4. Jawaban

6 BAB 6 – SINGLE ROW FUNCTION (DML)

6.1 IDENTITAS

Kompetensi

1. Praktikan memahami beberapa fungsi fungsi pada query single row
2. Praktikan dapat mengimplementasikan penggunaan query pada database

Topik

1. Pengurutan Data
2. Agregate function
3. Like , Between, having,
4. Ekspresi query
5. fungsi waktu

6.2 TEST AWAL

1. Tampilkan semua table pada skema orderentry antara lain table

- table customers
- table orderitems
- table orders
- table products
- table productnotes
- table vendors

2. Pastikan semua table telah terisi semua seragam sesuai dengan lampiran 1.

6.3 URUTAN DATA (ASC, DESC, ORDER BY)

6.3.1 Mengurutkan Data

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;
```

Contoh :

```
mysql> Select * From products Order By prod_name Asc;
```

artinya menampilkan data produk berdasarkan nama produk terurutan menaik a ke z

```
mysql> Select vend_name, vend_country From vendors Order By vend_country Desc;
```

artinya menampilkan data vendor berdasarkan nama vendor dan negaranya terurut menurut negara dari z ke a

```
MariaDB [orderentry]> select * from products order by prod_name asc;
```

prod_id	vend_id	prod_name	prod_price	prod_desc
ANV01	1001	.5 ton anvil	6	.5 ton anvil, black, complete with handy hook
ANV02	1001	1 ton anvil	10	1 ton anvil, black, complete with handy hook and carrying case
ANV03	1001	2 ton anvil	15	2 ton anvil, black, complete with handy hook and carrying case
FB	1003	Bird seed	10	Large bag (suitable for road runners)
FC	1003	Carrots	3	Carrots (rabbit hunting season only)
DTNTR	1003	Detonator	13	Detonator (plunger powered), fuses not included
FU1	1002	Fuses	3	1 dozen, extra long
JP1000	1005	JetPack 1000	35	JetPack 1000, intended for single use
JP2000	1005	JetPack 2000	55	JetPack 2000, multi-use
OL1	1002	Oil can	9	Oil can, red
SAFE	1003	Safe	50	Safe with combination lock
SLING	1003	Sling	4	Sling, one size fits all
TNT1	1003	TNT (1 stick)	3	TNT, red, single stick
TNT2	1003	TNT (5 sticks)	10	TNT, red, pack of 10 sticks

14 rows in set (0.06 sec)

```
MariaDB [orderentry]> select vend_name, vend_country from vendors order by vend_country desc;
```

vend_name	vend_country
Anvils R Us	USA
LT Supplies	USA
ACME	USA
Furball Inc.	USA
Jouets Et Ours	France
Jet Set	England

6 rows in set (0.00 sec)

6.4 AGREGATE FUNCTION

Fungsi agregat 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


```
MariaDB [orderentry]> select count(prod_id), sum(prod_price),avg(prod_price),min(prod_price),max(prod_price) from products;
+-----+-----+-----+-----+-----+
| count(prod_id) | sum(prod_price) | avg(prod_price) | min(prod_price) | max(prod_price) |
+-----+-----+-----+-----+-----+
|          14   |          226   |         16.1429 |                3 |                55 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

6.5 OPERATOR BETWEEN, IN, LIKE

Operator Between

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' ;

Contoh :

Select * From orderitems Where quantity Between '1' And '10' ;

```
MariaDB [orderentry]> select * from orderitems where quantity between 1 and 10;
```

order_num	order_item	prod_id	quantity
20005	1	ANV01	10
20005	2	ANV02	3
20005	3	TNT2	5
20005	4	FB	1
20006	1	JP2000	1
20009	1	FB	1
20009	2	OL1	1
20009	3	SLING	1
20009	4	ANV03	1

```
9 rows in set (0.08 sec)
```

Operator In

Operator In merupakan operator yang digunakan untuk mencocokkan suatu nilai.

Perintahnya :

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

Contoh :

Menampilkan nama customer, alamat dan email customer tertentu.

Select cust_name, cust_address, cust_email From customers Where cust_id In ('10002','10005');

```
MariaDB [orderentry]> select cust_name,cust_address,cust_email from customers where cust_id in('10002','10005');
```

cust_name	cust_address	cust_email
Mouse House	Gejayan Yogya	NULL
E Fudd	4545 53rd Street	NULL

```
2 rows in set (0.00 sec)
```

Operator Like

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

Perintahnya :

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

Contoh :

Select * From Products Where prod_name Like '%s' ;

Query yang pertama menampilkan produk dengan nama produk diawali huruf dan pada query yang kedua nama produk diakhiri huruf s.

```
MariaDB [orderentry]> select * from products where prod_name like 's%';
+-----+-----+-----+-----+-----+
| prod_id | vend_id | prod_name | prod_price | prod_desc |
+-----+-----+-----+-----+-----+
| SAFE    | 1003    | Safe      | 50         | Safe with combination lock |
| SLING   | 1003    | Sling     | 4          | Sling, one size fits all  |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

MariaDB [orderentry]> select * from products where prod_name like '%s';
+-----+-----+-----+-----+-----+
| prod_id | vend_id | prod_name | prod_price | prod_desc |
+-----+-----+-----+-----+-----+
| FC      | 1003    | Carrots   | 3          | Carrots (rabbit hunting season only) |
| FU1     | 1002    | Fuses     | 3          | 1 dozen, extra long |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

6.6 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;

Contoh :

**Select Kode_Mtkul As 'Kode Matakuliah',
Nama_Mtkul As 'Matakuliah' From Mtkul;**

```
MariaDB [orderentry]> select vend_name as 'Nama Produsen',vend_city as 'Kota Produksi' from vendors;
+-----+-----+
| Nama Produsen | Kota Produksi |
+-----+-----+
| Anvils R Us   | Southfield   |
| LT Supplies   | Anytown     |
| ACME          | Los Angeles  |
| Furball Inc.  | New York    |
| Jet Set       | London      |
| Jouets Et Ours | Paris       |
+-----+-----+
6 rows in set (0.00 sec)
```

Menambahkan Baris Teks Field Keluaran

Perintahnya :

Select 'Nama Field Tambahan', Nama_Field_Asal From Nama_Table;

Contoh :

```
Select vend_name, ' diproduksi di' , vend_city From vendors;
```

```
MariaDB [orderentry]> select vend_name as 'Nama Produsen','Produksi di',vend_city as 'Kota Produksi' from vendors;
```

Nama Produsen	Produksi di	Kota Produksi
Anvils R Us	Produksi di	Southfield
LT Supplies	Produksi di	Anytown
ACME	Produksi di	Los Angeles
Furball Inc.	Produksi di	New York
Jet Set	Produksi di	London
Jouets Et Ours	Produksi di	Paris

6 rows in set (0.00 sec)

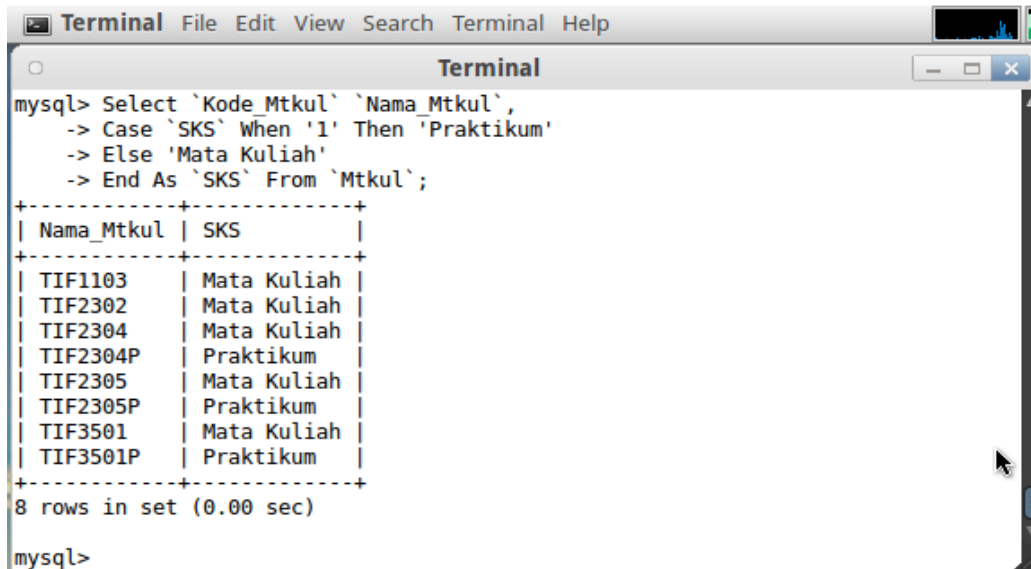
Ekspresi Kondisi

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;
```

Contoh :

```
Select Kode_Mtkul, Nama_Mtkul, Case Sks  
      When '1' Then 'Praktikum' Else 'Matakuliah'  
      End As Sks From Mtkul;
```



```
Terminal File Edit View Search Terminal Help
```

```
mysql> Select `Kode_Mtkul` `Nama_Mtkul`,  
-> Case `SKS` When '1' Then 'Praktikum'  
-> Else 'Mata Kuliah'  
-> End As `SKS` From `Mtkul`;
```

Nama_Mtkul	SKS
TIF1103	Mata Kuliah
TIF2302	Mata Kuliah
TIF2304	Mata Kuliah
TIF2304P	Praktikum
TIF2305	Mata Kuliah
TIF2305P	Praktikum
TIF3501	Mata Kuliah
TIF3501P	Praktikum

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

6.7 FUNGSI WAKTU

Beberapa Fungsi waktu dalam MySQL antara lain, seperti :

- Current_Date : Untuk menampilkan tanggal
- Current_Time : Untuk menampilkan waktu

Perintahnya :

```
Select Current_Date As 'waktu' ;
```

Contoh :

```
Select Current_Date As 'Tanggal Hari Ini' ;
```

```
MariaDB [orderentry]> select current_date as tanggal_hari_ini;
```

```
+-----+
| tanggal_hari_ini |
+-----+
| 2016-03-14       |
+-----+
```

```
1 row in set (0.00 sec)
```

```
MariaDB [orderentry]> select current_time as Jam_sekarang;
```

```
+-----+
| Jam_sekarang      |
+-----+
| 12:57:54          |
+-----+
```

```
1 row in set (0.00 sec)
```

Nilai waktu juga dapat digunakan untuk menampilkan waktu yang tersisa.

Contoh :

```
Select Time '16:18:14' - Time '08:12:10' ;
```

6.8 TEST AKHIR

1. Berdasarkan tabel yang telah dibuat pada praktikum sebelumnya untuk skema orderentry tampilkan data data berikut ini :
 - a. Tampilkan data customer secara terurut dari a sampai z
 - b. Tampilkan data vendor dan total vendors serta urutkan dari z ke a
 - c. Tampilkan data detail penjualan dari order item berdasarkan dari jumlah penjualan terbanyak.
 - d. Carilah banyaknya , total, rata-rata, jumlah minimal, jumlah maksimal dari quantity penjualan. Data berasal dari tabel orderitems.
 - e. Tampilkan nomer order yang memiliki total quantity antara 5-10.
 - f. Tampilkan data vendor id yang berasal dari 'USA'.
 - g. Tampilkan nama produk dan nama vendor yang mengandung huruf p.
 - h. Tampilkan data prod_name sebagai nama produk dan harganya.
2. Operasikan perintah SQL untuk menampilkan :
 - a. Tanggal praktikum
 - b. Jam sekarang
 - c. Jam sekarang – jam awal masuk praktikum
 - d. Tahun Sekarang – Tahun Lahir anda
3. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab6-NIM.odt

- b. Isi file laporan :
 - i. Source SSQL
 - ii. Screenshot CMD
- c. Simpan di directory "PrakDB-NIM" .

Jawaban

Nilai :	Asisten, (.....) Tanggal :
----------------	--

7 BAB 7 – JOIN DAN SUBQUERY (DML)

7.1 IDENTITAS

Kompetensi

1. Praktikan memahami cara joint table dan query lebih dari satu table
2. Praktikan mampu menggunakan jenis jenis joint dan penggunaanya

Topik

1. Macam macam clausa join
2. Join 3 table atau lebih

7.2 TEST AWAL

1. Tunjukkan print screen hasil semua table pada skema order entry, total 6 table.
2. Sebutkan macam macam join yang anda ketahui

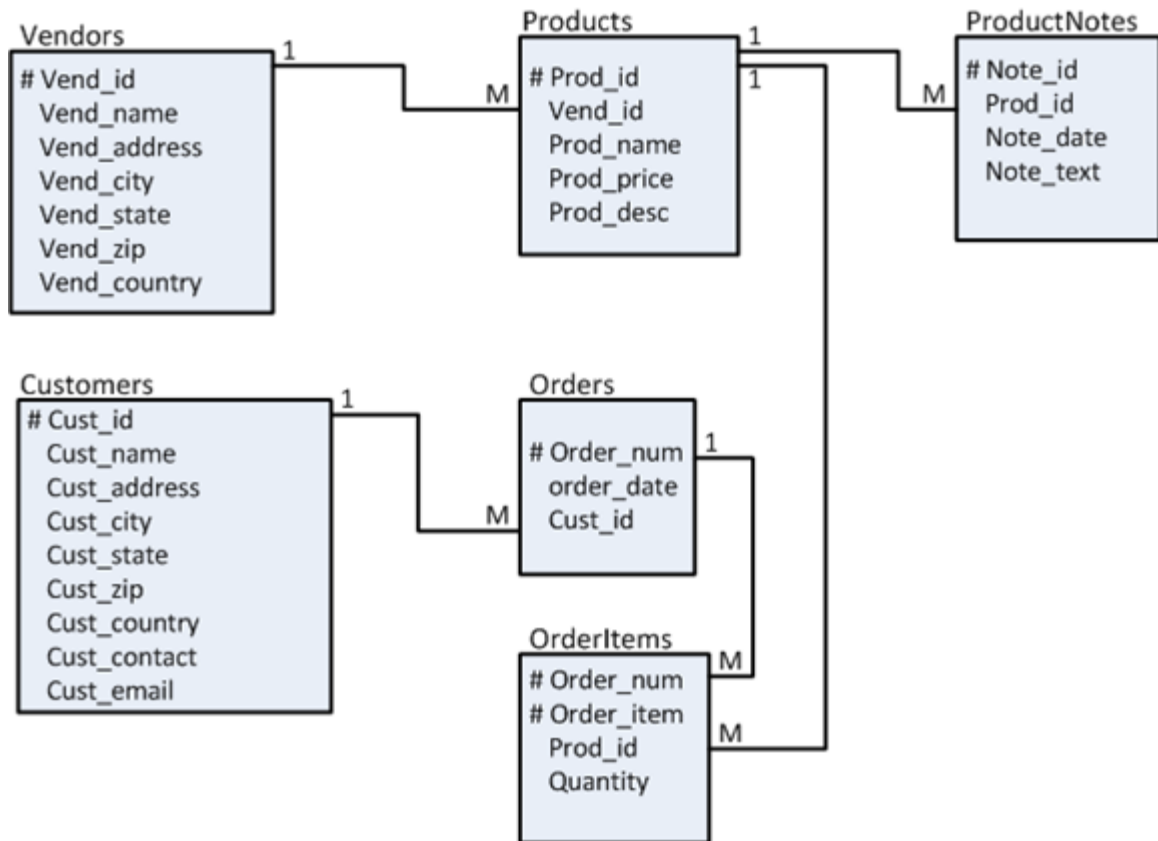
7.3 SELECTING DATA DENGAN JOIN TABLE

Join digunakan untuk menampilkan data dari gabungan dua tabel atau lebih. Ada dua jenis join inner join dan outer join. Inner join dibahas di bab ini, outer join dibahas di bab 8.

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)
```

Contoh penggunaan Join, kita lihat kembali skema order entry dibawah ini.



Contoh 1

Menampilkan prod_name, vend_name dari table vendors dan products.

Select vendors.vend_name, products.prod_name **from** vendors, products
Where vendors.vend_id = products.vend_id

```

MariaDB [orderentry]> select vendors.vend_name,products.prod_name
-> from vendors,products
-> where vendors.vend_id = products.vend_id;
  
```

vend_name	prod_name
Anvils R Us	.5 ton anvil
Anvils R Us	1 ton anvil
Anvils R Us	2 ton anvil
LT Supplies	Fuses
LT Supplies	Oil can
ACME	Detonator
ACME	Bird seed
ACME	Carrots
ACME	Safe
ACME	Sling
ACME	TNT (1 stick)
ACME	TNT (5 sticks)
Jet Set	JetPack 1000
Jet Set	JetPack 2000

14 rows in set (0.36 sec)

7.4 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;
```

```
MariaDB [orderentry]> select a.cust_name,b.order_date
-> from customers a join orders b on a.cust_id=b.cust_id
-> ;
```

cust_name	order_date
Yosemite Place	2005-09-30
E Fudd	2005-09-03

```
2 rows in set (0.00 sec)
```

7.5 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;
```

```
MariaDB [orderentry]> 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;
```

cust_name	order_date	quantity
Yosemite Place	2005-09-30	100
E Fudd	2005-09-03	50

```
2 rows in set (0.06 sec)
```

7.6 TEST AKHIR

1. Masih dengan skema order entry jawablah pertanyaan berikut ini
 1. Carilah nama vendor yang harga barangnya termurah.
 2. tampilkan kota -kota dari vendor product jetpack harganya dibawah 55 .
 3. siapa sajakah customer yang melakukan order tanggal 01-09-2005
 4. dimanakah alamat lengkap customer(Kota,negara,kodepos) yang melakukan pembelian barang sama dengan customer no 10001 ?

5. siapakah nama customer dan email kontaknya , yang paling boros belanja (dilihat dari pembelian barang paling banyak) ?

2. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab7-NIM.odt
 - b. Isi file laporan :
 - iii. Source SQL
 - iv. Screenshot CMD
 - c. Simpan di directory "PrakDB-NIM" .

--	--

Nilai :	Asisten, (.....) Tanggal :
----------------	--

8 BAB 8 – ADVANCE JOIN, TRIGGER, VIEW (DML)

8.1 IDENTITAS

Kompetensi

1. Praktikan memahami perbedaan macam macam join, trigger dan view
2. Praktikan dapat menerapkan penggunaan join,trigger dan view

Topik

1. Righ Join
2. Left Join
3. SelJoint
4. Trigger
5. View

8.2 TEST AWAL

Kerjakan pada satu lembar kertas dan kumpulkan sebelum memulai praktikum , Beri nim , nama dan kelas.

1. Apa arti dari righ join, Left join, inner join
2. Jelaskan pengertian trigger dan view

8.3 OUTER JOIN

Pada OUTER JOIN hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang **hanya** yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data *foreign key* dengan *primary key* tabel yang diacu) maupun data yang tidak memiliki kesamaan data berdasarkan relasinya dari salah satu tabel. Terdapat dua tipe OUTER JOIN, yaitu:

1. LEFT OUTER JOIN atau biasa disebut left join
2. RIGHT OUTER JOIN atau biasa disebut righ join

8.4 LEFT JOIN

Pada LEFT JOIN *output*/hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang hanya yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data *foreign key* dengan *primary key* tabel yang diacu) maupun data-data yang tidak memiliki kesamaan data berdasarkan relasinya dari tabel sebelah **kiri** dari klausa LEFT JOIN. Berikut adalah bentuk umum:

```

SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2,
          nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2
FROM    tabel1
LEFT JOIN tabel2
ON      tabel1.nama_kolom1(primary key)=tabel2.nama_kolom1(foreign key yg mengacu ke
tabel1)
WHERE kondisi;

```

Contoh :

```

select a.cust_name,b.order_date
from customers a left join orders b on a.cust_id=b.cust_id

```

```

MariaDB [orderentry]> select a.cust_name,b.order_date
-> from customers a left join orders b on a.cust_id=b.cust_id;
+-----+-----+
| cust_name | order_date |
+-----+-----+
| Yosemite Place | 2005-09-30 |
| E Fudd | 2005-09-03 |
| Mouse House | NULL |
| WascaIs | NULL |
+-----+-----+
4 rows in set (0.00 sec)

```

8.5 RIGHT JOIN

Pada RIGHT JOIN *output*/hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang hanya yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data *foreign key* dengan *primary key* tabel yang diacu) maupun data-data yang tidak memiliki kesamaan data berdasarkan relasinya dari tabel sebelah **kanan** dari klausa RIGHT JOIN.

```

SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2,
          nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2
FROM    tabel1
RIGHT JOIN tabel2
ON      tabel1.nama_kolom1(primary key)=tabel2.nama_kolom1(foreign key yg
mengacu ke tabel1)
WHERE kondisi;

```

Contoh :

```

select a.cust_name,b.order_date
from customers a right join orders b on a.cust_id=b.cust_id

```

```
MariaDB [orderentry]> select a.cust_name,b.order_date
-> from customers a right join orders b on a.cust_id=b.cust_id;
```

cust_name	order_date
NULL	2005-09-01
Yosemite Place	2005-09-30
E Fudd	2005-09-03
NULL	2005-08-10

4 rows in set (0.00 sec)

8.6 SELF JOIN

Self join adalah melakukan join dengan dirinya sendiri. Atau join dengan table yang sama.

Sintak nya sbb :

```
select nama alias1_table.kolom1, nama alias2_table.kolom2,
from table alias1 inner join table alias2 on alias1.kolom3=alias2.kolom3
```

contoh

```
select a.vend_name,b.vend_state, 'negaranya' ,b.vend_country
from vendors a inner join vendors b on a.vend_id=b.vend_id
```

```
MariaDB [orderentry]> select a.vend_name,b.vend_state, 'negaranya' ,b.vend_country
-> from vendors a inner join vendors b on a.vend_id=b.vend_id;
```

vend_name	vend_state	negaranya	vend_country
Anvils R Us	MI	negaranya	USA
LT Supplies	OH	negaranya	USA
ACME	CA	negaranya	USA
Furball Inc.	NY	negaranya	USA
Jet Set	NULL	negaranya	England
Jouets Et Ours	NULL	negaranya	France

6 rows in set (0.25 sec)

8.7 TRIGGER

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.

```
MariaDB [orderentry]> delete from products where prod_id='tnt3';  
Query OK, 1 row affected (0.16 sec)
```

```
MariaDB [orderentry]> select * from produk_hapus;
```

prod_id	vend_id	prod_name	prod_price	prod_desc	tgl_hapus	nama_user
Tnt3		hanger	100	NULL	2016-03-18	root@localhost

1 row in set (0.00 sec)

8.8 VIEW

View adalah salah satu *object database*, yang secara logika merepresentasikan sub himpunan dari data yang berasal dari satu atau lebih table. Kegunaan untuk mempermudah pengaksesan data transaksi yang sering di lihat tanpa harus merubah struktur table atau menambah tabel baru.

Sintaks nya adalah

```
create view namaview as [query]
```

contohnya :

```
create view namaview as
```

```
select cust_name,cust_country from customers
```

menampilkan nama customer dan negaranya.

```
MariaDB [orderentry]> create view view1 as  
-> select cust_name,cust_country from customers  
-> ;  
Query OK, 0 rows affected (0.14 sec)
```

```
MariaDB [orderentry]> select * from view1;
```

cust_name	cust_country
Mouse House	INA
Wasca's	USA
Yosemite Place	USA
E Fudd	USA

4 rows in set (0.05 sec)

8.9 TEST AKHIR

1. Masih menggunakan order entry buatlah sebuah query untuk masing masing :
 - a. join 2 table clause where
 - b. Query dengan join on dari 3 table
 - c. Right join
 - d. Left join

- e. Self join
 - f. view
2. Buatlah triger after delete untuk salah satu table di order entry. Serta tunjukan hasil akibat penerapan trigger.
 3. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab8-NIM.odt
 - b. Isi file laporan :
 - i. Source SSQL
 - ii. Screenshot CMD
 - c. Simpan di directory "PrakDB-NIM" .

7.6 Jawaban

Nilai :	Asisten,
	(.....)
	Tanggal :

9 BAB 8 – DATA CONTROL LANGUAGE (DCL)

9.1 IDENTITAS

Kompetensi

1. Praktikan memahami tentang Data Control Language

Topik

1. Data Control Language

9.2 TEST AWAL

Kerjakan pada satu lembar kertas , beri nama dan nim serta kelas kumpulkan sebelum praktikum dimulai. Jawablah pertanyaan dibawah ini :

1. Apakah yang anda ketahui tentang Data Control Language?
2. Berilah satu contoh DCL !

9.3 USER

Sebagai seorang administrator database, user ROOT mempunyai hak dalam membuat user dan memberikan hak-hak akses terhadap user baru tersebut. Adapun struktur dari Table User adalah sebagai berikut :

Field	Type	NULL	Key	Default	Extra
Host	char(60)		PRI		
User	char(16)		PRI		
Password	char(16)				
Select_priv	enum('N','Y')			N	
Insert_priv	enum('N','Y')			N	
Update_priv	enum('N','Y')			N	
Delete_priv	enum('N','Y')			N	
Create_priv	enum('N','Y')			N	
Drop_priv	enum('N','Y')			N	
Reload_priv	enum('N','Y')			N	
Shutdown_priv	enum('N','Y')			N	
Process_priv	enum('N','Y')			N	
File_priv	enum('N','Y')			N	
Grant_priv	enum('N','Y')			N	

Menambahkan User Baru

> Insert Into user (host,user,password)


```
Values ( 'localhost', 'kulo', password('nuwun'));  
> Flush Privileges;
```

```
MariaDB [mysql]> use mysql;  
Database changed  
MariaDB [mysql]> select * from user;  
MariaDB [mysql]> insert into user(user,host,password) values ('kulo','localhost','nuwun');  
Query OK, 1 row affected, 4 warnings (0.25 sec)  
  
MariaDB [mysql]> flush privileges;  
Query OK, 0 rows affected (0.44 sec)  
  
MariaDB [mysql]> select * from user;
```

Keterangan :

- Host : Localhost
User hanya dapat mengakses MySQL hanya di komputer local / server saja.
- IP Address
User hanya dapat mengakses MySQL di komputer dengan alamat IP yang telah didefinisikan.
- %
User dapat mengakses MySQL dari komputer manapun.
- User
Nama atau ID yang digunakan untuk Login.
- Password
Digunakan untuk keamanan server database dan Password di enkripsi untuk keamanan user.
- Flush Privileges
Wajib diberikan untuk menetapkan user dalam server dan digunakan selain pembuatan user baru juga dalam mengedit user maupun menghapus user dari server.

```
MariaDB [mysql]> select host, user,password from user;  
+-----+-----+-----+  
| host      | user  | password |  
+-----+-----+-----+  
| localhost | root  |          |  
| 127.0.0.1 | root  |          |  
| ::1       | root  |          |  
| localhost | pma   |          |  
| localhost | kulo  | nuwun    |  
+-----+-----+-----+  
6 rows in set (0.00 sec)
```

9.4 HAK AKSES USER

DCL merupakan kontrol keamanan terhadap database dan tabelnya., yaitu mengatur hak akses dan cara mencabut hak akses, agar tabel-tabel tertentu hanya bisa diakses oleh

orang-orang yang dikehendaki
Setiap User mempunyai 14 Hak Akses, yaitu :

Nama Field	Type	Default	Keterangan
Select_priv	('N','Y')	N	Select (Table)
Insert_priv	('N','Y')	N	Insert (Table)
Update_priv	('N','Y')	N	Update (Table)
Delete_priv	('N','Y')	N	Delete (Table)
Create_priv	('N','Y')	N	Create (Database,Table,index)
Drop_priv	('N','Y')	N	Drop (database,table)
Reload_priv	('N','Y')	N	Reload (Server Administration)
Shutdown_priv	('N','Y')	N	Shutdown (Server dministration)
Process_priv	('N','Y')	N	Prosess (server administration)
File_priv	('N','Y')	N	FILE (Akses File di server)
Grant_priv	('N','Y')	N	Grant (Database,table)
References_Priv	('N','Y')	N	References (Database,Table)
Index_Priv	('N','Y')	N	Index (Table)
Alter_Priv	('N','Y')	N	Alter (Table)

9.5 GRANT

Grant digunakan untuk mengizinkan seorang user mengakses tabel dalam database tertentu. Pemberian hak akses ini dengan clausa GRANT.

Perintah :

```
Grant hak_akses On Nama_Tabel To Nama_User
```

Keterangan :

- Hak Akses adalah hak-hak yang diberikan server administrator kepada user, antara lain : ALTER, CREATE, DELETE, DROP, UPDATE, INSERT, FILE, PROCESS, RELOAD, REFERENCES,LOAD, SHUTDOWN DAN USAGE
- Nama_Tabel adalah nama-nama tabel yang akan akan diakses atau pemberian hak kepada user.
- Pemakai adalah nama user yang akan diberi hak, dengan ketentuan nama pemakai diikuti nama dari host diawali tanda @.

Contoh :

- Diberikan semua hak akses semua tabel dalam database praktikum terhadap user **kulo** di localhost :

```
> Grant all privileges On praktikum.* To kulo@localhost;
> Flush Privileges;
```

```
MariaDB [mysql]> Grant all privileges On praktikum.* To kulo@localhost;
Query OK, 0 rows affected (0.13 sec)
```

```
MariaDB [mysql]> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

- Diberikan hak akses insert dan select dalam database orderentry dengan tabel customers terhadap user kulo di localhost

- > **Grant** select,insert On orderentry.customers To kulo@localhost;
- > **Flush Privileges**;

```
MariaDB [mysql]> Grant select,insert On orderentry.customers To kulo@localhost;
Query OK, 0 rows affected (0.05 sec)
```

```
MariaDB [mysql]> flush privileges;
Query OK, 0 rows affected (0.06 sec)
```

9.6 REVOKE

Revoke digunakan untuk Mencabut hak akses seorang user mengakses tabel dalam database tertentu. Pencabutan hak akses ini dengan clausa REVOKE.

Perintah :

```
Revoke hak_akses On Nama_Tabel From Nama_User
```

Keterangan :

- Hak Akses adalah hak-hak yang diberikan server administrator kepada user, antara lain : ALTER, CREATE, DELETE, DROP, UPDATE, INSERT, FILE, PROCESS, RELOAD, REFERENCES,LOAD, SHUTDOWN DAN USAGE
- Nama_Tabel adalah nama-nama tabel yang akan akan diakses atau pemberian hak kepada user.
- Pemakai adalah nama user yang akan diberi hak, dengan ketentuan nama pemakai diikuti nama dari host diawali tanda @.

Contoh :

- Dicabut semua hak akses semua tabel dalam database praktikum terhadap user dnd di localhost.

- > **Revoke** all privileges **On** praktikum.* **From** kulo@localhost;
- > **flush privileges**;

```
MariaDB [(none)]> Revoke all privileges On praktikum.* From kulo@localhost;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

- Dicabut hak akses insert dan select dalam database orderentry dengan tabel customers terhadap user kulo di localhost.

- > **Revoke** select, insert **On** orderentry. customers **From** kulo@localhost;
- > **flush privileges**;

```
MariaDB [(none)]> Revoke select,insert On orderentry.customers From kulo@localhost;
Query OK, 0 rows affected (0.05 sec)
MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.02 sec)
```

9.7 TEST AKHIR

4. Buatlah sebuah user baru dengan nama anda
5. Berikan/Cabut hak akses terhadap user baru tersebut dengan grant insert pada table vendors.
6. Cek hasil pemberian hak akses dengan melakukan login dan insert/update/delete data table vendors menggunakan user tersebut
7. Cabut hak akses tersebut dengan revoke.
8. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - d. Nama file laporan : PrakDB_Bab9-NIM.odt
 - e. Isi file laporan :
 - iii. Source SSQL
 - iv. Screenshot CMD
 - f. Simpan di directory "PrakDB-NIM" .

Jawaban

Nilai :	Asisten, (.....) Tanggal :

10 BAB 10 – PHP MY ADMIN

10.1 IDENTITAS

Kompetensi

1. Praktikan mengetahui dan memahami relasi dengan phpMyadmin designer, operasi input,edit,delete dengan melibatkan relasi antar table

Topik

1. SQL
2. PhpMyAdmin

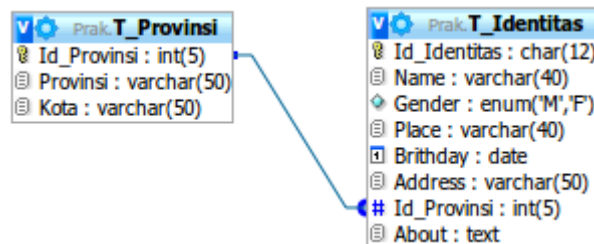
10.2 TEST AWAL

1. Pastikan komputer anda telah terinstal phpMyAdmin
2. Apakah yang ada ketahui tentang relasi antar table

10.3 SQL

Terlebih dahulu buat **Database Prak**, Selanjutnya buat **Tabel Provinsi** dan **Table Identitas** dengan struktur sebagai berikut :

Relasi Antar Table :



Tabel T_Provinsi

```
MySQL 5.5 Command Line Client
mysql> Desc `T_Provinsi`;
+----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+----+-----+-----+-----+-----+-----+
| Id_Provinsi | int(5)        | NO   | PRI | NULL    | auto_increment |
| Provinsi    | varchar(50)   | NO   |     | NULL    |                |
| Kota        | varchar(50)   | NO   |     | NULL    |                |
+----+-----+-----+-----+-----+-----+
3 rows in set (0.20 sec)

mysql>
```

Tabel T Identitas :

```

MySQL 5.5 Command Line Client
mysql> Desc `T_Identitas`;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Id_Identitas   | char(12)      | NO   | PRI | NULL    |       |
| Name           | varchar(40)   | NO   |     | NULL    |       |
| Gender         | enum('M','F') | NO   |     | NULL    |       |
| Place          | varchar(40)   | NO   |     | NULL    |       |
| Brithday       | date          | NO   |     | NULL    |       |
| Address        | varchar(50)   | NO   |     | NULL    |       |
| Id_Provinsi    | int(5)        | YES  |     | NULL    |       |
| About          | text          | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql> _

```

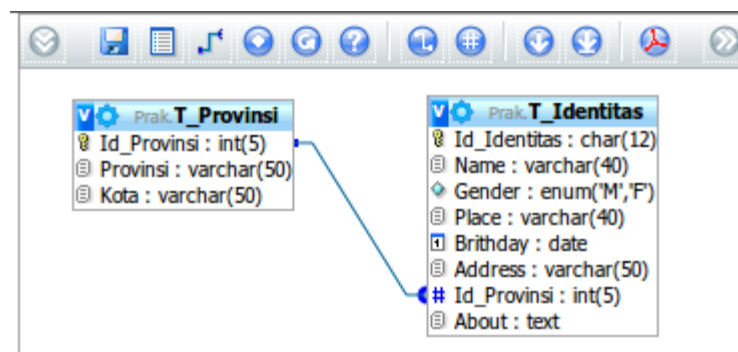
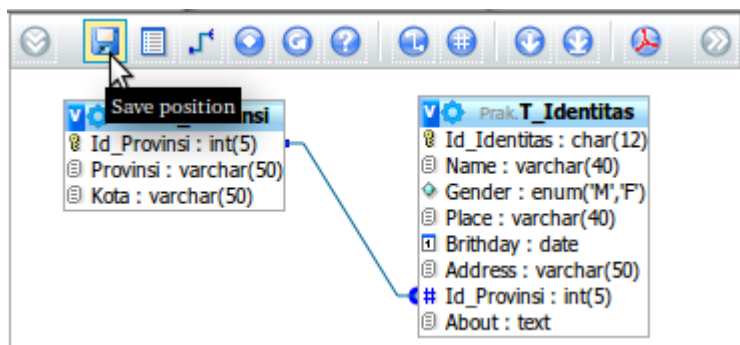
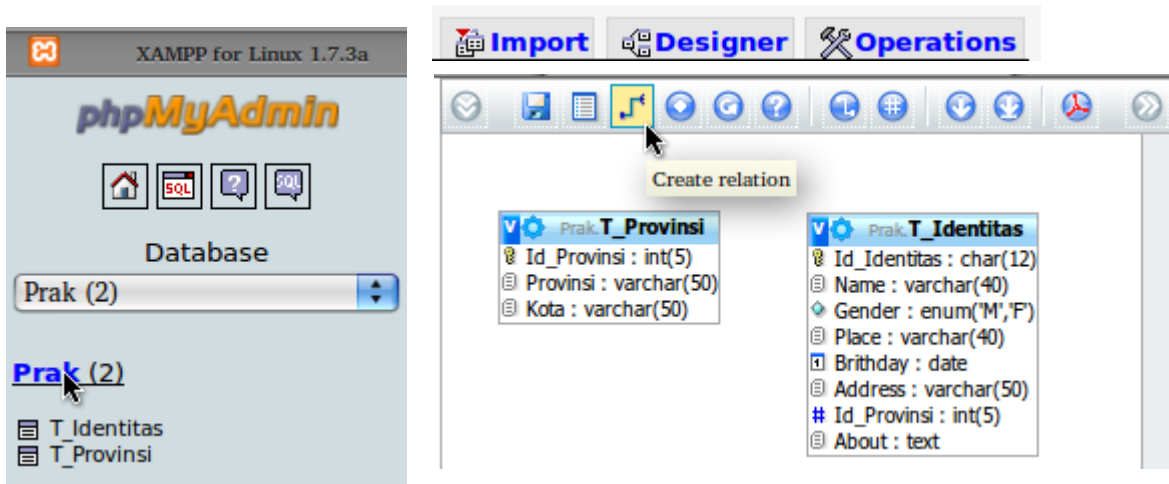
Selanjutnya isi table Provinsi dan Identitas dengan data berikut :

Table - (DBPrak).Prak.t_provinsi ×			
	Id_Provinsi	Provinsi	Kota
▶	1	DKI Jakarta	Jakarta
	2	Banten	Banten
	3	Jawa Barat	Bandung
	4	Jawa Tengah	Semarang
	5	DI Yogyakarta	Yogyakarta
	6	Jawa Timur	Semarang

Table - (DBPrak).Prak.t_identitas ×		Table - (DBPrak).Prak.t_provinsi					
	Id_Identitas	Name	Gender	Place	Brithday	Address	Id_Provinsi
	09102084	Widodo	M	Sleman	12/15/1987	Sleman	5
	09111057	Taufan Ardi Wahyuda	M	Sleman	1/12/1987	Sleman	5
	09111086	Tri Suwarno	M	Sleman	12/13/1990	Sleman	5
	09112022	Setiorini	F	Lampung	12/12/1987	Wates	5
	09112033	Yuli Antoro	M	Bantul	1/14/1986	Bantul	5
	09112064	Nur Suci Amashanti	F	Gunung Kidul	1/15/1986	Gunung Kidul	5
	09112081	Zaenal Abidin	M	Cilacap	1/11/1987	Cilacap	4
	10112008	Eka Yani Arsari	F	Solo	1/12/1991	Solo	4
	10112038	Anik Tataria	F	Banyumas	9/9/1987	Banyumas	4
	10112077	Dian Sulistywo Widodo	M	Kebumen	11/11/1986	Kebumen	4
▶	10112099	Sri Sumarahati	F	Kebumen	10/10/1987	Kebumen	4

10.4 RELASI TABLE

Relasi Database dengan phpMyAdmin Designer



10.5 TEST AKHIR

1. Tuliskan perintah SQL untuk membuat tabel T_Provinsi dan T_Identitas pada pembahasan sub bab 10.3 di atas, dengan syarat sebagai berikut :
 - a. Struktur tabel sesuai dengan di atas
 - b. Menggunakan keyword untuk merelasikan tabel
 - c. Dengan menggunakan perintah SQL, tambahkan kolom “No_Telp” pada tabel T_Identitas (pilih tipe data yang sesuai dan berikan alasan anda mengenai tipe data yang dipilih tersebut).
 - d. Dengan menggunakan perintah SQL Tambahkan tabel “T_Kabupaten” selanjutnya relasikan dengan tabel yang sudah ada (T_Identitas dan/atau T_Provinsi).
 - e. Dengan menggunakan perintah SQL, isikan data T_Kabupaten minimal 5 data.
 - f. Tuliskan perintah SQL untuk menampilkan data :
Nama Mahasiswa “Setiorini” berasal dari Provinsi “Lampung” dan Kabupaten “Lampung Tengah”.
2. Dengan menggunakan Designer (di PhpMyAdmin), buatlah relasinya antar tabel tersebut.
3. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab10-NIM.odt
 - b. Isi file laporan :
 - v. Source SSQL
 - vi. Screenshot CMD
 - c. Simpan di directory “PrakDB-NIM” .

Jawaban

Nilai :	Asisten, (.....) Tanggal :

11 BAB 11 – STUDI KASUS SQL

11.1 IDENTITAS

Kompetensi

1. Mampu mengetahui dan memahami relasi dengan phpMyAdmin Designer, Operasi Input, Edit, Delete MySQL Dengan Melibatkan Relasi Antar Table Dengan PHP.

Topik

1. Studi kasus

11.2 STUDI KASUS

Kasus 1

No	Nama Tabel
1	Mahasiswa
2	KRS
3	Matakuliah
4	Program Studi
5	Fakultas
6	Provinsi
7	Kelas
8	Dosen
9	DPA
10	Agama
11	Jadwal Kuliah

Kasus 2

No	Nama Tabel
1	Member
2	Ongkos Kirim
3	Pembayaran
4	Pemesanan
5	Jenis Buku
6	Buku
7	Provinsi
8	Berita
9	Kategori Berita
10	Admin
11	Buku Tamu

11.3 TEST AKHIR

1. Pilihlah salah satu kasus di atas, selanjutnya rancanglah struktur masing-masing tabel, dengan syarat tabel yang dirancang memenuhi kriteria Normalisasi 3 NF.
2. Buatlah relasi database dengan menggunakan PhpMyAdmin Designer
3. Isi sampel data masin-masing tabel minimal 5 data
4. Operasikan perintah SQL untuk :
 - a. Kasus 1
 - i. Mahasiswa melihat KRS
 - ii. Mahasiswa melihat Jadwal
 - iii. Mahasiswa melihat DPA nya
 - b. Kasus 2
 - i. Member melakukan pemesanan
 - ii. Member melakukan pembayaran
 - iii. Member mengetahui ongkos kirim untuk provinsinya “misal di Yogyakarta”
5. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab11-NIM.odt
 - b. Isi file laporan :
 - i. Source SQL
 - ii. Screenshot CMD
 - iii. Screenshot relasi dengan phpMyDesigner
 - c. Simpan di directory “PrakDB-NIM” .

Jawaban

Nilai :	Asisten, (.....) Tanggal :

12 BAB 12 – STUDI KASUS NORMALISASI

12.1 IDENTITAS

Kompetensi

1. Memantapkan pemahaman praktikan tentang NORMALISASI.

Topik

1. Studi Kasus pada bab 11

12.2 STUDI KASUS

1. Perhatikan tata cara pada praktikum sebelumnya
2. Berdasarkan studi kasus yang telah ditentukan pada praktikum 1 (ditentukan oleh asisten)

12.3 TEST AKHIR

1. Berdasarkan petunjuk pada poin 10.3, Rancanglah struktur masing-masing tabel, dengan syarat tabel yang dirancang memenuhi kriteria Normalisasi 3 NF.
2. Buatlah relasi database dengan menggunakan PhpMyAdmin Designer
3. Isi sampel data masing-masing tabel minimal 5 data
4. Operasikan perintah SQL untuk (instruksi asisten per masing-masing kasus) :
5. Buatlah laporan praktikum dengan ketentuan sebagai berikut :
 - a. Nama file laporan : PrakDB_Bab12-NIM.odt
 - b. Isi file laporan :
 - i. Rancangan spesifikasi baik nama tabel, tipe data, lebar tipe data, PK dan FK
 - ii. Source SQL
 - iii. Screenshot CMD
 - iv. Screenshot relasi dengan phpMyDesigner
 - c. Simpan di directory "PrakDB-NIM" .

Jawaban

--	--

Nilai :

Asisten,

(.....)
Tanggal :

13 BIBLIOGRAPHY

- [1] S. Sumathi and S. Esakkirajan, Fundamental of Relational Database Management Systems, Verlag Berlin Heidelberg: Springer, 2007.
- [2] S. Bagui and R. Earp, Database Design Using Entity-Relationship Diagrams, Boca Raton, Florida: Auerbach Publications, 2003.
- [3] Silberschatz, Korth and Sudarshan, Database System Concepts Fourth Edition.