Ada beberapa hal yang harus dianalisis sebelum membuat perancangan sistem, yaitu :

1. ruang lingkup atau batasan system
2. apa yang ingin dihasilkan oleh sistem (tujuan sistem/ output)
3. siapa saja yang terlibat di dalamnya, dan sebagainya.

Di bahasan ini, penggambaran perancangan sistem yang digunakan adalah *Data Flow Diagram* (DFD), *Entity/ Relationship Diagram* (E/R Diagram), dan *Data Normalization*. DFD dibagi menjadi tiga strata (tingkatan), yaitu *Context Diagram*, *Zero Diagram*, dan *Detail Diagram*.

**Diagram Alir Data (DAD)** atau **Data Flow Diagram (DFD)**

DFD bukan flowchart. DFD adalah suatu diagram yang menggunakan notasi-notasi untuk menggambarkan menggambarkan aliran data dalam sebuah system. DFD menggambarkan semua proses, meskipun proses tersebut terjadi dalam waktu yang berbeda. Proses dalam DFD bisa berjalan secara parallel. DFD sering disebut juga Bubble chart, Bubble diagram, model proses, diagram alur kerja, atau model fungsi.Ketika menggambarkan sebuah sistem kontekstual data flow diagram yang akan pertama kali muncul adalah interaksi antara sistem dan entitas luar. DFD didisain untuk menunjukkan sebuah sistem yang terbagi-bagi menjadi suatu bagian sub-sistem yang lebih kecil adan untuk menggarisbawahi arus data antara kedua hal yang tersebut diatas. Diagram ini lalu “dikembangkan” untuk melihat lebih rinci sehingga dapat terlihat model-model yang terdapat di dalamnya.

**Tujuan DFD**

1. Memberikan indikasi mengenai bagaimana data ditransformasi pada saat data bergerak melalui system
2. Menggambarkan fungsi-fungsi(dan sub fungsi) yang mentransformasi aliran data

**Manfaat DFD**

1. Data Flow Diagram (DFD) adalah alat pembuatan model yang memungkinkan profesional sistem untuk menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual maupun komputerisasi.
2. DFD ini adalah salah satu alat pembuatan model yang sering digunakan,khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem.Dengan kata lain, DFD adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem.
3. DFD ini merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program.

**Simbol DFD**

**Terminator/Kesatuan luar (*External Entity*)**

Setiap sistem pasti mempunyai batas sistem (boundary) yang memisahkan suatu sistem dengan lingkungan luarnya. Kesatuan luar (external entity) merupakan kesatuan (entity) di lingkungan luar sistem yang berupa orang, organisasi atau sistem lainnya yang berada di lingkungan luarnya yang akan membeikan input atau menerima output dari sistem (Jogiyanto, 1989).

Suatu kesatuan luar dapat disimbolkan dengan suatu notasi kotak.

[Entitas Luar (external Entity)](http://fairuzelsaid.files.wordpress.com/2010/01/image001.gif)

Notasi terminator/Kesatuan Luar di DFD

Terminator dapat berupa orang, sekelompok orang, organisasi, departemen di dalam organisasi, atau perusahaan yang sama tetapi di luar kendali sistem yang sedang dibuat modelnya. Terminator dapat juga berupa departemen, divisi atau sistem di luar sistem yang berkomunikasi dengan sistem yang sedang dikembangkan.

**Arus data (data flow)**

Menggambarkan aliran data dari suatu proses ke proses lainnya. Merepresentasikan dengan menggunakan anak panah. Arus data (data flow) di DFD direpresentasikan dengan simbol suatu panah. Arus data ini mengalir diantara proses (Process), simpanan data (data store) dan kesatuan luar (external entity).



Notasi Arus Data di DFD

Nama proses ditulis untuk menjelaskan arti dalam aliran tersebut dan ditulis untuk

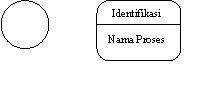
mengidentifikasi aliran tersebut.

Arus Arus data  data dapat dapat berbentuk berbentuk sebagai sebagai berikut berikut : :

* Formulir atau atau dokumen dokumen yang  yang digunakan digunakan perusahaan perusahaan
* Laporan tercetak tercetak yang  yang dihasilkan dihasilkan sistem sistem
* Output dilayar  komputer
* Masukan untuk komputer komputer
* Komunikasi ucapan
* Surat atau memo
* Data yang dibaca atau atau direkam di  file
* Suatu isian yang  yang dicatat pada buku agenda
* Transmisi data  dari suatu komputer ke komputer lain

**Proses (process)**

Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin, atau komputer dan hasil suatu arus data yang masuk ke dalam proses untuk dilakukan arus data yang akan keluar dari prises. Suatu proses dapat ditunjukkan dengan simbol lingkaran atau dengan simbol empat persegi panjang tegak dengan sudut-sudutnya tumpul.

[](http://fairuzelsaid.files.wordpress.com/2010/01/image004.gif)

Notasi Proses di DFD

Ada beberapa hal yang perlu diperhatikan tentang proses :

* Proses harus memiliki input dan output.
* Proses dapat dihubungkan dengan komponen terminator, data store atau proses melalui alur data.
* Sistem/bagian/divisi/departemen yang sedang dianalisis oleh profesional sistem digambarkan dengan komponen proses.

**Simpanan data (data store)**

Simpanan data (*data store*) merupakan simpanan dari data yang dapat berupa file atau database di sistem komputer, arsip atau catatan manual, kotak tempat data di meja seseorang, tabel acuan manual, agenda atau buku. Simpanan data di DFD dapat disimbolkan dengan sepasang garis horizontal paralel yang tertutup di salah satu ujungnya.



Lambang Penyimpan Data

**Peraturan penting dalam DFD**

1. Semua objek harus mempunyai nama
2. Aliran data harus diawali dan diakhiri oleh proses
3. Pemberian nomor pada komponen proses
4. Semua aliran data harus mempunyai tanda panah.
5. Penggambaran DFD sesering mungkin agar enak dilihat
6. Penghindaran penggambaran DFD yang rumit
7. Pemastian DFD yang dibentuk itu konsiten secara logika

**Langkah membuat/menggambar DFD**

Tidak ada aturan baku untuk menggambarkan DFD. Tapi dari berbagai referensi yang ada, secara garis besar langkah untuk membuat DFD adalah :

**Identifikasi Entitas Luar, Input dan Output**

Identifikasi terlebih dahulu semua entitas luar, input dan ouput yang terlibat di sistem.

**Buat Diagram Konteks (diagram context)**

Diagram konteks berisi gambaran umum (secara garis besar) sistem yang akan dibuat. Secara kalimat, dapat dikatakan bahwa diagram konteks ini berisi “siapa saja yang memberi data (dan data apa saja) ke sistem, serta kepada siapa saja informasi (dan informasi apa saja) yang harus dihasilkan sistem.”

Jadi, yang dibutuhkan adalah :

1. **Siapa** saja pihak yang akan memberikan data ke **system**
2. Data **apa** saja yang diberikannya ke system
3. kepada **siapa** sistem harus memberi informasi atau laporan, dan
4. **apa** saja isi/ jenis laporan yang harus dihasilkan **sistem**.

Kata “**Siapa**” di atas dilambangkan dengan kotak persegi (disebut dengan *terminator*), dan kata “**apa**” di atas dilambangkan dengan aliran data (disebut dengan *data flow*), dan kata “sistem” dilambangkan dengan lingkaran (disebut dengan *process*).



Contoh Diagram Konteks

**Buat Diagram Level Zero**

Tujuan dari diagram nol adalah untuk “memerinci” sebuah sistem menjadi “proses-proses” yang harus dilakukan ‘orang dalam.’ Atau jika dibuat dalam kalimat adalah : “Apa saja proses yang harus dilakukan agar mencapai sistem tersebut ?.”

Jadi, diagram ini adalah kelanjutan dari diagram konteks, yang “memperbanyak lingkaran,” sedangkan untuk (jumlah dan isi) *terminator* serta (jumlah dan isi) *data flow* dari dan ke terminator tersebut **harus tetap**.

Pada diagram ini pula mulai ditampilkan *data store* (penyimpan data/ *file*) yang dibutuhkan.

Caranya :

* Tentukan proses utama yang ada pada sistem.
* Tentukan apa yang diberikan/diterima masing-masing proses ke/dari sistem sambil memperhatikan konsep keseimbangan (alur data yang keluar/masuk dari suatu level harus sama dengan alur data yang masuk/keluar pada level berikutnya).
* Apabila diperlukan, munculkan data store (master) sebagai sumber maupun tujuan alur data.
* Hindari perpotongan arus data
* Beri nomor pada proses utama (nomor tidak menunjukkan urutan proses).



Diagram Nol

**Buat Diagram Level Satu**

Diagram ini merupakan dekomposisi dari diagram level zero.  
Caranya :

* Tentukan proses yang lebih kecil (sub-proses) dari proses utama yang ada di level zero.
* Tentukan apa yang diberikan/diterima masing-masing sub-proses ke/dari sistem dan perhatikan konsep keseimbangan.
* Apabila diperlukan, munculkan data store (transaksi) sebagai sumber maupun tujuan alur data.
* Hindari perpotongan arus data.
* Beri nomor pada masing-masing sub-proses yang menunjukkan dekomposisi dari proses sebelumnya.C ontoh : 1.1, 1.2, 2



Diagram Detil Proses 1.

***Entity/ Relationship Diagram* (Diagram E/R)**

Diagram E/R digunakan untuk memperlihatkan hubungan antar*file* (*data store*) yang ada di DFD. Di sini, *file* (*data store*) disebut dengan *entity* (entitas). Bagian dari *file*, yaitu *fields*, di sini disebut dengan *attributes* (atribut-atribut). Berikut lambang-lambangnya.



Gambar 6. Lambang-lambang Diagram E/ R

Berikut contoh diagram E/R dalam kasus ini :



Contoh Diagram E/R

Berikut penjelasan atribut-atribut yang digunakan :

|  |  |  |
| --- | --- | --- |
| Nama *File* (Entitas) | Nama *Field* (Atribut) | Keterangan |
| KASIR | NOPEG | Nomor Pegawai |
|  | NAMA | Nama Pegawai |
|  | ALAMAT | Alamat Pegawai |
| BARANG | KD\_BRG | Kode Barang |
|  | NM\_BRG | Nama Barang |
|  | HARGA | Harga Satuan Barang |
|  | STOK | Jumlah Barang Tersedia |
|  | MEREK | Merek Barang |
|  | JENIS | Jenis Barang |
| JUAL | NOPEG | Nomor Pegawai |
|  | KD\_BRG | Kode Barang |
|  | NO\_KWI | Nomor Kwitansi |
|  | JML\_BRG | Jumlah barang yang dibeli per kode barang |
|  | TTL\_BYR | Jumlah uang yang dibayar |

Atribut-atribut ini disesuaikan dengan kebutuhan oleh perancang sistemnya, misalkan, KASIR boleh saja memiliki atribut tempat dan tanggal lahir “TT\_Lahir,” tetapi karena atribut itu tidak diperlukan maka tidak perlu dibuat/ ditulis. Tetapi harus dipatuhi bahwa setiap atribut yang ada di entitas adalah memang merupakan atribut (identitas) dari entitasnya. Jadi, jangan masukkan atribut “NM\_BRG” di atribut KASIR, karena nama barang bukanlah atribut si KASIR.

Entitas “JUAL” adalah entitas yang merupakan *file* transaksi, jadi, (atribut-atribut) yang tercantum di sana adalah bagian-bagian transaksi yang harus dicatat. Adapun atribut NOPEG dan KD\_BRG adalah atribut dari entitas lain yang menjadi “jembatan” untuk mengambil atribut-atribut dari *master file*-nya. (Penjelasan ini ada di bagian berikutnya).

Ada kekuatan hubungan di dalam Diagram E/R yang dinamakan dengan derajat kardinalitas (*cardinality degree*). Ada empat jenis derajat kardinalitas yaitu (1) *one to one* (dilambangkan dengan 1 : 1), (2) *one to* *many (*dilambangkan dengan 1 : M), (3) *many to one* (dilambangkan dengan M : 1), dan (4) *many to many* (yang dilambangkan dengan M : M). Untuk menetapkan derajat kardinalitas di atas, ikuti kalimat-kalimat berikut ini :

“**Satu** KASIR bisa menJUAL **satu atau lebih** BARANG.” Satu atau lebih = *Many*.



Proses Pertama Penderajatan Kardinalitas Diagram E/R

Selanjutnya, kalimat yang dibalik :

“**Satu** (kode) BARANG bisa diJUAL oleh **satu atau lebih** KASIR”



Proses Kedua Penderajatan Kardinalitas Diagram E/R

Selanjutnya, pilih yang terbesar dari masing-masing sisi :



Proses Ketiga Penderajatan Kardinalitas

Kini kita dapatkan “M” (*many*) di kedua sisinya yang berarti derajat kardinalitas relasi tersebut adalah “*many to many*.” Namun, karena dalam matematika, nilai M akan selalu sama dengan M, sedangkan belum tentu kalimat (jika nilai M di atas = 10), “Sepuluh KASIR akan selalu menJUAL sepuluh BARANG,” maka penulisan M di salah satu sisinya diganti dengan N, tetapi pembacaannya tetap *many*. Jadi, nilai M dan N bisa jadi sama, dan bisa jadi tidak sama (M = N atau M ≠ N).



Proses Akhir Penderajatan Kardinalitas Diagram E/R

**NORMALISASI DATA TINGKAT PERTAMA**

Normalisasi data adalah salah satu cara membentuk sebuah *file* yang efektif dan efisien, sehingga dapat memanfaatkan *space* memori komputer seoptimal mungkin. Proses normalisasi data juga bertingkat-tingkat, dan di sini hanya akan dibahas mulai tingkat 1 hingga tingkat 3 saja.

Normalisasi data tingkat pertama (*First Normal Form/* 1NF) adalah proses penganalisisan setiap atribut yang ada di semua *file* yang terbentuk hingga Diagram E/R. Penganalisisan ditujukan agar setiap atribut yang dibentuk dapat diolah untuk menghasilkan informasi yang dibutuhkan.

Syarat 1NF adalah “setiap atribut harus bersifat atomik,” artinya, setiap atribut merupakan unsur terkecil dari identitas entitas (tidak perlu dipecah-pecah lagi). Contoh : akan kita periksa atribut “NAMA,” apakah atribut tersebut sudah atomik ?. Misalkan salah satu isi atribut NAMA adalah “Rachmi Hidayat,” jika nama itu selamanya akan digunakan demikian, maka atribut tersebut sudah atomik. Tetapi, jika nama itu suatu saat harus dicetak menjadi “Hidayat, Rachmi,” maka atribut NAMA tersebut belum atomik. Bagaimana mencetak “Rachmi Hidayat” menjadi “Hidayat Rachmi” ?.

Jadi, jika di “negara barat,” biasanya atribut nama dipecah-pecah menjadi *first name* (FNAME), *middle name* (MNAME), dan *last name* (LNAME), karena memang nama mereka sering dibolak-balik.

Periksa juga atribut-atribut lainnya, seperti atribut ALAMAT, perlu tidak dipecah-pecah. Bayangkan saja, jika panjang atribut ALAMAT 100 karakter, kemudian kita diminta mengirim surat, apa jadinya ketika di amplop kita *print* alamat yang panjangnya 100 karakter ?.

**NORMALISASI DATA TINGKAT KEDUA**

Normalisasi data tingkat kedua (*second normal form*/ 2NF) bersyarat : (1) telah memenuhi 1NF, (2) setiap atribut *non key* harus tergantung secara fungsional dengan atribut *key*-nya. Apa itu atribut *key* dan *non key* ?.

***Key Field* (Kunci Atribut)**

Kunci atribut adalah atribut yang dipilih untuk dapat mewakili atribut-atribut lain dalam sebuah *record* yang membedakan dengan *record-record* lainnya Misalkan, untuk seorang mahasiswa, apa atribut yang bisa mewakili dirinya yang bisa membedakan dengan mahasiswa lainnya ?.

Jika dipilih nama, banyak nama mahasiswa lain yang sama. Jika dipilih tanggal lahir, kemungkinan akan ada yang sama, dan sebagainya. Bagaimana cara menentukannya ?.

***Super Key* (Kunci Super)**

Kunci super adalah satu atau beberapa kombinasi atribut yang mungkin dapat dipilih menjadi kunci atribut. Misalkan, *File* “Mahasiswa” memiliki atribut-atribut : NPM, NAMA, KELAS, ALAMAT, dan TGL\_LAHIR, maka atribut(-atribut) yang bisa dipilih menjadi kunci atribut adalah :

1. NPM
2. NAMA (dengan syarat tidak ada nama mahasiswa yang sama)
3. NPM + NAMA
4. NPM + TGL\_LAHIR
5. NPM + NAMA + TGL\_LAHIR
6. Dan seterusnya, berbagai kombinasi yang mungkin

***Candidate Key* (Kunci Calon)**

Kunci calon adalah kunci atribut yang merupakan (kumpulan) kunci atribut yang jumlahnya paling sedikit di kunci super. Kita dapatkan dua buah atribut yang merupakan kunci atribut dengan jumlah atributnya tersedikit (1 atribut), yaitu :

1. NPM
2. NAMA (dengan syarat tidak ada nama mahasiswa yang sama)

***Primary Key* (Kunci Utama),** yang sering disebut dengan ***key field***.

Adalah kunci kandidat yang dipilih untuk dijadikan *key field*. Pemilihan dilakukan dengan mempertimbangkan kemungkinan yang tidak akan pernah sama, maka kunci atribut yang dipilih adalah NPM.

***Alternate Key* (Kunci Alternatif)**

Kunci alternatif adalah kunci kandidat yang tidak terpilih menjadi *primary key*, dalam hal ini, NAMA.

***Foreign Key* (Kunci Tamu)**

Adalah kunci utama dari *file* (*master*) lain yang digunakan di *file* (transaksi). Kunci atribut tersebut digunakan sebagai “jembatan” untuk mengambil nilai data dari atribut-atribut lain. Perhatikan Diagram E/R berikut ini (dari kasus sebelumnya) :



Gambar 12. Penulisan Kunci Atribut di Diagram E/R

Di *master file* (*entity*) KASIR, kunci utamanya : NOPEG (nomor pegawai)

Di *master file* (*entity*) BARANG, kunci utamanya : KD\_BRG (kode barang)

Di *transaction file* (*relationship*) JUAL, kunci utamanya : NO\_KWI (nomor kwitansi)

kunci tamunya : NOPEG dan KD\_BRG

NOPEG sebagai kunci tamu di JUAL digunakan untuk menganbil nilai data atribut NAMA dan ALAMAT di *file* KASIR. KD\_BRG sebagai kunci tamu di JUAL digunakan untuk mengambil nilai data NM\_BRG, HARGA, dan berbagai atribut di *file* BARANG, sehingga bisa diketahui barang tertentu dijual oleh kasir yang mana.

Kembali ke 2NF, kata “tergantung secara fungsional” menurut pengertiannya adalah (untuk contoh kasus *file* KASIR yang kunci atributnya NOPEG ) : “jika NOPEG berubah, maka harus pasti orangnya (NAMA, dan ALAMAT si Kasir) berubah pula.” Meski demikian, bisa saja alamatnya tidak berubah karena dua orang kasir tinggal di alamat yang sama, itu tidak masalah.

Kasus di atas sudah memenuhi syarat 2NF. Contoh jika belum memenuhi syarat 2 NF adalah jika pada *file* BARANG terdapat atribut NAMA (kasir), atau TGL\_BYR (tanggal bayar di JUAL), dan berbagi contoh lainnya yang pada satu *file* terdapat atribut yang tidak semestinya dapat dijadikan atribut *file* tersebut.

**NORMALISASI DATA TINGKAT KETIGA**

Normalisasi data tingkat ketiga (*third normal form*/ 3NF) bersyarat : (1) telah memenuhi 2NF, (2) setiap atribut *non key* tidak boleh tergantung dengan atribut *non key* lainnya (tidak boleh terjadi ketergantungan transitif).

Contoh transitif : Jika A → B, dan B → C, maka sudah pasti A → C.

Kebetulan, di kasus di atas tidak ada atribut yang bersifat transitif.

Contoh atribut yang bersifat transitif. Jika A = NPM, B = KODE\_POS, dan C = KOTA yang misalnya ada di *file* MAHASISWA, maka *file* tersebut harus dipecah menjadi, *file* 1 (MAHASISWA) berisi atribut NPM, dan KODE\_POS, dan *file* 2 (KODEPOS) berisi atribut KODE\_POS dan KOTA.