BAB II

LANDASAN TEORI

2.1. Tinjauan Jurnal

Menurut Ladjamudin dalam Octafian (2011:150) Sistem Informasi dapat didefinisikan sebagai berikut :

1. Suatu sistem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk mencapai suatu tujuan yaitu menyajikan informasi.
2. Sekumpulan prosedur organisasi yang pada saat dilaksanakan akan memberikan informasi bagi pengambilan keputusan dan atau untuk mengendalikan organisasi.
3. Suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi, mendukung operasi, bersifat manajerial, dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.

Menurut Hermawan dalam Nurcahyono, dkk (2012:230) Aplikasi adalah penggunaan atau penerapan suatu konsep yang menjadi pokok pembahasan. Aplikasi dapat diartikan juga sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu. Aplikasi *Software* yang dirancang untuk penggunaan praktisi khusus, klasifikasi luas ini dapat dibagi menjadi 2 (dua), yaitu :

1. Aplikasi *Software* Spesialis, program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.

2. Aplikasi paket, suatu program dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu.

2.2. Konsep Dasar Sistem

Menurut McLeod (2008:10) ” Sistem informasi adalah suatu sistem virtual yang memungkinkan manajement mengendalikan operasi sistem fisik perusahaan”.

Sistem fisik (*physical system*) perusahaan terdiri atas sumber-sumber daya berwujud seperti bahan baku, karyawan, mesin dan uang. Sedangkan sistem virtual (*virtual system*) terdiri atas sumber daya informasi yang digunakan untuk mewakili sistem fisik.

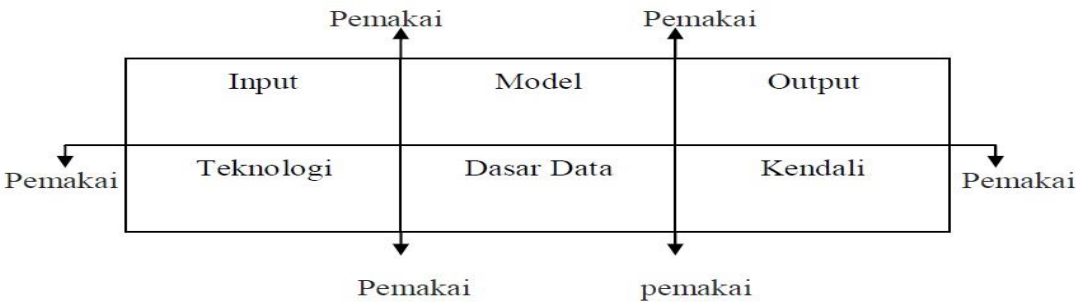
Sedangkan dalam buku Yakub (2012 : 20) yang berjudul Pengantar Sistem Informasi, komponen sistem informasi tersebut disebut dengan istilah blok bangunan (*building block*). Komponen sistem informasi tersebut terdiri dari blok masukan *(input block)*, blok model *(model block)*, blok keluaran *(output block)*, blok teknologi *(technology block)*, blok basis data *(database block)*.

a. Blok masukan *(input block)*, input blok memiliki data yang masuk ke dalam sistem informasi, juga metode-metode untuk menangkap data yang dimasukkan.
b. Blok model *(model block)*, blok ini terdiri dari kombinasi prosedur logika dan model matematik yang memanipulasi data *input* dan data yang tersimpan di basis data.
c. Blok keluaran *(output block)*, produk dari sistem informasi adalah keluaran yang merupakan informasi berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

d. Blok teknologi *(technology block)*, blok teknologi diguankan untuk menerima *input*, menyimpan, mengakses data, menghasilkan dan mengirimkan keluaran dari sistem secara

keseluruhan. Teknologi terdiri dari tiga bagian utama, yaitu teknisi (*brainware*), perangkat lunak (*software*), dan perangkat keras (*hardware*).

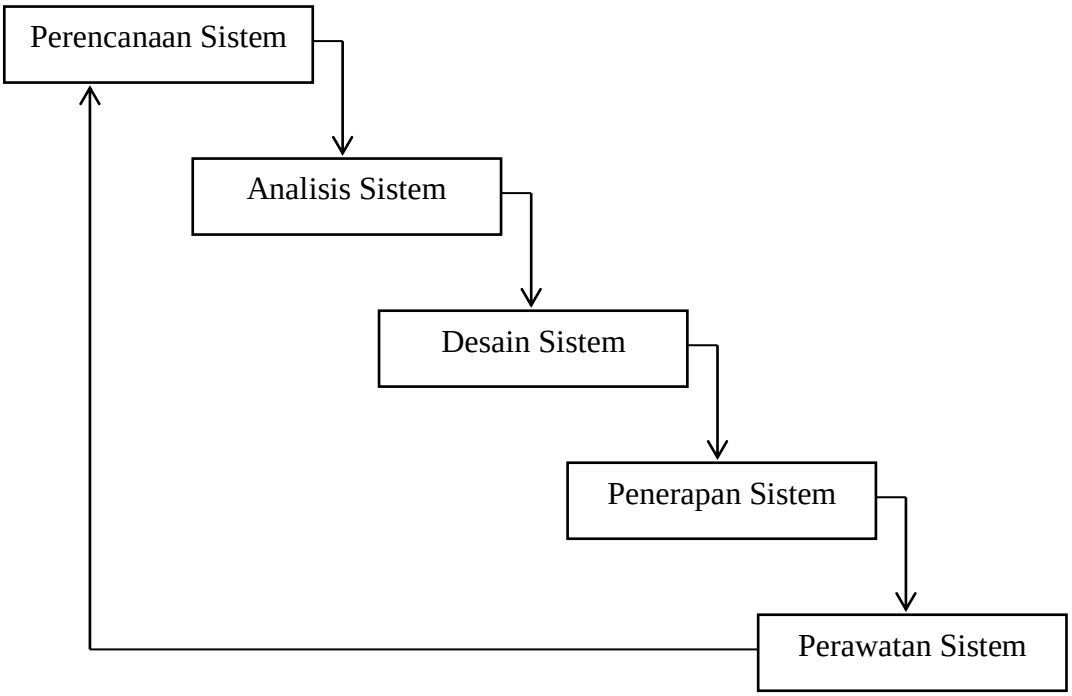
e. Blok basis data *(database block),* basis data merupakan kumpulan data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras computer dan digunakan perangkat lunak (*software*) untuk memanipulasinya.



(Sumber : Yakub(2012:20))
Gambar II.1
Blok Sistem Informasi yang Berinteraksi

A. *System Development Life Cycle(SDLC)*

Menurut Supriyanto (2007:271) "Siklus hidup pengembangan sistem (SDLC) pendekatan sistem yang disebut pendekatan air terjun (*waterfall approach*), yang menggunakan beberapa tahapan dalam mengembangkan sistem". Alasan menggunakan analis sistem di metode SDLC adalah karena metode ini digunakan untuk mengembangkan sistem teknologi informasi yang kompleks. Tahapan-tahapan dalam metode SDLC digambarkan dalam struktur metodologi SDLC sebagai berikut :



(Sumber : Supriyanto (2007:272))
Gambar II.2
System Devlopment Life Cycle (SDLC)

Penjelasan dari gambar tersebut adalah sebagai berikut :

1. Perencanaan Sistem (*System Planning*)

   Sebagai tahap awal pengembangan sistem yang mendefinisikan perkiraan kebutuhan-kebutuhan sumber daya seperti perangkat fisik, manusia, metode (teknik dan operasi), dan anggaran yang sifatnya masih umum. Langkah-langkah perencanaan sebagai berikut:

   a. Menyadari adanya masalah

      Setiap bagian organisasi harus bisa merasakan apasaja permasalahan yang selama ini terjadi.

   b. Mendefinisikan masalah

      Setelah menyadari adanya masalah, selanjutnya organisasi harus memahami sebab-sebab terjadinya permasalahan.

c. Menentukan tujuan sistem.

Manajer dan anilis sistem menentukan tujuan sistem yang harus dipenuhi oleh sistem untuk memuaskan atau memenuhi kebutuhan pemakainya.

Dalam tahap ini hal yang pertama dilakukan adalah melakukan wawancara dan observsi

2. Analisis Sistem (*System Analysis*)

Tahap penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem yang baru atau diperbarui. Rincian langkahnya tahap analisis adalah identifikasi masalah dengan melakukan penelitian, mengorganisasi tim dengan menyusun tim proyek yang terlibat termasuk pemakai sistem yang nantinya digunakan pada kegiatannya, mendefinisikan kebutuhan informasi (seperti: dengan melakukan wawancara, pengamatan, pencarian pencatatan dan survei), mendefinisikan kriteria kinerja sistem yaitu dengan memahami bagaimana pengguna melakukan pekerjaannya dari awal hingga (bagaimana mulai melakukan hingga mengakhiri aktivitas, data, informasi dan laporan yang dibutuhkan dan dihasilkannya) dan membuat laporan hasil analisis.

a. Analisa Teknologi

Menganalisis teknologi apa yang digunakan dalam membuat program, maka memerlukan teknologi seperti *Visual Basic 6.0.* .Memerlukan data transaksi dan stok barang digunakan *database* seperti *SQL, Ms. Access.*

b. Analisa Informasi

Mengenai informasi data yang akan menjadi data tetap dan data dinamis, kategori informasi data tetap adalah : kasir (pengguna). Informasi dinamis

adalah informasi yang selalu berubah dalam setiap periodik dapat setiap hari atau setiap minggu. Informasi dinamis dalam sistem ini adalah :

1. Informasi persediaan (*stock*) produk
2. Informasi harga produk
3. Informasi produk baru

c. Analisa User

Mengkatogorikan user yang digunakan dalam *Login* program. Pengguna sebagai pemilik atau kasir.

d. Analisa Biaya dan Resiko

Dalam tahap ini diperhitungkan biaya yang akan dikeluarkan seperti biaya *maintenance* atau perbaikan.

3. Desain/Perancangan Sistem (*System Design*)

Tahap setelah analisis sistem yang menentukan proses dan data yang diperlukan oleh sistem baru. Langkah-langkah yang dilakukan adalah menyiapkan rancangan sistem yang terinci/grafis, dan yang umum berupa informasi serta menyiapkan usulan implementasi.

4. Penerapan/Implementasi Sistem (*System Implementation*)

a. Penulisan Program dan Instalasi

Merupakan tahap penulisan program yang telah dianalisis dan dirancang semua maka program yang digunakan adalah *Java Netbeans*.

b. Desain *Review*

Dalam tahap ini tidak hanya menguji *design* yang digunakan namun menguji semua sistem yang telah diterapkan seperti stok barang dan faktur.

c. Pemilihan Sumber daya *Hardware* dan *Software*

Dalam tahap ini *software* dan *hardware* digunakan untuk Aplikasi Penjualan.

d. Pengujian Program

Menguji Program dengan melakukan transaksi dengan program yang telah dibuat, serta pemeriksaan dokumen laporan.

5. Perawatan Sistem (*System Maintenance*)

Sistem perlu dirawat karena beberapa hal, yang meliputi penggunaan sistem, audit sistem, penjagaan, perbaikan, dan peningkatan sistem.

2.3. Konsep Dasar Program

Pembuatan program tentunya tidak terlepas dari tahapan-tahapan yang harus dikerjakan secara terstruktur untuk membantu pemrogram dalam menyelesaikan programnya dengan baik. Untuk lebih jelasnya tahapan-tahapan perancangan program secara umum adalah sebagai berikut :

1. Definisi masalah

Tujuannya untuk mendapatkan pemahaman tentang permasalahan yang ada, sehingga akan diperoleh asumsi-asumsi yang benar untuk dapat memecahkan permasalahan.

2. Analisis kebutuhan

Langkah ini dilakukan untuk menentukan masukan dan keluaran yang diinginkan serta sebagai gambaran tentang data yang akan diproses sehingga program yang disusun terarah dan menghasilkan informasi yang dibutuhkan.

3. Desain algoritma

   Menulis langkah-langkah dalam pemecahan yang ada dengan menggunakan simbol-simbol untuk menceritakan aktivitas data yang akan diolah menjadi informasi.

4. Pengkodean

   Merupakan pengkodean dari algoritma yang dibuat, diterjemahkan kedalam bentuk *statement* yang sesuai dengan bahasa pemrograman yang digunakan.

5. Melakukan tes program

   Dari proses logika yang sudah dibuat, diperiksa apakah program tersebut sudah benar dan bebas dari kesalahan atau masih harus diperbaiki kembali. Semua kesalahan yang terjadi diperbaiki agar program komputer dapat dijalankan dan memberi hasil sesuai yang diharapkan.

6. Dokumentasi program

   Dokumentasi merupakan informasi dan gambaran tambahan yang sangat membantu untuk memahami sebuah kode yang diberikan. Tujuannya untuk menjadi pedoman dan penjelasan bagi para pemakai.

7. Pemeliharaan

   Pemeliharaan digunakan untuk menjabarkan aktivitas dari analisis sistem pada saat perangkat lunak telah dipergunakan oleh pemakai.

Konsep dasar program yang digunakan dalam pembuatan Aplikasi program adalah :

1. Program *Java Netbeans*

Dalam penulisan skripsi ini penulis menggunakan bahasa pemrograman *Java Development Kit (JDK)* sebagai *tools* pengembang aplikasi *Java* yang berisi sekumpulan kakas baris perintah (*command - line tool*) untuk menciptakan

program java. *Java Development Kit (JDK)* berisi sekumpulan kakas, utilitas, dan dokumentasi serta kode applet contoh untuk pengembangan program *Java*. (Bambang, 2007)

2. *MySQL*

MySQL merupakan *software Database Management System* (DBMS) artinya *database* yang paling populer dikalangan pemrogram digunakan untuk membangun aplikasi yang menggunakan *database* sebagai sumber dan pengelola datanya. Pada *MySQL* sebuah *database* terdiri atas tabel-tabel. Sebuah tabel terdiri atas baris dan kolom (Suryatiningsih, 2009).

MySQL AB tersedia sebagai *software* gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL.

MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia *MySQL* AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL* AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius. *MySQL* memiliki beberapa keistimewaan, antara lain yaitu :

1. Portabilitas.

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti *Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga*, dan masih banyak lagi.

2. Perangkat lunak sumber terbuka (*Open Source*).

MySQL didistribusikan sebagai perangkat lunak sumber terbuka (*open source*), *Open source* berarti semua orang diizinkan menggunakan dan

memodifikasi *MySQL*. Semua orang dapat mendownload *software MySQL* dari internet dan menggunakannya tanpa membayar sehingga tidak perlu mengeluarkan biaya tambahan untuk membeli *software* tersebut.

3. *Performance tuning*.

MySQL memiliki kecepatan yang menakjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak *SQL* per satuan waktu.

4. Ragam tipe data.

MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double*, *char, text, date, timestamp*, dan lain-lain

5. Perintah dan Fungsi.

MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*)

6. Keamanan.

MySQL memiliki beberapa lapisan keamanan seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi. *MySQL* menyimpan data di dalam direktori khusus yang terpisah dari *file* program sehingga keamanan data lebih terjamin

7. Skalabilitas dan Pembatasan.

MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas *indeks* yang dapat ditampung mencapai 32 indeks pada tiap tabelnya

8. Konektivitas.

MySQL dapat melakukan koneksi dengan klien menggunakan protocol *TCP/IP*, *Unix socket*, atau *Named Pipes*.

9. Antar Muka.

MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*

10. Klien dan Peralatan.

MySQL dilengkapi dengan berbagai peralatan (*tool)* yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.

11. Struktur table.

MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani *ALTER TABLE,* dibandingkan basis data lainnya semacam *PostgreSQL* ataupun *Oracle.*

2.4 . Peralatan Pendukung Sistem (*Tools System*)

Peralatan pendukung yang digunakan penulis dalam pembuatan skripsi adalah :

2.4.1 *Unifield Modelling Language* (UML)

Menurut UML (*Unified Modeling Language*) menyediakan teknologi yang diperlukan untuk mendukung rekayasa perangkat lunak berorientasi objek, tetapi UML tidak menyediakan kerangka kerja proses untuk memandu tim-tim proyek dalam mengaplikasikan teknologi ini. setelah beberapa tahun berikutnya, Jacobson, Rumbaugh mengembangkan Proses Terpadu (*Unified*

Process), suatu kerangka kerja untuk rekayasa perangkat lunak menggunakan UML. Saat ini UP (*Unified Process*) dan UML digunakan secara luas pada proyek-proyek berorientasi objek yang beragam. Model yang bersifat iteratif dan incremental diusulkan oleh UP dan sebaiknya diadaptasi untuk memperoleh kebutuhankebutuhan proyek perangkat lunak yang bersifat spesifik. UML (*Unified Modeling Language*) merupakan sarana bagus untuk mengekspresikan model orientasi objek diberagam level abstraksi mulai level konseptual sampai level implementasi, dan beragam pandangan statis dan dinamis. Pemodelan memperjelas yang perlu dan telah dilakukan pengembang.

1. *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Hal yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

2. *Activity Diagram*

Pada tahap, pemodelan bisnis, diagram aktivitas dapat digunakan untuk menunjukan aliran kerja bisnis (*business work-flow*). Dapat juga digunakan untuk menggambarkan aliran kejadian (*flow of events*) dalam *use case.*

Activity Diagram mempunyai peran seperti halnya flowchart, akan tetapi perbedaannya dengan flowchart adalah *activity diagram* bisa mendukung perilaku paralel sedangkan flowchart tidak bisa. *Activity* diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya *(internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan *behaviour internal* sebuah sistem dan interaksi antar subsistem secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana *actor* menggunakan sistem untuk melakukan aktivitas. Sama seperti *state*, standar UML menggunakan segi empat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu, digambarkan dengan simbol belah ketupat. Untuk mengilustrasikan proses paralel *(fork and join)* digunakan titik *sinkronisasi* yang dapat berupa titik, garis *horizontal* atau *vertikal*. *Aktivity diagram* dapat dibagi menjadi beberapa *objectswimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

3. *Sequence Diagram*

Sequence diagram terdiri antar dimensi vertikal (waktu) dan dimensi *horizontal* (objek-objek yang terkait). Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki

lifeline vertikal. Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya.

Pada fase *desain* berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class. Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary, controller* dan *persistent entity*.

4. *Deployment Diagram*

Bagian *hadware* adalah node yaitu nama untuk semua jenis sumber komputasi. Ada dua tipe node yaitu *processor* dan *device.* Processor adalah node yang bisa mengeksekusi sebuah komponen sedangkan *device* tidak. *Device* adalah perangkat keras seperti printer, monitor dan perangkat keras.

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya.

2.4.2 *Entity Relationship Diagram (ERD)*

Menurut Fathansyah (2007:79) "*Entity Relationship Model*/ER_Mberisi komponen-komponen himpunan entitas dan himpunan relasi yang masing-masing dilengkapi dengan atribut-atribut yang merepresentasikan seluruh fakta dari 'dunia nyata' yang kita tinjau, dapat digambarkan dengan lebih sistematis dengan menggunakan *Diagram Entity – Relationship* (Diagram E-R)". ER_M

digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pemakai secara logik. ER_M didasarkan pada suatu persepsi bahwa *real world*terdiri atas obyek-obyek dasar yang mempunyai hubungan/kerelasian antar obyek-obyek dasar tersebut. ER_M digambarkan dalam bentuk diagram yang disebut dengan ER (*ER_Diagram*/ER_D) dengan menggunakan simbol-simbol grafis terntentu.Sebuah diagram E-R tersusun atas tiga komponen, yaitu: Entitas, Atribut (*Attribute*) dan Kerelasian Antar Entitas (*Relationship*).

Menurut Fathansyah (2007:80) notasi simbolik didalam diagram E-R yang dapat digunakan sebagai berikut:

1. Persegi panjang menyatakan himpunan entitas
2. Lingkaran/elip menyatakan atribut (atribut yang berfungsi sebagai *key* digaris bawahi)
3. Belah ketupat menyatakan himpunan relasi
4. Garis sebagai penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya.
5. Kardinalitas relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka (1 dan 1 untuk relasi satu-ke-satu, dan N untuk relasi satu-ke-banyak atau N dan N untuk relasi banyak ke banyak)

Menurut Fathansyah (2007:84) tahapan dalam pembuatan diagram E-R, yaitu:

1. Mengidentifikasikandan menetapkan seluruh himpunan entitas yang akan terlibat
2. Menentukan atribut *key* dari masing-masing himpunan entitas

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

Menurut Fathansyah (2007:80) kerelasian antar entitas dapat dikelompokkan dalam tiga jenis, yaitu:

1. Kerelasian jenis 1-ke-1/satu ke satu (*one to one*)

   Kerelasian jenis ini terjadi jika kejadian atau tansaksi diantara dua entitas yang berhubungan hanya memungkinkan terjadi sebuah kejadian atau transaksi pada kedua entitas

2. Kerelasian jenis n-ke-1/banyak ke satu (*many to one*) atau 1-ke-n/satu ke banyak (*one to many*)

   Kerelasian jenis ini terjadi jika kejadian atau transaksi diantara dua entitas yang berhubungan hanya memungkinkan terjadi satu kali dalam entitas pertama dan dapat terjadi lebih dari satu kali kejadian atau transaksi pada entitas kedua.

3. Kerelasian jenis n-ke-n/banyak ke banyak (*many to many*)

   Kerelasian jenis ini terjadi jika kejadian atau transaksi di antara dua entitas yang berhubungan memungkinkan terjadi lebih dari satu kali dalam entitas pertama dan entitas kedua.