UNIVERSITAS PAHLAWAN

MODUL

PRAKTIKUM ANALISA DAN PERANCANGAN SISTEM INFORMASI

2024-2025

Lailatul Syifa Tanjung,S.T,M.T

KATA PENGANTAR

Assalaamu'alaikum warohmatullaahi wabarakaatuh Alhamdulillaahirobbil'aalamiin.

Puji dan syukur tim penulis panjatkan kehadirat Allaah Subhaanahu wa ta'ala atas segala petunjuk, kasih sayang, ridho dan nikmat-Nya sehingga tim dapat menyelesaikan petunjuk praktikum analisa dan perancangan sistem informasi. Petunjuk praktikum ini merupakan bahan penunjang pelaksanaan praktikum untuk mata kuliah analisa dan perancangan sistem informasi pada program studi teknik industri di Universitas Pahlawan Tuanku Tambusai.

Praktikum dilaksanakan agar mahasiswa mampu mencapai kemampuan akhir yang diharapkan dari mata kuliah ini yaitu mampu memodelkan dan merancang sistem infomasi dengan rancangan basis data. Selain itu diharapakan mahasiswa mampu membuat sistem informasi sederhana dari proses bisnis yang dibuat. Diharapkan praktikum akan memperkuat kompetensi akhir dari mata kuliah analisa dan perancangan sistem informasi.

Tim penulis menyadari bahwa petunjuk praktikum ini masih jauh dari sempurna, perbaikan berkelanjutan akan terus dilakukan di lapangan seiring melihat perkembangan proses pembelajaran di kelas. Kritik dan saran yang membangun akan diharapkan untuk perbaikan petunjuk praktikum ini. Akhir kata, tim penulis berharap petunjuk praktikum dapat memberikan manfaat.

Wassalaamu'alaikum warohmatullaahi wabarakaatuh

Bangkinang, Maret 2023

Tim Praktikum Analisa dan
Perancangan Sistem Informasi

DAFTAR ISI

MODUL I
IDENTIFIKASI PERMASALAHAN

A. Tujuan Praktikum

1. Mahasiswa mampu mengambil data di lapangan dengan metode yang ada.
2. Mahasiswa mampu memahami dan mengetahui simbol-simbol pada *flowmap*.
3. Mahasiswa mampu membuat *flowmap* untuk proses bisnis saat ini berdasarkan pengambilan data.
4. Mahasiswa mampu membuat *flowmap* revisi perbaikan proses bisnis.

B. Teknik Pengambilan Data di Lapangan

Beberapa teknik yang dapat digunakan untuk mengambil data di lapangan untuk merumuskan aliran proses bisnis yang sekarang ada, adalah sebagai berikut (Fatta, 2007):

a. Wawancara

Wawancara adalah teknik pengumpulan kebutuhan yang paling umum digunakan. Langkah-langkah dasar dalam teknik wawancara adalah:

i. Memilih target wawancara.

ii. Mendesain pertanyaan-pertanyaan untuk wawancara.

iii. Persiapan wawancara

iv. Melakukan wawancara

v. Menindaklanjuti hasil wawancara.

Wawancara adalah metode yang paling mudah digunakan, jika sistem yang dianalisis tidak terlalu besar (Fatta, 2007). Sebagai contoh, untuk melakukan wawancara pada bagian gudang di sebuah UMKM karena personelnya tidak banyak. Tetapi jika sistem informasi yang dibangun berskala *enterprise*, metode wawancara akan memakan waktu yang

sangat besar karena banyak departemen-departemen yang harus diwawancarai secara terpisah (Fatta, 2007).

b. Kuesioner

Kuesioner adalah sekumpulan pertanyaan tertulis dan biasanya melibatkan banyak orang (Fatta, 2007). Kuesioner bisa berbentuk tertulis (*paper-based*) atau *online*. Bila melakukan penyebaran kuesioner maka aturan sampel dan populasi berlaku dalam pengambilan datanya. Sampel yang dipilih harus mewakili populasinya. Setelah hasil kuesioner diambil maka diperlukan analisis untuk mengambil data yang sesuai dengan kebutuhan pengumpulan kebutuhan (Fatta, 2007).

c. Analisis dokumen

Teknik ini dilakukan dengan mempelajari material yang menggambarkan sistem yang sedang berjalan. Biasanya dokumen yang diamati adalah form, laporan, manual kebijakan, grafik organisasi. Untuk perusahaan atau organisasi berskala kecil dan belum memiliki sistem yang terkomputerisasi. Cara ini adalah cara yang efektif untuk menyusun kebutuhan sistem (Fatta, 2007).

d. Observasi

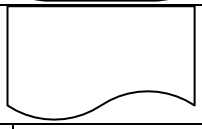
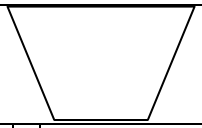
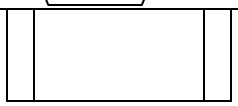
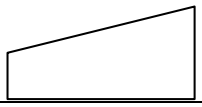
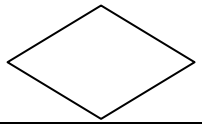
Teknik observasi dilakukan dengan melakukan pengamatan secara langsung pada proses-proses yang sedang berjalan. Hal ini penting karena kadang-kadang manajer tidak dapat mengingat secara keseluruhan apa yang dilakukan dan menceritakan kembali kepada analis. Teknik observasi biaanya dilakukan bersama-sama dengan teknik pengumpulan data yang lain (Fatta, 2007)

C. Flow Map

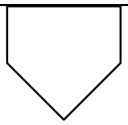
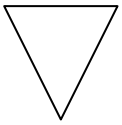
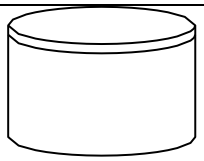
1. Dasar Teori

Flow Map adalah campuran peta dan *Flow Chart* yang menunjukan pergerakan objek dari satu lokasi ke lokasi lain. *Flow Map* mempunyai fungsi sebagai mendefinisikan hubungan antara bagian (pelaku proses), proses (manual maupun berbasis komputer) dan aliran data (dalam bentuk dokumen keluaran dan masukan). Adapun simbol *FlowMap* yang digunakan dapat dilihat pada tabel 3.1.

Tabel 3.1 Simbol *Flow Map*

Simbol	Keterangan
	*Terminator*, Digunakan mulai atau berakhirnya kegiatan
	Dokumen atau Formulir, Menunjukan I/O untuk proses manual, mekanik dan komputer
	Proses, Menunjukan kegiatan proses dari operasi program komputer
	*Manual Operation*, Menunjukan kegiatan/proses manual
	*Sub Process*, menujukan proses yang lebih rinci penjelasnya
	*Manual Input*, Menujukan input manual yang menggunakan online keyboard
	*Decision*, Menunjukan pilihan yang akan dikerjakan atau keputusan yang harus dibuat dalam proses pengolahan data
	*Connector (on-page)*, digunakan penghubung dalam satu halaman.

Tabel 3.2 Simbol *Flow Map*

Simbol	Keterangan
	Catatan, digunakan untuk menggambarkan catatan yang digunakan untuk mencatat data dalam dokumen atau formulir
	*Connector (Off-page)*, digunakan dalam penghubung berbeda halaman.
	*Off Line Storage*, digunakan untuk menyimpan data secara manual. Jika A berarti data disimpan menurut abjad, N berarti Nomor dan T menurut kronologis atau menurut tanggal
	Eksternal Data, Digunakan untuk menyimpan data (Hard Disk, Flash Disk, Memory)
	Penyimpanan/*Storage*, Menunjukan akses langsung perangkat penyimpanan pada disket

2. Cara Pembuatan Flowmap

Langkah-langkah dalam pembuatan *flowmap* adalah sebagai berikut:

1. Pembuatan pertama adalah membagi-bagi diagram ke dalam kolom- kolom.
2. Setiap kolom diberi nama pelaku proses yang terlibat.
3. Diagram dibaca dari atas ke bawah dan dari kiri ke kanan.
4. Setiap kolom terdapat siklus pengolahan data Input - Proses - Output.
5. Ketika meyeberangi garis yang memisahkan antara satu kolom dengan kolom lain, gunakan Simbol Konektor.
6. Cara mengakses file komputer adalah melalui simbol proses komputer (berbentuk kotak).
7. Prosedur kerja yang kejadiannya tidak bersamaan dapat digambarkan melalui *Flowmap* yang terpisah.

D. Latihan

Buatlah sebuah perancangan *flowmap* dengan tema mengenai proses penjualan *online* yang terjadi di beberapa sistem (contoh Lazada, bukalapak). Gambarkan *flowmap* saat ini yang menggambarkan proses penjualan *online* tersebut antara pembeli, penjual, dan bank.

MODUL II
DIAGRAM KONTEKS DAN DFD

A. Tujuan Praktikum

1. Mahasiswa mampu memahami dan mengetahui simbol-simbol pada diagram konteks dan DFD.
2. Mahasiswa mampu merancang diagram konteks dan DFD.

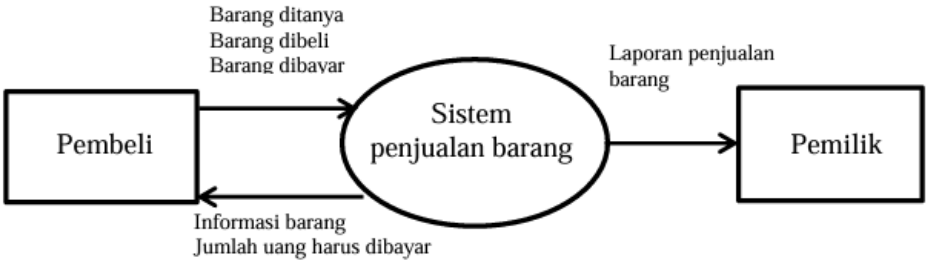
B. Diagram Konteks

Diagram konteks merupakan diagram alir data yang terdiri dari suatu proses dan menunjukkan gambaran ruang lingkup sistem secara keseluruhan. Diagram konteks merupakan diagram tertinggi dan sering disebut diagram level 0 dalam alir data. Diagram ini berisi beberapa entitas yang berhubungan ke dalam sistem. Entitas adalah sesuatu yang memiliki keberadaan yang unik dan berbeda, walaupun tidak harus dalam bentuk fisik. Abstraksi, misalnya, biasanya dianggap juga sebagai suatu entitas. Dalam pengembangan sistem, entitas digunakan sebagai model yang menggambarkan komunikasi dan pemrosesan internal. Notasi dalam diagram konteks dapat dilihat pada Tabel 2.1.

Tabel 2.1 Notasi Dalam Diagram Konteks

Notasi	Arti
	Entitas
	Proses
	Aliran data atau informasi

Contoh diagram konteks :

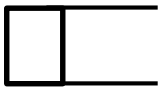
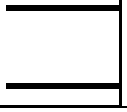


C. DFD/DAD (*Data Flow Diagram*/ Diagram Alir Data)

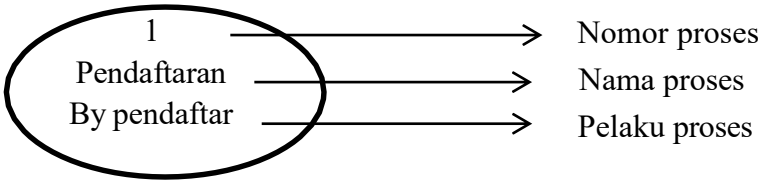
Didalam kasus studi alir data kita akan sering mendengar istilah DFD atau DAD. DFD dan DAD merupakan hal yang sama hanya saja penyebutan yang menggunakan bahasa indonesia dan bahasa inggris. DFD merupakan singkatan dari *data flow diagram* sedangkan DAD merupakan singkatan dari diagram alir data.

DFD merupakan diagram yang digunakan untuk menggambarkan proses-proses yang terjadi pada sistem yang akan dikembangkan. Dengan model ini, data-data yag terlibat pada masing-masing proses dapat diidentifikasi. Dengan kata lain DFD merupakan diagram lanjut dari diagram konteks, dimulai dari level 1 hingga level-level selanjutnya. DFD berbentuk level-level di mana setiap level dijabarkan dari setiap proses didalam DFD level 1. Jumlah level dalam DFD ditentukan dari proses bisnis atau daftar kejadian yang dijelaskan pada DFD level di atasnya. Notasi dalam DFD dapat dilihat pada tabel 2.2.

Tabel 2.2 Notasi Dalam DFD

Notasi	Arti
	Entitas
	Proses
	Aliran data atau informasi
 atau 	Tempat penyimpanan data

Proses terdiri dari nomor proses, nama proses dan pelaku proses. Khusus pelaku proses bersifat opsional contoh :

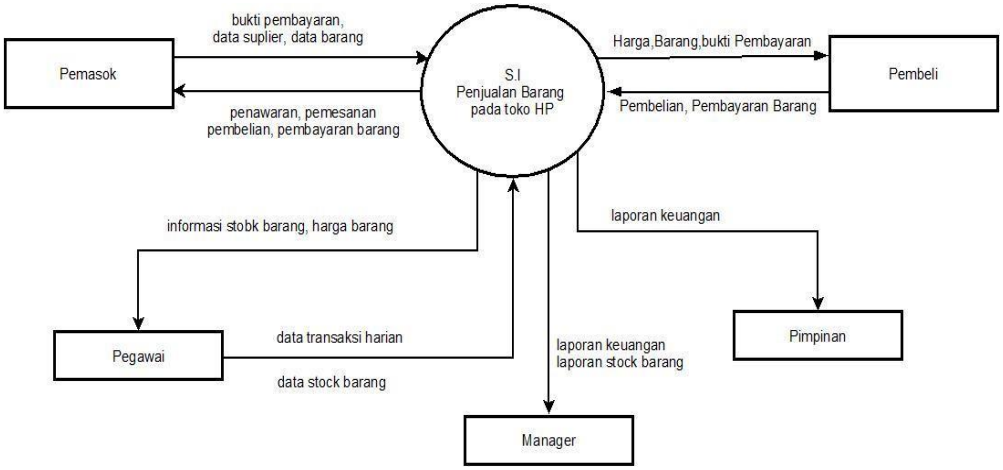


Aliran data atau informasi dapat berupa formulir input dan formulir output. Terminator atau entitas merupakan sumber atau tujuan data, terminator dapat berupa orang, organisasi ataupun bagian/departemen. Sedangkan simbol tempat penyimpanan dapat berupa arsip manual ataupun arsip elektronik (database, tabel, file).

Contoh kasus :

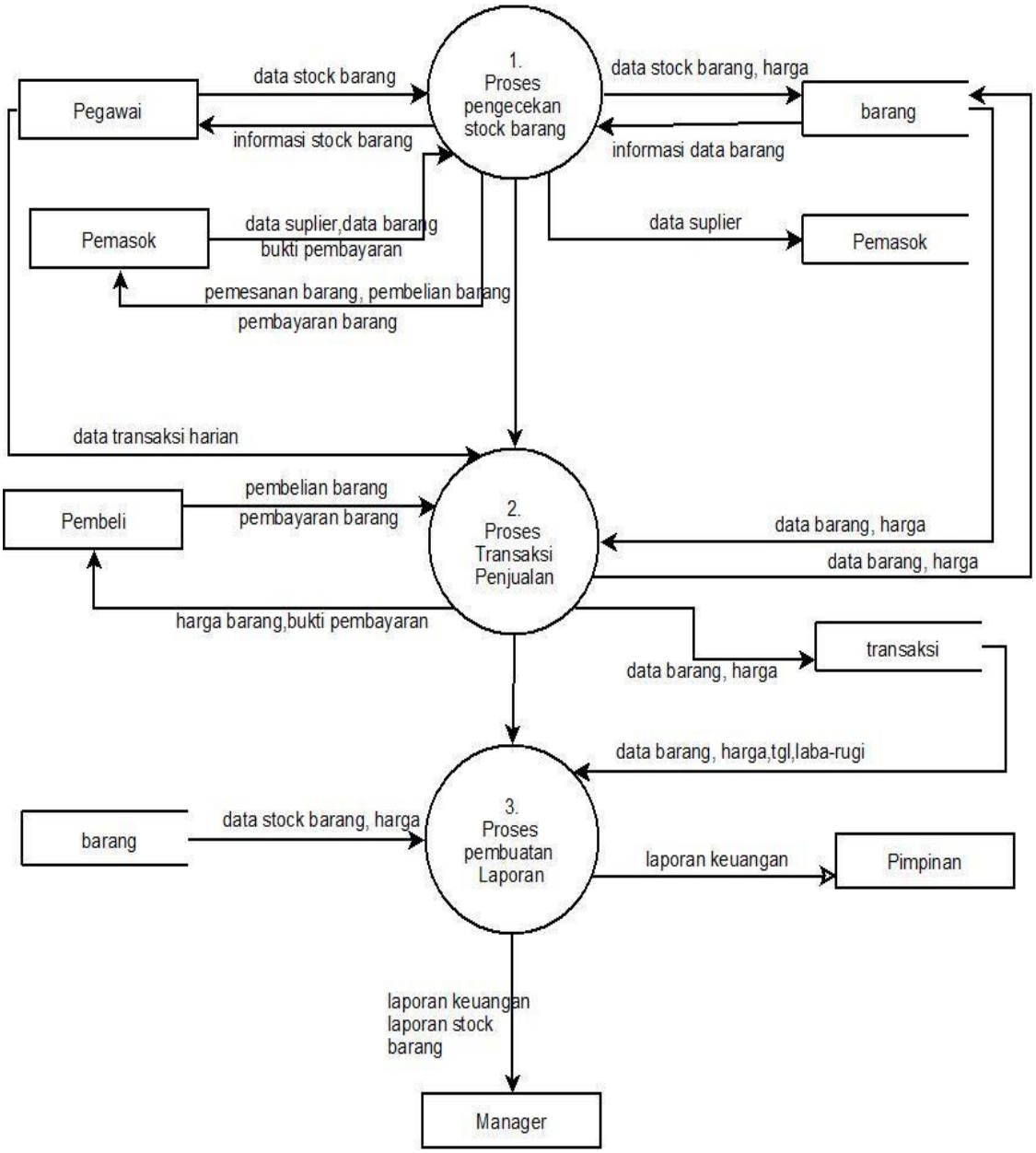
Diagram Penjualan Pada Toko Handphone

1. Diagram Konteks



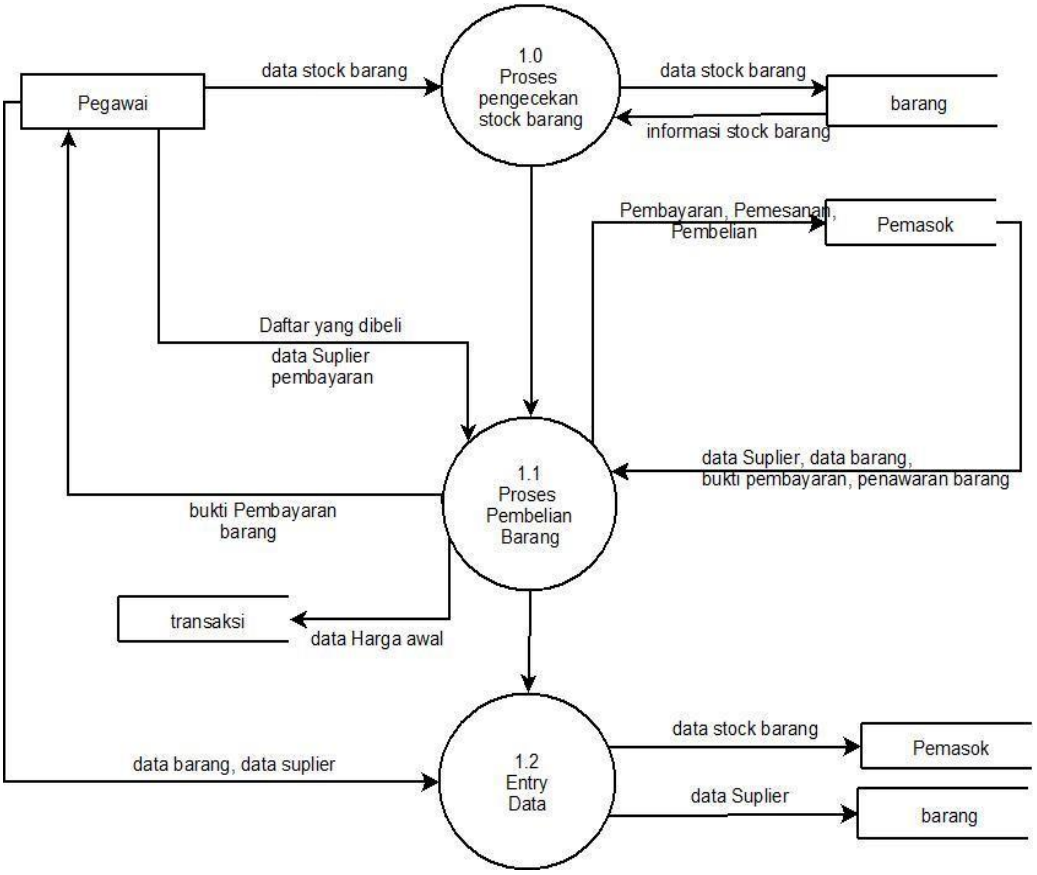
Pada diagram ini dapat dilihat bahwa sistem penjualan toko handphone ini melibatkan lima *external entity* yaitu pemasok, pegawai, manager, pimpinan dan pembeli. Penjelasan secara singkat dari diagram di atas adalah pemasok akan menawarkan barang pada sistem, lalu sistem akan memberikan daftar barang apa saja yang akan dibeli, kemudian pembeli akan membeli barang. Semua transaksi yang dilakukan baik antara pemasok ataupun pembeli akan dicatat oleh pegawai. Selanjutnya, sistem akan memberikan laporan keuangan dan stock barang kepada manager sedangkan laporan yang diberikan pada pimpinan hanyalah laporan keuangan.

2. *Data Flow Diagram* level 1



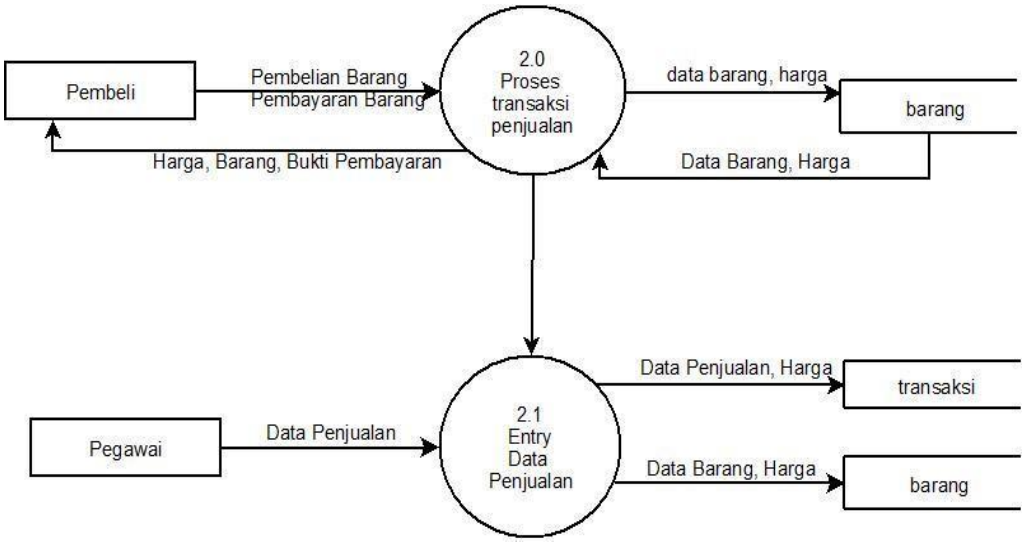
Pada tahap ini terdapat tiga proses utama yang dilakukan oleh sistem, yaitu proses pengecekan barang, proses transaksi penjualan, dan proses pembuatan laporan.

3. *Data Flow Diagram* level 2 proses 1(pengecekkan stock barang)



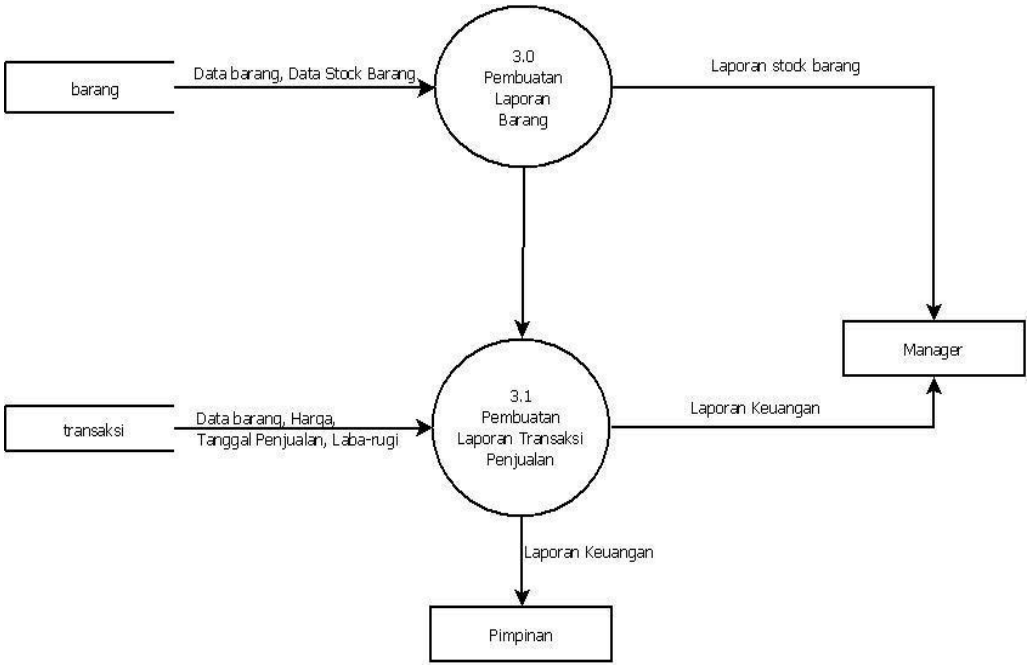
Tahap ini adalah penjabaran dari proses pertama pada DFD. Sistem akan melakukan pemeriksaan barang terlebih dahulu. Apakah setiap barang yang ada kurang dari sepuluh, karena di sini ditetapkan bahwa setiap barang yang jumlah stock-nya kurang dari sepuluh akan dipesan ke pemasok, tapi jika tidak, maka pemesanan tidak akan dilakukan. Pemesanan juga dapat dilakukan jika terdapat barang baru. Pengecekan dilakukan dengan pegawai akan memasukkan data stock barang pada sistem dan data akan dimasukkan ke dalam data store barang. Selanjutnya adalah pembelian barang kepada pemasok. Di sini pegawai kembali berperan, pegawai akan memasukkan daftar pembelian, data supplier dan pembayaran. Semua transaksi disini akan dimasukkan ke dalam data store transaksi. Setelah itu terdapat proses entry data. Entry data disini dimaksudkan untuk memasukkan data-data pembelian yang akan disimpan ke dalam data store supplier dan data store barang. Sehingga data akan mudah untuk di-*update*.

4. *Data Flow Diagram* level 2 proses 2 (transaksi penjualan)



Pada tahap ini dapat dilihat bahwa pembeli akan melakukan pembelian barang, kemudian sistem akan memberikan harga barang, barang dan bukti pembayarannya. Kemudian sistem akan menyimpan data tersebut ke dalam data store barang, agar pada data store barang di-*update* lagi perubahannya. Lalu dilakukan peng-input-an data penjualan ke dalam data store transaksi dan data store barang.

5. *Data Flow Diagram* Level 2 proses 3 (pencatatan data)



Pada proses ini, pembuatan laporan barang akan dilakukan. Sistem akan mengambil data barang dan data stock barang pada data store barang kemudian akan mengolahnya dan laporan jumlah stock barangnya akan diberikan kepada manager, selanjutnya adalah pembuatan laporan transaksi penjualan. Data-data untuk pembuatan laporan transaksi penjualan barang akan diambil dari data store transaksi kemudian akan diolah dan akan menghasilkan laporan keuangan, yang akan diberikan kepada manager dan pimpinan.

D. Latihan

Sistem KRS di UPTT dilakukan untuk memulai proses perkuliahan setiap semester. Terdapat proses pembayaran, perwalian, dan KRS di dalam sistem tersebut. Proses pembayaran dilakukan mahasiswa dengan membayarkan uang kuliah melalui bank. Data dari bank akan diberikan kepada UPTT dan disimpan ke database UPTT, hasil pengolahan data dilaporkan kepada manajer keuangan. Ketika dalam proses pembayaran mahasiswa melakukan proses perwalian. Mahasiswa memberikan data nilai semester sebelumnya dan data

mata kuliah yang akan diambil semester ini kepada dosen Pembimbing Akademik (PA). Dosen PA akan mengecek data tersebut sesuai IP dan daftar mata kuliahnya. Setelah pengecekan mahasiswa mendapatkan bukti acc dari dosen PA.

Setelah perwalian dilakukan mahasiswa, dapat melakukan proses KRS dan

memasukkan data mata kuliah yang ingin diambilnya. Setelah memasukkan semua mahasiswa submit data tersebut ke dalam sistem. Mahasiswa mendapatkan output berupa kartu KRS yang ditampilkan di layar. Data yang di-submit semua mahasiswa kemudian akan disimpan pada database BAA. Rekap data keseluruhan mahasiswa akan diterima manajer BAA dan diberikan pula kepada kaprodi berupa output di tampilan.

Berdasarkan kasus tersebut buatlah :

1. Diagram konteks.
2. DFD level 1 untuk proses pembayaran dan proses KRS.

MODUL III
SPESIFIKASI PROSES DAN KAMUS DATA

A. Tujuan Praktikum:

1. Mahasiswa mampu memahami langkah dalam pembuatan spesifikasi proses dan kamus data.
2. Mahasiswa mampu membuat spesifikasi proses dan kamus data.

B. Dasar Teori

1. Spesifikasi Proses

Pada bab ini, akan menjelaskan mengenai spesifikasi proses. Kegunaan spesifikasi proses cukup penting untuk ke depannya, hal ini mendefinisi apa yang harus dikerjakan untuk merubah input menjadi output. Hal tersebut merupakan gambaran detail kebijakan bisnis yang dibawa oleh setiap lingkaran (*entitas*).

Spesifikasi proses mendefinisikan kegiatan yang harus dilakukan untuk mengubah input menjadi output (*Yourdon, 1989*). Sepsifikasi proses digunakan untuk mendeskripsikan proses yang terjadi pada level yang paling dasar dalam DFD. Model ini berfungsi mendeksripsikan apa yang dilakukan ketika masukan ditransformasi menjadi keluaran.

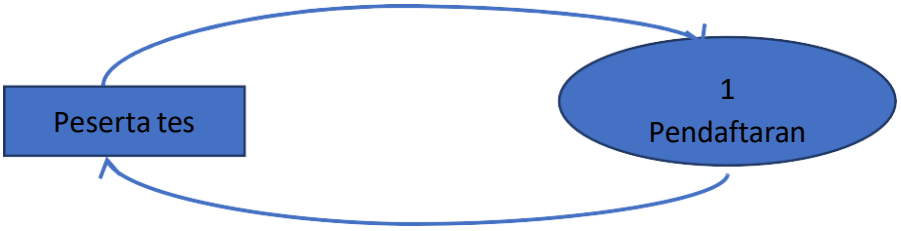
Ada berbagai macam *tools* yang dapat kita gunakan untuk menghasilkan suatu spesifikasi proses: tabel keputusan, Bahasa Inggris terstruktur, pre/post condition, flowcharts, diagram Nassishneiderman,dan lain sebagainya. Bentuk penyajian spesifikasi proses dapat dilihat pada Tabel 3.1.

Tabel 3.1 Spesifikasi Proses

No Proses : menyatakan nomor proses **Nama Proses : menyatakan nama proses** **Deskripsi : penjelasan tujuan proses**	
Source	**Data**
(menyatakan sumber data input menuju proses)	(menyatakan isi data yang masuk ke proses)
Destination	**Data**
(menyatakan tujuan data output dari proses)	(menyatakan isi data yang keluar dari proses)
Logika proses (menyatakan algoritma dari proses)	

a. Contoh proses spesifikasi :

contoh kasus pada proses pendaftaran ujian tes.



Maka proses spesikasinya adalah :

No Proses	: 1
Nama Proses	: pendaftaran
Keterangan	: Pendaftaran Peserta Test
Input	: data peserta test
Output	: kartu peserta test
Proses/logika proses	:
• Terima berkas pendaftaran	
• Periksa kelengkapan	
• Scan formulir	
• Cetak kartu test	

Gambar 3.2 contoh proses spesifikasi

b. Konstruksi atau logika yang digunakan dalam proses spesifikasi

If-Then-Else

Digunakan untuk mendeskripsikan pilihan secara biner (pada satu saat hanya ada dua pilihan), bentuknya:

Terima berkas Periksa kelengkapan

Repeat

Scan formulir

If tidak ada kesalahan form **Then**

Cetak kartu tes

Else

periksa form

Endif

until tidak ada kesalahan

2. Kamus data

Model berikutnya yang akan dibahas adalah *data dictionary*/DD (Kamus Data/KD). KD tidak menggunakan notasi grafis sebagaimana halnya DFD, tetapi porsinya dalam memodelkan sistem tidak perlu diragukan lagi (sebuah model tidak lengkap tanpa KD). KD juga mempunyai fungsi yang sama dalam pemodelan sistem. Selain itu KD berfungsi membantu pelaku sistem untuk mengerti aplikasi secara detil, kamus data mereorganisasi semua elemen data yang digunakan dalam sistem dengan presisi yang sedemikan rupa sehingga pemakai dan penganalisas sistem memiliki dasar pengertian yang sama tentang masukan, keluaran, penyimpanan dan proses. Kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Kamus data selain digunakan untuk dokumentasi dan mengurangi redudansi.

a. Elemen – elemen data

1) **Nama arus data**, karena kamus data dibuat berdasarkan arus data yang mengalir di DFD, maka nama dari arus data juga harus dicatat di KD.
2) **Alias**, alias atau nama lain dari data dapat dituliskan bila nama lain ini ada. Alias perlu ditulis karena data yang sama mempunyai nama yang berbeda untuk orang atau departemen satu dengan yang lainnya.
3) **Bentuk data**, yang telah diketahui bahwa data dapat mengalir. Dengan demikian bentuk dari data yang mengalir dapat berupa: dokumen dasar atau formulir, dokumen hasil cetakan komputer, laporan tercetak, tampilan di layar monitor, variabel, parameter, field.
4) **Arus data**, arus data menunjukkan dari mana data mengalir dan ke mana data akan menuju. Keterangan ini perlu dicatat di KD agar mudah mencari arus data di DAD.
5) **Penjelasan,** Untuk lebih memperjelas lagi tentang makna dari arus data yang dicatat di KD, maka bagian penjelasan dapat diisi dengan keterangan-keterangan tentang arus data tersebut.
6) **Struktur data**, struktur data menunjukkan arus data yang dicatat di KD terdiri dari item item data apa saja.

Tabel 3.2 Contoh Kamus Data

KAMUS DATA

Nama arus data : Tembusan permintan persedian
Alias : Faktur
Tembusan jurnal
Tembusan kredit
Bentuk data : Dokumen cetakan komputer
Arus data : Proses 1.4 - Gudang
Proses 1.4 - Bagian pengiriman

Penjelasan : tembusan dari faktur penjualan untuk meminta barang dari gudang
Periode : Setiap kali terjadi penjualan (harian)
Volume : Volume rata-rata adalah 100 dan volume puncak adalah 150
Struktur data : Tembusan permintaan persediaan terdiri dari item data :

Kode langganan
Nama langganan
Tanggal penjualan
Nomor faktur
Satu sampai dengan maksimum 5 kali
Kode barang
Nama barang
Unit jual
Total harga
Total penjualan
Potongan penjualan
Pajak penjualan
Total dibayar
Jenis penjualan

b. Menggambarkan struktur data (kamus data komposit)

Kebanyakan sistem dalam dunia nyata (di mana kita bekerja), kadang-kadang ditemui elemen data yang terlalu kompleks untuk didefinisikan. Kekompleksan tersebut seharusnya diuraikan melalalui sejumah elemen data yang lebih sederhana. Kemudian elemen data yang lebih sederhana tersebut didefinisikan kembali hingga nilai dan satuan yang relevan (yang sifatnya elementer). Pendefinisian tersebut menggunakan notasi yang umumnya digunakan dalam menganalisas sistem dengan menggunakan sejumlah simbol pada Tabel 3.3.

Tabel 3.3 Simbol pada Kamus Data

No	Simbol	Uraian
1	=	Terdiri dari, mendefisikan, diuraikan menjadi
2	+	Dan
3	( )	Menunjukan suatu elemen yang bersifat pilihn. Elemen-elemen yang bersifat pilihan ini dikosongkan pada layar masukan atau bisa juga dengan membuat spasi.
4	{ }	Menunjukan elemen-elemen repetitive, juga disebut kelompok berulang atau tabel-tabel. Kemungkinan bisa ada satu atau berapa elemen berulang didalam kelompok tersebut. Kelompok brulang bisa mengundang keadaan-keadaan tertentu, seperti misalnya, jumlah pengulangan yang pasti atau batas tertinggi dan batas terendah untuk jumlah pengulangan.
5	[ ]	Menunjukkan salah satu dari 2 situasi tertentu. Satu elemen bisa ada sedangkan lainnya juga ada, tetapi tidak bisa kedua-duanya secara bersamaan. Elemen-elemen yang ada di dalam tanda kurung ini saling terpisah satu sama lain.
6	|	Pemisah sejumlah alternatif pilihan antara si-mbol
7	@	Identifikasi atribut kunci
8	**	komentar

C. Latihan

Menggunakan sistem KRS seperti pada modul I. Pada proses KRS mahasiswa memasukkan mata kuliah yang dipilih dan di-submit ke sistem. Jumlah SKS yang diambil mahasiswa sesuai dengan IP semester sebelumnya. Ketentuan pengambilan sks adalah sebagai berikut:

- IP ≥ 3,00 dapat mengambil 24 SKS
- IP 2,50 sampai 2,99 dapat mengambil 22 SKS
- IP 2,00 sampai 2,49 dapat mengambil 20 SKS
- IP 1,50 sampai 1,99 dapat mengambil 18 SKS
- IP < 1,50 dapat mengambil 15 SKS

Data yang dibutuhkan untuk KRS adalah data nilai mata kuliah semester lalu dan data IP. Buatlah proses spesifikasinya dan kamus data sesuai dengan kasus di atas!

MODUL IV
PEMODELAN DATA

A. Tujuan Praktikum

1. Mahasiswa mampu merancang ERD.

B. Dasar Teori

1. *Entity Relationship Diagram* (ERD)

ERD (*Entity Relationship Diagram*) adalah model data yang pembuatannya didasarkan pada anggapan bahwa dunia nyata terdiri dari kumpulan objek dasar yang disebut entitas dan relasi diantaranya. Entitas adalah sebuah objek yang bisa dibedakan dari objek lainnya berdasarkan sekumpulan atribut yang spesifik. ERDmerupakan notasi grafis dalam pemodelan data konseptual yang mendeskripsikan hubungan antara penyimpanan. ERD digunakan untuk memodelkan struktur data dan hubungan antar data,karena hal ini relatif kompleks. Notasi-notasi simbolik yang digunakan dalam diagram E-R adalah sebagai berikut :

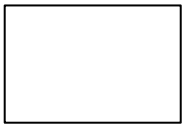
1. Persegi Panjang menyatakan himpunan entitas.
2. Lingkaran/Elips menyatakan atirbut (atribut yang berfungsi sebagai *key* digaris bawahi).
3. Belah Ketupat/Jajar Genjang menyatakan himpunan relasi.
4. *Link*/Garis sebagai penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dan atributnya.

Kardinalitas relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka 1 ke 1 untuk relasi satu-ke-satu, dan N untuk relasi satu-ke-banyak atau N ke N untuk relasi banyak-ke-banyak.

C. Notasi Diagram E-R

1. Entitas

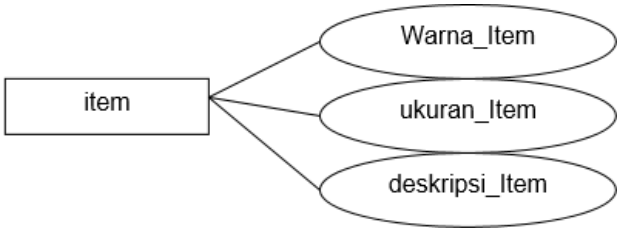
Suatu objek yang dapat diidentifikasi dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat. Entitas digambarkan dengan bentuk persegi panjang. Gambar dapat dilihat pada gambar 4.1 berikut.



Gambar 4.1 Notasi Entitas

2. Atribut

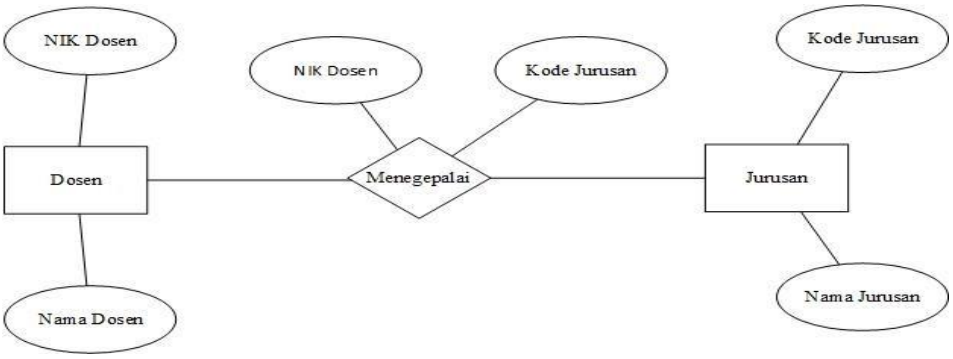
Entitas mempunyai elemen yang disebut atribut dan berfungsi mendeskripsikan karakter entitas. Misalnya atribut nama pakaian dan entitaspakaian. Setiap ERD bisa terdapat lebih dari satu atribut. Atribut digambarkan dalam bentuk elips. Contoh dapat dilihat pada gambar 4.2 berikut.



Gambar 4.2 Contoh Atribut

3. Hubungan

Menunjukan adanya hubungan atau relasi diantara sejumlah entitas berasal dari himpunan entitas yang berbeda. Sebagaimana halnya entitas maka berhubungan harus dibedakan antara hubungan atau bentuk hubungan antar entitas dengan isi dari hubungan itu sendiri. Misalnya dalam kasus hubungan antara entitasdosen dan entitasjurusan adalah mengepalai sedangkan isi hubunganya dapat berupa kode dosen dan kode jurusan. Hubungan digambarkan dalam bentuk Belah Ketupat/Jajar Genjang. Contohdapat dilihat pada gambar 4.3 berikut.

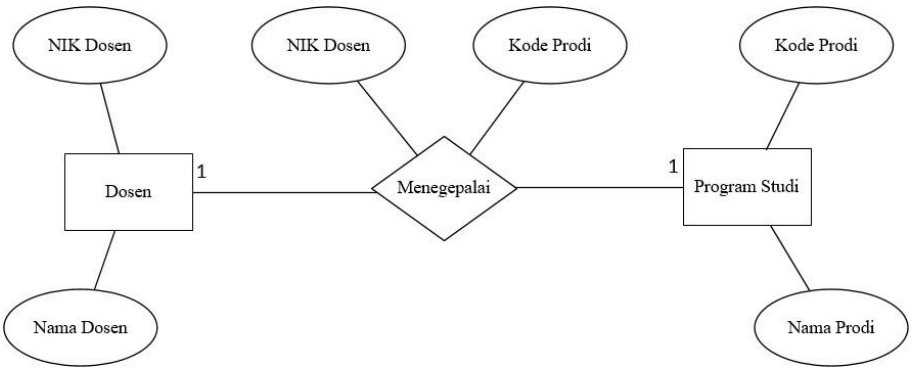


Gambar 4.3 Hubungan Antar Relasi

D. Penggambaran Relasi

1. Relasi satu-ke-satu (*one-to-one*)

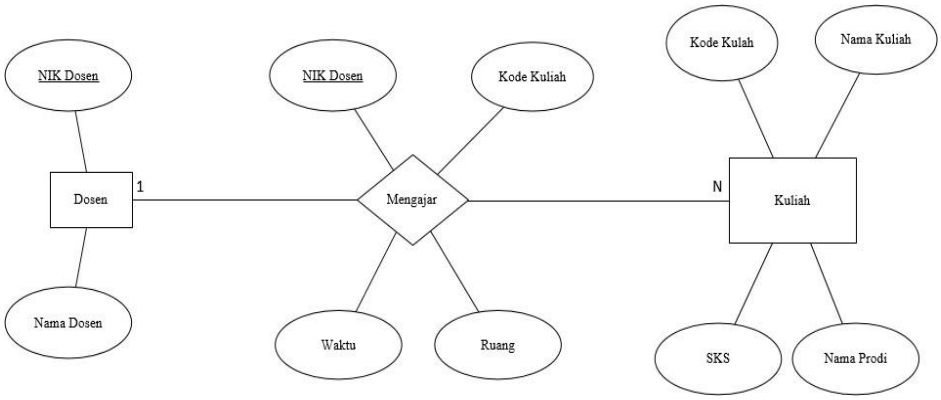
Relasi antar hubungan antara entitas dimana himpunan hanya memiliki paling banyak 1 hubungan. Contoh dari relasi ini adalah adanya relasi antara himpunan entitas dosen dengan himpunan entitas program studi. Himpunan relasinya diberi nama “mengepalai”. Pada relasi ini setiap jurusan dikepalai oleh satu dosen. Maka penggambaranya dapat dilihat pada gambar 4.4 berikut :



Gambar 4.4 Contoh Relasi *One-to-one*

2. Relasi satu-ke-banyak (*one-to-many*)

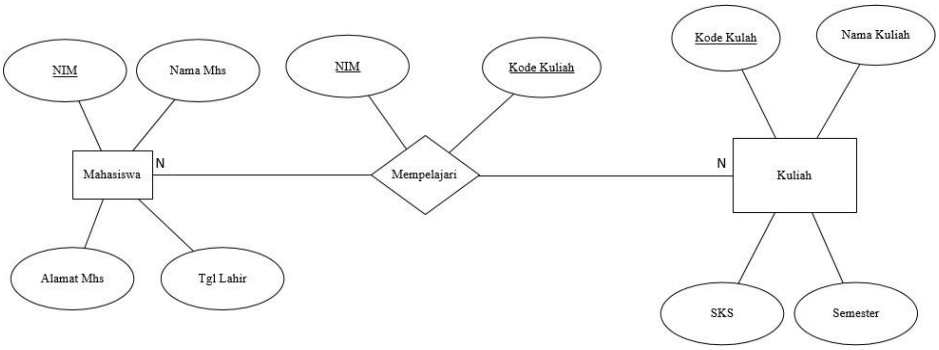
Relasi antar hubungan antara entitas misalnya himpunan A memiliki hubungan ke himpunan B lebih dari satu. Contoh adanya relasi antara himpunan entitas dosen dengan himpunan entitas kuliah. Himpunan relasinya diberi nama "mengajar".Pada relasi ini setiap dosen mengajar lebih dari satu mata kuliah. Maka penggambarannya dapat dilihat pada gambar 4.5 berikut :



Gambar 4.5 Relasi *One-to-many*

3. Relasi Banyak-ke-banyak (*many-to-many*)

Relasi antar entitas dimana setiap himpunan memiliki lebih dari satu atau beberapa hubungan. Contohnya adalah adanya relasi antara himpunan entitas mahasiswa dengan himpunan entitas kuliah. Himpunan relasinya diberi nama "mempelajari". Pada relasi ini setiap mahasiswa dapat mempelajari lebih dari satu mata kuliah. Demikian sebaliknya, setiap mata kuliah dapat dipelajari oleh lebih dari satu orang mahasiswa. Maka penggambaranya dapat dilihat pada gambar 4.6 berikut :



Gambar 4.6 Relasi *Many-to-many*

E. Tahapan Pembuatan Diagram E-R

Diagram E-R dibuat secara bertahap. Paling tidak ada dua kelompok tahapan yang bisa ditempuh dalam pembuatan diagram E-R, yaitu :

1. Tahap pembuatan diagram E-R awal (*Preliminary Design*)

   Untuk mendapatkan rancangan basis data minimal yang dapat mengakomodasi kebutuhan hubungan penyimpanan data terhadap sistem yang akan dibangun. Pada umumnya mengabaikan adanya penyimpangan-penyimpangan

2. Tahap optimasi diagram E-R (*Final Design*)

   Dilakukan koreksi terhadap hasil tahap awal dengan memperhatikan aspek efisiensi, performasi dan fleksibilitas. Bentuk koreksi yang dilakukak yaitu :

   a. Dekomposisi himpunan entitas
   b. Penggabungan himpunan entitas
   c. Pengubahan derajat relasi
   d. Penambahan relasi baru
   e. Penambahan dan pengurangan atribut untuk masing-masing entitas dan relasi.

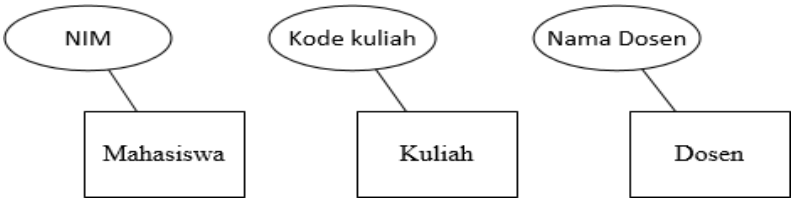
   Langkah-langkah dalam menyusun diagram E-R yaitu :

   1. Mengidentifikasi dan menetapkan seluruh himpunan entitas yang terlibat.
   2. Menentukan atribut-atribut *key* dari masing-masing himpunan entitas.
   3. Mengidentifikasi dan menetapkan seluruh himpunan relasi di antara himpunan entitas-himpunan entitas yang ada beserta *foreign key*-nya.
   4. Menentukan derajat/kardinalitas relasi dan himpunan relasi dan atribut-atribut deskriptif (non *key*).

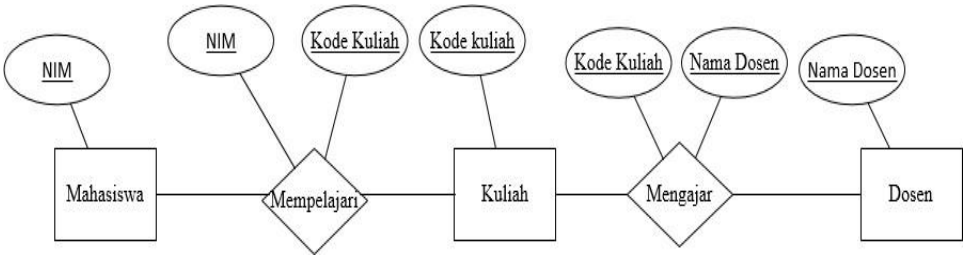
   Contoh kasus pada perkuliahan langkah-langkah teknis pada mewujudkan perancangan basis data pada kasus perkuliahan adalah sebagai berikut:

1. Mengidentifikasi dan menetapkan seluruh himpunan entitas yang akan terlibat.

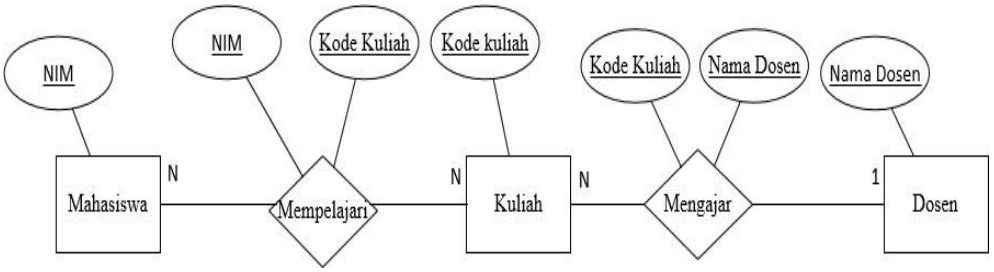
2. Menenrukan atribut-atribut *key* dari masing-masing himpunan entitas.



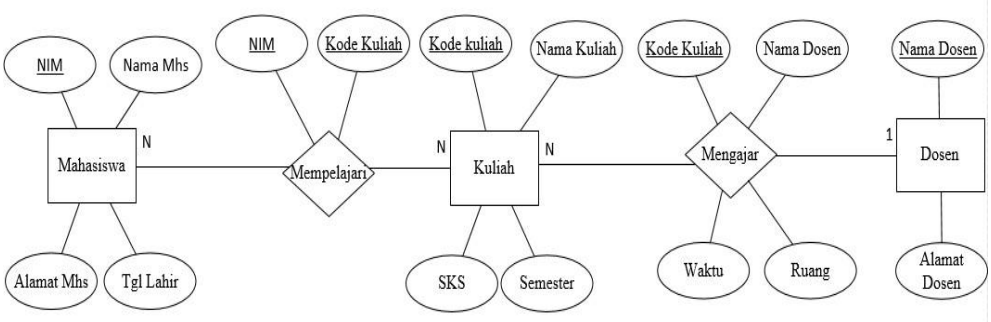
3. Mengidentifikasi dan menetapkan seluruh himpunan himpunan relasi diantara entitas-himpunan entitas yang adda beserta *foreign-key* nya.



4. Menentukan derajat/kardinasi relasi untuk setiap himpunan relasi.



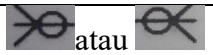
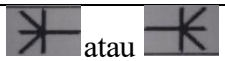
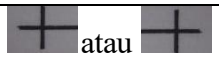
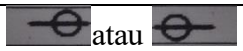
5. Melengkapi himpunan Entitas dan himpunan relasi dengan atribut-atribut deskriptif (*non key*).



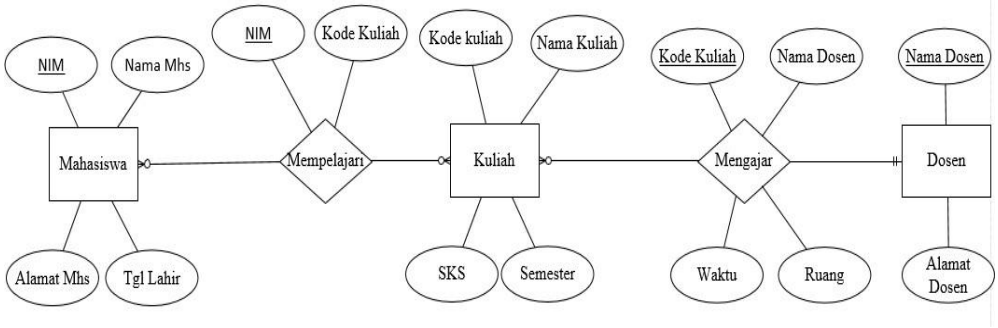
F. Diagram E-R dengan Notasi Lain

Pada notasi ini diagram E-R yang digunakan berbeda. Contohnya dapat di lihat pada tabel 4.1 berikut.

Tabel 4.1 Notasi Lain Diagram E-R

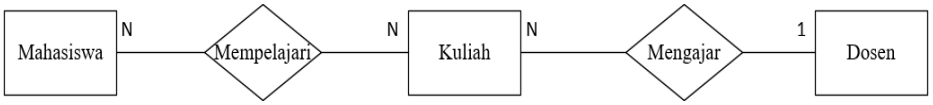
Notasi	Derajat Relasi Minimum-Maksimum
 atau	(0,N)
 atau	(1,N)
 atau	(1,1)
 atau	(0,1)

Contoh :



G. Diagram E-R dengan Kamus Data

Objektif utama dari pembuatan diagram E-R adalah untuk menunjukan objek-objek (himpunan entitas) apa saja yang ingin dilibatkan dalam sebuah basis data dan bagaimana hubungan yang terjadi di antara objek-objek tersebut. Pada sebuah sistem yang ruang lingkupnya lebar dan kompleks, penggambaran atribut-atribut dalam sebuah diagram E-R seringkali malah mengganggu obyektif yang ingin dicapai tersebut. Hal tersebutdapat dipisahkandenganmendeklarasikan atribut-atribut ini dari diagram E-R dan menyatakannya dalam sebuah kamus data. Kamus data berisi daftar atribut yang di jepit kurung kurawal "{ dan }". Atribut yang berfungsi sebagai *key* juga dibedakan dengan yang bukan *key* dengan menggarisbawahi atribut tersbut. Karena itulah diperbolehkan menggambarkan Diagram E-R dengan tambahan kamus data seperti pada gambar 4.7 berikut ini.



Gambar 4.7 Diagram E-R dengan Kamus Data

Kamus Data :

Mahasiswa	: {NIM, Nama Mahasiswa, Alamat Mhs, Tgl Lahir}
Kuliah	: {Kode Kuliah, Nama Kuliah, SKS, Semester}
Dosen	: {Nama Dosen, Alamat Dosen}
Mempelajari	: {NIM, Kode Kuliah}
Mengajar	: {Kode Kuliah, Nama Dosen, Waktu, Tempat}

H. Latihan

Buatlah sebuah perancangan diagram ERD dengan menggunakan notasi diagram dengan tema mengenai masalah perkuliahan dari hubungan antara atribut mahasiswa, dosen, Prodi, TU, Mata kuliah dan sampai ke nilai kuliah. Atribut bisa ditambahkan sesuai dengan kebutuhan. Kerjakan sesuai dengan pemahaman dan pandangan anda masing-masing.

MODUL V

NORMALISASI DATA

A. Tujuan Praktikum

1. Menciptakan suatu relasi database yang terstruktur
2. Memghilangkan kerangkapan data
3. Mengurangi kompleksitas
4. Mempermudah modifikasi data

B. Dasar Teori

1. Mengenal Normalisasi Data

Normalisasi adalah teknik yang menstrukturkan atau memecah atau mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan (*anomallies*) yang terjadi akibat adanya kerangkapan data dalam relasi dan inefisiensi pengolahan. Dalam uraiaan tentang normalisasi basis data kita akan banyak menggunakan istilah-istilah baru seperti **Atribut, Key, Domain,** dan **Ketergantungan Fungsional.**

a. Atribut

Setiap entitas pasti memiliki atribut yang mendeskripsikan karakteristik dari entitas tersebut.

Contoh : entitas pelanggan

Atributnya **kd_pelanggan**, nm_pelanggan, alamat, no_telepon. Dari atribut di samping yang menjadi *primary key* adalah kd_pelanggan karena bersifak **unik.** *Primary key* adalah salah satu dari *candidate key* yang terpilih. Contoh *primary key* adalan **NIM.**

Alasan pemilihan *primary key* :

1) lebih sering dijadikan acuan
2) lebih ringkas
3) jaminan keunikan *key* lebih baik

b. Key

Key adalah satu atau gabungan dari beberapa atribut yang dapat membedakan semua baris data *(row)* dalam tabel secara unik. Artinya jika suatu atribut dijadikan sebagai *key*, maka tidak boleh ada dua atau lebih baris

data dengan nilai yang sama untuk atribut tersebut. Ada tiga macam *key* yang dapat diterapkan pada suatu tabel, yaitu :

1) *Superkey*

Superkey merupakan suatu atau lebih atribut yang dapat membedakan setiap baris data dalam sebuah tabel secara unik. Contoh di tabel mahasiswa yang terdiri dari atribut-atribut seperti NIM, nama_mahasiswa, tgl_lahir, maka yang dapat menjadi super key adalah :

- (nim, nama_mhs, tgl_lahir)
- (nim, nama_mhs)
- (nim, tgl_lahir)
- (nama_mahasiswa)
- (nim)

2) *Candidate key*

Merupakan kumpulan atribut **minimal** yang dapat membedakan setiap baris data dalam sebuah tabel secara unik. *Candidate key* tidak boleh berisi atribut atau kumpulan atribut yang telah menjadi *superkey* yang lain. Jadi, sebuah *candidate key* pastilah *superkey*, tapi belum tentu sebaliknya. Pada tabel mahasiswa yang dapat menjadi *candidate key* adalah:

- (nim)
- (nama_mhs)

3) *Key primer*

Sebuah tabel dimungkinkan adanya lebih dari satu *candidate key* seperti contoh di atas. Salah satu *candidate key* ini (jika memang ada lebih dari satu) dapat dijadikan sebagai *key primer*. Pemilihan *key primer* umumnya didasari oleh:

- *Key* tersebut lebih sering dijadikan sebagai acuan
- *Key* tersebut lebih ringkas
- Jaminan keunikan *key* tersebut lebih baik

Maka dari contoh di atas yang lebih cocok dipilih sebagai *key primer* adalah (nim)

c. Domain

Pada saat pekerjaan merancang basis data dilakukan, yang perlu dilihat dan dipertimbangkan hanyalah domain nilai dari setiap atribut. Penetapan tipe data bagi suatu atribut baru akan relevan dan penting diperhitungkan pada saat implementasi basis data.

d. Ketergantungan Fungsional (*functional dependency)*

Functional dependency merupakan ketergantungan suatu atribut terhadap atribut lain

- a → b

  setiap nilai a hanya mempunyai satu nilai b

  atau nilai b bergantung kepada nilai a

  Contoh: nim → nama

2. Normalisasi dengan Ketergantungan Fungsional

a. *Partial Functional Dependency*

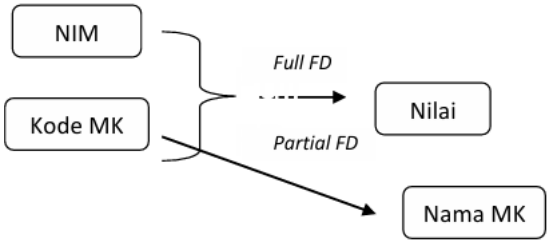
Ketergantungan parsial (sebagian) adalah ketergantungan suatu atribut hanya pada sebagian kunci di sebuah tabel.

b. *Full Functional Dependency*

ketergantungan fungsional penuh Merupakan ketergantungan penuh suatu atribut terhadap atribut kunci.

Contoh Partial FD dan Full FD

NIM	Kode_MK	Nama_MK	Nilai
001	MK01	APSI	B
002	MK02	Bahasa Inggris	C

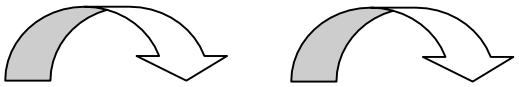


c. *Transitive Dependency* (ketergantungan transitif)

Ketergantungan tidak langsung suatu atribut terhadap atribut lainnya. Ilustrasi:

a → b → c

Contoh:

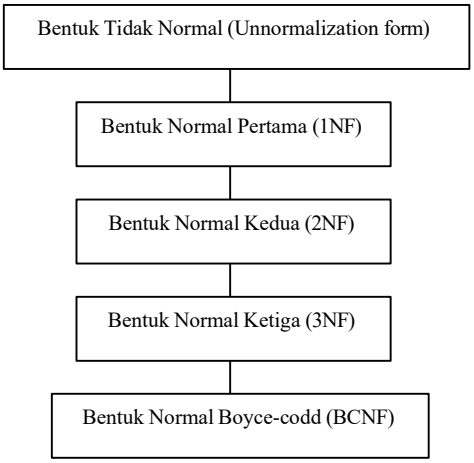


Kode dosen	Kode PT	Nama PT

{kode_dosen → kode_PT → nama_PT}

3. Bentuk-bentuk dalam normalisasi

Tahapan Normalisasi



a. Bentuk tidak normal

Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja tidak lengkap dan terduplikasi, data dikumpulkan apa adanya sesuai keadaanya. Data didapat dari bentuk dokumen yang ada. Bentuk data ini merupakan data mentah yang dikumpulkan dari beberapa sumber. Contoh :

Tabel Kirim 1 Un-Normal

No-pem	Kode-Kota	Kota	No-bar	Jumlah
P01	1	Jakarta	B01	1000
			B02	1500
			B03	2000
P02	3	Bandung	B03	1000
P03	2	Surabaya	B02	2000

b. Bentuk normal pertama (1st normal form)

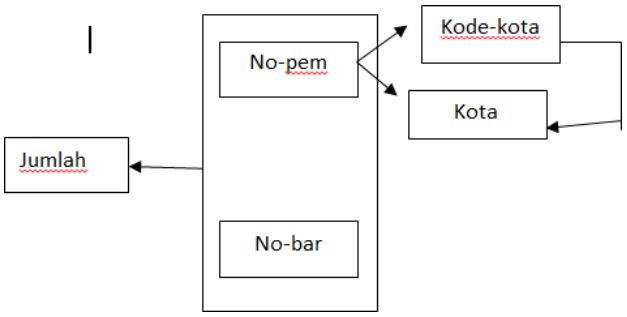
Data yang tidak normal akan dinormalkan dengan pertama menggunakan bentuknormal pertama. Syarat normal pertama:

- Atomik : satu file hanya memiliki satu nilai yang tidak dapat dipecah lagi.
- Tidak ada pengulangan baris pada tabel.

Tabel KIRIM-2 (1NF)

No-pem	Kode-Kota	Kota	No-bar	Jumlah
P01	1	Jakarta	B01	1000
P01	1	Jakarta	B02	1500
P01	1	Jakarta	B03	2000
P02	3	Bandung	B03	1000
P03	2	Surabaya	B02	2000

Karena tabel tidak atomic dan masih ada pengulangan data, maka dibuat diagram ketergantungan penuh, maka harus ada dekomposisi tabel.



c. Bentuk normal kedua (2nd normal form)

Suatu relasi dikatakan sudah memenuhi bentuk normal. Kedua bila relasi tersebut sudah memenuhi bentuk Normal kesatu, dan atribut yang bukan key sudah tergantung penuh terhadap *key*-nya. Syarat:

- Sudah 1st NF
- Tidak ada *partial functional dependency* (ketergantungan partial). Ketergantungan parsial (sebagian) di mana suatu atribut hanya bergantung pada sebagian kunci.

  Contoh: kolom (atribut) mulai dikelompokkan sesuai dengan *primary key*-nya (no_pem). Sebelum mengelompokkan tentukan

terlebih dahulu mana *primary key*. Atribut yang memiliki ketergantungan *primary key* dipecah menjadi tabel baru menjadi tabel pemasok.

Tabel PEMASOK-1 (2NF)

No-pem	Kode-Kota	Kota
P01	1	Jakarta
P02	3	Bandung
P03	2	Surabaya

d. Bentuk normal ke 3rd NF

Suatu relasi dikatakan sudah memenuhi bentuk normal ketiga bila relasi tersebut sudah memenuhi bentuk normal kedua dan atribut yang bukan key tidak tergantung transitif terhadap keynya. Syarat:

- Sudah 2nd NF
- Tidak ada *transitive functional dependency* (ketergantungan tidak langsung). Ketergantungan tidak langsung suatu atribut terhadap atribut lainnya.

Contoh : setiap atribut harus bergantung pada primary key (no_pem) secara menyeluruh. Hilangkan anomali–anomali yang masih mempunyai ketergantungan fungsional. Pada tabel kirim terdapat kolom kode_kota dan kota, perubahan kode_kota dan kota milik tabel kirim 2 dapat menyebabkan data tidak konsisten sekiranya hanya satu baris yang diubah sementara seharusnya ada beberapa baris.

Relasi tersebut juga terkena anomali penyisipan dan penghapusan. Atribut yang memiliki ketergantungan parsial akan dipecah kembali menjadi tabel baru sehingga memiliki ketergantungan penuh. Seperti pada tabel kirim 2, di dalamnya terdapat atribut kode_kota dan kota di mana atribute ini harus dipecah menjadi tabel baru karena dia bergantung dengan atribut lainnya, yakni no_pem. Dan no_bar. Kemudian, untuk kode_kota tidak mungkin.

Tabel KIRIM-3 (3NF)

No-pem	No-bar	Jumlah
P01	B01	1000
P01	B02	1500
P01	B03	2000
P02	B03	1000
P03	B02	2000

Pada tabel kirim 2 yang terdapat dalam normalisasi bentuk ketiga merupakan pemecahan dari tabel kirim 2 yang terdapat dalam normalisasi bentuk kedua, yang di mana atribut No-pem, No-bar, Jumlah memiliki nilai sendiri dalam setiap baris.

Tabel PEMASOK-2 (3NF)

No-pem	Kode-Kota
P01	1
P02	3
P03	2

Tabel PEMASOK-3 (3NF)

Kode-Kota	Kota
1	Jakarta
2	Surabaya
3	Bandung

Pada tabel pemasok 1 yang terdapat dalam normalisasi bentuk ketiga juga merupakan lanjutan pemecahan dari tabel pemasok 1 yang terdapat dalam normalisasi bentuk kedua, yang di mana atribut No_pem menjadi *Primary Key* atau kunci utama dan diikuti oleh kode_kota sebagai atribut yang mempunyai ketergantungan pada kunci utama. Sehingga data tersebut menjadi data yang sederhana sehingga *user* tidak kebingungan dalam penyusunan database yang tersedia dan mudah untuk dimengerti

4. Kriteria Minimal Normalisasi untuk Tabel

a. Jika ada dekomposisi (penguraian) tabel maka dekomposisinya harus dijamin aman.
b. Terpeliharanya ketergantungan fungsional pada saat perubahan.

c. Tidak melanggar *Boyce-Code Normal Form* (BCNF).

Jika BCNF tidak dapat terpenuhi maka diupayakan tidak melanggar bentuk 3rd NF.

5. Kelemahan

- Pengulangan informasi
- Tidak konsistennya data ketika diubah
- Tersembunyinya informasi tertentu

C. Latihan

I. Perhatikan tabel di bawah ini:

NIM	nama	alamat	Kode_MK	Nilai
001	Kiki	Jl Glagah Sari no 5	MK01	B
001	Kiki	Jl Glagah Sari no 5	MK02	A

1. Apakah tabel sudah baik atau belum?
2. Apa alasannya jika belum baik?
3. Bagian mana yang tidak baik?

II. Perhatikan tabel di bawah dengan kasus sebagai berikut:

- Seorang mahasiswa dapat mengambil beberapa mata kuliah
- Satu mata kuliah dapat diambil oleh lebih dari satu mahasiswa
- Satu mata kuliah hanya diajarkan oleh satu dosen
- Satu dosen dapat mengajar beberapa mata kuliah
- Seorang mahasiswa pada mata kuliah tertentu hanya mempunyai satu nilai
- Buat Tabel bentuk normal dari tahap1nf sampai 3rd NF

Tabel Mahasiswa1 -UNNORMAL

No-Mhs	Nama	Jurusan	Kode-MK	Nama-MK	Kode-Dosen	Nama-Dosen	Nilai
2683	Rifqa	TI	MI350	APSI	B104	Amalia	A
			MI465	Simulasi	B317	Utaminingsih	B
5432	Ayu	SI	MI350	APSI	B104	Amalia	C
			AKN201	Keuangan	D310	Endah	B
			MK300	Pemasaran	B212	Faishal	A

MODUL VI
BASIS DATA DALAM MySQL

A. Tujuan Praktikum

1. Mahasiswa mampu mengetahui dan memahami *Structures Query Language* (SQL)
2. Mahasiswa mampu mengetahui dan memahami struktur *database.*

B. Dasar Teori

1. Mengenal MySQL

   MySQL merupakan aplikasi berbasis *Database Management System* (DBMS) yang mampu menerima dan mengirimkan datanya sangat cepat yang menggunakan suatu bahasa perintah SQL. MySQL pertama kali dirilis oleh seorang programer database yang bernama Michael Widenius. terdapat beberapa kelebihan yang dimiliki MySQL di antaranya:

   a. MySQL sebagai *Relation Database Managemen System*.
   b. MySQL merupakan database yang mampu menyimpan data berkapasitas sangat besar hingga berukuran GigaByte sekalipun.
   c. MySQL didukung oleh drive ODBC, artinya MySQL dapat diakses menggunakan aplikasi apa saja termasuk berupa visual seberti Visual Basic dan Delphi.
   d. MySQL merupakan *database server multi user*.

2. Mengenal *Structures Query Language* (SQL)

   Structures Query Language (SQL) merupakan bahasa basis data yang terdiri atas sejumlah perintah yang diformulasikan dan dapat diberikan oleh *user* dan dikenali atau diproses oleh DBMS untuk melakukan suatu aksi tertentu. Terdapat dua perintah bahasa basis data yang digunakan, di antaranya:

   a. *Data Definition Language* (DDL)

      Data Definition Laguage (DDL) adalah sebuah metode *Query* SQL yang berguna untuk mendefinisikan data pada sebuah database. Perintah yang dimiliki DDL adalah sebagai berikut:

      1) *Create* : digunakan untuk membuat database dan tabel.
      2) *Drop* : digunakan untuk menghapus database dan tabel
      3) *Alter* : Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat baik menambah *Field* (*Add*), mengganti nama *Field* (*Change)* dan menghapus *Field* (*Drop*).

Dalam mendefinisikan data pada sebuah tabel, dibutuhkan sebuah tipe data yang nantinya membantu dalam pembuatan sebuah tabel. Terdapat beberapa tipe data yang digunakan di antaranya tipe data *numeric*, tipe data huruf (*String)*, tipe data *date* (tanggal), dan tipe data ENUM dan SET. Pada modul ini tipe data yang digunakan adalah tipe data String yaitu CHAR dan VARCHAR. Perbedaan kedua tipe data tersebut terletak pada ukuran penyimpanan datanya.

b. *Data Manipulation Language* (DML)

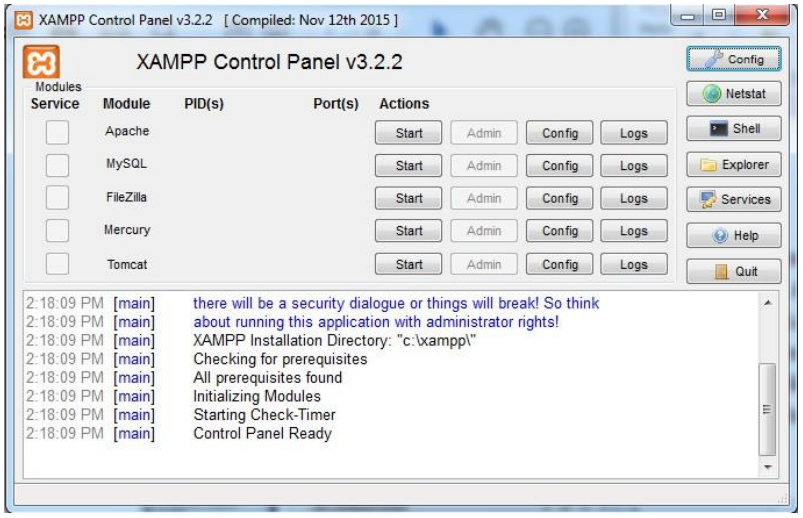
Merupakan metode *Query* yang berguna untuk melakukan manipulasi dan pengambilan data pada suatu database. Perintah yang dimiliki oleh DML untuk melakukan manipulasi database yang telah dibuat adalah sebagai berikut:

1) *Insert* : Digunakan Untuk memasukkan data pada database
2) *Update* : Digunakan untuk pengubahan data yang ada pada tabel database.
3) *Delete* : Digunakan untuk menghapus data pada database.

Untuk memperdalam DML akan dibahas pada modul berikutnya.

3. Menjalankan MySQL

Aplikasi yang digunakan untuk mengakses *server* MySQL berasal dari aplikasi XAMPP. XAMPP adalah sebuah paket kumpulan *software* yang terdiridari Apache, MySQL, PHPmyadmin, dan lain sebagianya.Aplikasi ini berfungsi untuk memudahkan instalasi sehingga tidak perlu menginstal aplikasi tersebut satu per satu.

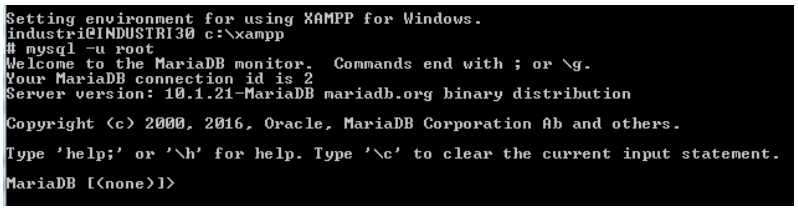


Gambar 5.1 Tampilan Aplikasi

XAMPP Berikut cara menjalankan MySQL dengan aplikasi XAMPP:

a. Buka aplikasi XAMPP
b. Aktifkan MySQL lalu klik Start
c. Setelah MySQL aktif lalu klik Shell.

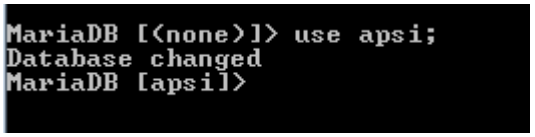
Untuk mengakkifkan MySQL ketikkan mysql –u root



4. Cara membuat database dengan menggunakan aplikasi XAMPP
    a. Membuat database

       CREATE database <nama database>; CREATE DATABASE apsi;

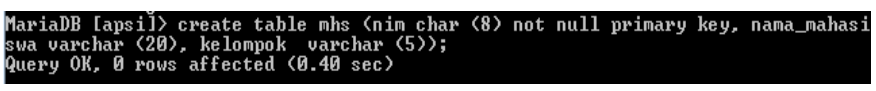
    b. Untuk menggunakan database yang telah dibuat ketikkan USE <nama database> USE apsi;



5. Cara membuat tabel dalam database
    a. Pastikan database yang akan digunakan, sudah digunakan.
    b. Membuat tabel dengan perintah

       CREATE TABLE <nama table> (<nama kolom><tipe data><panjang>);

       CREATE TABLE mhs (Nim char (8) NOT NULL PRIMARY KEY, nama_mahasiswaVARCHAR (20) kelompokVARCHAR (5));



Keterangan:

NOT NULL artinya setiap kolom tidak boleh kosong sedangkan jika diijinkan untuk dikosongkan dapat dengan menggunakan parameter. PRIMARY KEY adalah kunci utama, dalam setiap tabel harus ada minimal 1 kolom yang dijadikan sebagai *primary key*.

c. Untuk melihat struktur tabel dengan perintah DESCRIBE<namatabel>; DESC<namatabel>;

DESC mhs;

```
MariaDB [apsil]> desc mhs;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| nim            | char(8)     | NO   | PRI | NULL    |       |
| nama_mahasiswa | varchar(20) | YES  |     | NULL    |       |
| kelompok       | varchar(5)  | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
3 rows in set (0.02 sec)
```

d. Mengubah struktur tabel

Terdapat beberapa perintah dalam mengubah struktur tabel. Berikut perintah yang bisa digunakan:

1) Menambah kolom pada tabel dengan perintah

ALTER TABLE <nama table> ADD <kolom baru> INT AFTER <kolom lama>;

ALTER TABLE mhs ADD kelas INT AFTER kelompok;

```
MariaDB [apsil]> alter table mhs ADD kelas int after kelompok;
Query OK, 0 rows affected (0.26 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [apsil]> desc mhs;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| nim            | char(8)     | NO   | PRI | NULL    |       |
| nama_mahasiswa | varchar(20) | YES  |     | NULL    |       |
| kelompok       | varchar(5)  | YES  |     | NULL    |       |
| kelas          | int(11)     | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

2) Mengubah nama kolom

ALTER TABEL <nama table> CHANGE <kolom lama><kolom baru>< tipe><panjang>;

ALTER TABLE mhs CHANGE kelompok KEL VARCHAR (5);

```
MariaDB [apsil]> alter table mhs change kelompok KEL varchar(5);
Query OK, 0 rows affected (0.19 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [apsil]> desc mhs;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| nim            | char(8)     | NO   | PRI | NULL    |       |
| nama_mahasiswa | varchar(20) | YES  |     | NULL    |       |
| KEL            | varchar(5)  | YES  |     | NULL    |       |
| kelas          | int(11)     | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

3) Menghapus atau menghilangkan komponen pada tabel.

Menghapus ini dapat mencangkup menghilangkan primary key, kolom tabel dengan perintah

ALTER TABLE <namatabel> DROP <kolom yang akan dihapus>;

ALTER TABEL mhs DROP kelas;

```
MariaDB [apsil> alter table  mhs drop kelas;
Query OK, 0 rows affected (0.41 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [apsil> desc mhs;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| nim            | char(8)     | NO   | PRI | NULL    |       |
| nama_mahasiswa | varchar(20) | YES  |     | NULL    |       |
| KEL            | varchar(5)  | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

C. Latihan

Buatlah database baru yang berisikan tabel dengan ketentuan tabel seperti berikut ini :

Namakolom	Tipe data	Panjang	Keterangan
NIK	CHAR	10	Primary key
NAMA	Varchar	20	
Tempat lahir	Varchar	20	
Tanggal lahir	Varchar	10	
Alamat	Varchar	30	

MODUL VII
DML (DATA MANIPULATION LANGUAGE)

A. Tujuan Praktikum

1. Memahami Statement DML (Data Manipulation Language)
2. Menyisipkan baris ke dalam table
3. Merubah baris dalam table
4. Menghapus baris dari table
5. Mengontrol transaksi

B. Dasar Teori

1. Pengertian DML(Data Manipulation Language)

   DML atau Data Manipulation Language adalah kumpulan perintah *query* yang digunakan untuk memanipulasi data dalam tabel, seperti menambah, merubah atau menghapus data. Perintah ini tidak terkait dengan struktur dan metadata dari objek-objek yang berada pada tabel- tabel database.

2. Perintah-perintah pada DML (Data Manipulation Language)

   Berikut adalah perintah-perintah yang paling sering digunakan pada DML:

   a. **Insert** berfungsi untuk menambah data atau record pada database.

   b. **Delete** berfungsi untuk menghapus data pada database.

   c. **Update** yaitu perintah yang berfungsi untuk merubah maupun memperbarui data pada database.

   d. **Select** yaitu perintah yang digunakan untukmenampilkan data dari suatu tabel pada database.

Berikut contoh penggunaan perintah-perintah di dalam DML :

1) Mengisi data tabel

   Sejauh ini tabel-tabel baru tersebut masih kosong. Untuk mengisi tabel yang berupa *record* dapat diisi dengan perintah **insert :**

```
MariaDB [apsi]> insert into mhs values ('14000181', 'Fajar Kurniawan', '5');

Query OK, 1 row affected (0.34 sec)

MariaDB [apsi]>
```

   Perintah di atas berarti menambahkan sebuah *record* baru dengan isi dari *field* berturut-turut adalah ‘14000181’,’Fajar kurniawan’,’5’, sesuai dengan urutan deskripsi tabel **apsi ,** yaitu Nim, Nama_Mahasiswa dan

Kelompok.

2) Melihat isi table

```
MariaDB [apsi]> select *from mhs;
+----------+-----------------+----------+
| Nim      | nama_mahasiswa  | kelompok |
+----------+-----------------+----------+
| 14000110 | M. Rifki NH     | 10       |
| 14000112 | Ayu Lisa P      | 11       |
| 14000181 | Fajar Kurniawan | 5        |
| 14000192 | widi astuti     | 5        |
+----------+-----------------+----------+
4 rows in set (0.00 sec)
```

Perintah di atas akan memunculkan data apa adanya, tanpa syarat, tanpa pemilihan kolom dan tanpa urutan. Karakter '*'(bintang) menunjukkan semua *field* dibaca untuk -dalam kasus ini-ditampilkan secara tabular.

3) Membatasi jumlah *record* yang dibaca

Untuk membatasi *record* yang muncul atau untuk mencari *record* dengan kriteria tertentu, dapat digunakan klausa **where.** Misalkan untuk melihat nama_mahasiswa dan kelompok dengan Nim =14000110:

```
MariaDB [apsi]> select *from mhs where nim ='14000110';
+----------+---------------+----------+
| Nim      | nama_mahasiswa | kelompok |
+----------+---------------+----------+
| 14000110 | M. Rifki NH    | 10       |
+----------+---------------+----------+
1 row in set (0.06 sec)
```

4) Membatasi jumlah *field* yang dibaca

Untuk hanya melihat *field-field* tertentu dari tabel, gantikan karakter '*' dengan nama-nama *field* yang dikehendaki.Misalkan untuk melihat nim dan nama_mahasiswa:

```
MariaDB [apsi]> select nim, nama_mahasiswa from mhs;
+----------+-----------------+
| nim      | nama_mahasiswa  |
+----------+-----------------+
| 14000110 | M. Rifki NH     |
| 14000112 | Ayu Lisa P      |
| 14000181 | Fajar Kurniawan |
| 14000192 | widi astuti     |
+----------+-----------------+
4 rows in set (0.00 sec)
```

Jika dikombinasikan dengan filtering tingkat *record*, maka hasilnya akan semakin mengerucut.

5) Pengurutan kemunculan data

Klausa **order by** digunakan untuk mengurutkan data yang diminta dengan *query*.

a) Untuk menampilkan NIM dari yang terkecil (*keyword Ascending*/ **asc**):

```
MariaDB [apsi]> select nim, nama_mahasiswa,kelompok from mhs order by nim as
c;
+----------+----------------+----------+
| nim      | nama_mahasiswa | kelompok |
+----------+----------------+----------+
| 14000110 | M. Rifki NH    | 10       |
| 14000112 | Ayu Lisa P     | 11       |
| 14000181 | Fajar Kurniawan | 5       |
| 14000192 | widi astuti    | 5        |
+----------+----------------+----------+
4 rows in set (0.00 sec)
```

b) Untuk menampilkan NIM dari yang terbesar (*keyword Descending*/ **desc**):

```
MariaDB [apsi]> select nim, nama_mahasiswa,kelompok from mhs order by nim de
sc;
+----------+----------------+----------+
| nim      | nama_mahasiswa | kelompok |
+----------+----------------+----------+
| 14000192 | widi astuti    | 5        |
| 14000181 | Fajar Kurniawan | 5       |
| 14000112 | Ayu Lisa P     | 11       |
| 14000110 | M. Rifki NH    | 10       |
+----------+----------------+----------+
4 rows in set (0.00 sec)
```

6) Mengubah isi data *table*

Pengubahan data dilakukan pada saat ada data dari *real-world* yang berubah atau memang ada koreksi/perbaikan kesalahan dari pemasukan data sebelumnya. Untuk itu dapat digunakan perintah **update**. *Update* adalah perintah yang harus dilakukan secara hati-hati karena untuk kondisi ekstrim dapat mengubah isi keseluruhan *record* tabel. Intruksi ini pada umumnya dilakukan sambil dikomendasikan dengan filter *record* (menggunakan klausa *where*). Misalkan nama widi astuti akan diperbaiki/diubah menjadi widi A:

```
MariaDB [apsi]> update mhs set nama_mahasiswa='Widi A' where nim= '14000192'
;
Query OK, 1 row affected (0.08 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [apsi]> select *from mhs;
+----------+----------------+----------+
| Nim      | nama_mahasiswa | kelompok |
+----------+----------------+----------+
| 14000110 | M. Rifki NH    | 10       |
| 14000112 | Ayu Lisa P     | 11       |
| 14000181 | Fajar Kurniawan | 5       |
| 14000192 | Widi A         | 5        |
+----------+----------------+----------+
4 rows in set (0.00 sec)
```

7) Menghapus data *table*

Karena sesuatu hal, *record* dapat dihilangkan secara permanen dari suatu tabel. Perintahnya dengan menggunakan ***delete from***. Seperti halnya *update*, perintah *delete from* juga sangat berbahaya jika tidak dilakukan secara hati-hati. Kombinasi dengan klausa *where* selalu dilakukan. Pemilihan ekspresi *where* juga harus dipastikan hanya berefek pada *record-record* yang memang memenuhi kriteria yang untuk dihapus. Misalkan *record* widi A akan dihapus dari tabel:

```
MariaDB [apsi]> delete from mhs where nim='14000192';
Query OK, 1 row affected (0.30 sec)
```

```
MariaDB [apsi]> select *from mhs;
+----------+----------------+----------+
| Nim      | nama_mahasiswa | kelompok |
+----------+----------------+----------+
| 14000110 | M. Rifki NH    | 10       |
| 14000112 | Ayu Lisa P     | 11       |
| 14000181 | Fajar Kurniawan | 5       |
+----------+----------------+----------+
3 rows in set (0.00 sec)
```

DAFTAR PUSTAKA

Fathansyah. (2015). *Basis Data Revisi Kedua*. Bandung: Penerbit INFORMATIKA.

Fatta, Hanif Al. (2007). *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern*. Yogyakarta: Penerbit ANDI.