

BAB II

LANDASAN TEORI

2.1. Konsep Dasar Sistem

Pada umumnya setiap organisasi mempunyai sistem informasi dalam mengumpulkan, menyimpan, melihat, dan menyalurkan informasi dalam membuat perancangan sistem informasi.

Konsep dasar sistem merupakan sekelompok komponen berbasis komputer yang dibuat oleh manusia dalam mengelola data, menyimpan, menghimpun kerangka kerja serta mengkoordinasikan sumber daya manusia dan komputer untuk mengubah sistem masukan menjadi sistem keluaran untuk mencapai tujuan dan sasaran yang telah ditetapkan sebelumnya.

2.1.1. Pengertian sistem

Secara garis besar sistem merupakan suatu kumpulan komponen dan elemen yang saling terintegrasi, komponen yang terorganisir dan bekerja sama dalam mewujudkan suatu tujuan tertentu.

Menurut Sutanto dalam Djahir dan Pratita (2015:6) mengemukakan bahwa “sistem adalah kumpulan/grup dari subsistem/bagian/komponen apapun, baik fisik ataupun nonfisik yang saling berhubungan satu sama lain dan bekerja sama secara harmonis untuk mencapai satu tujuan tertentu”. Sedangkan menurut Mulyani (2016:2) menyatakan bahwa “sistem bisa diartikan sebagai sekumpulan sub sistem, komponen yang saling bekerja sama dengan tujuan yang sama untuk menghasilkan *output* yang sudah ditentukan sebelumnya”. Selain itu menurut

Hutahaean (2015:2) mengemukakan bahwa “sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan kegiatan atau untuk melakukan sasaran yang tertentu”.

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa sistem merupakan suatu kumpulan komponen dari subsistem yang saling bekerja sama dari prosedur-prosedur yang saling berhubungan untuk menghasilkan *output* dalam mencapai tujuan tertentu.

A. Karakteristik sistem

Suatu sistem mempunyai ciri-ciri karakteristik yang terdapat pada sekumpulan elemen yang harus dipahami dalam megidentifikasi pembuatan sistem. Adapun karakteristik sistem (Hutahaean, 2015:3) yang dimaksud adalah sebagai berikut:

1. Komponen

Sistem terdiri dari sejumlah komponen yang saling berinteraksi dan bekerja sama untuk membentuk satu kesatuan. Komponen sistem dapat berupa sub sistem atau bagian-bagian dari sistem.

2. Batasan sistem (*boundary*)

Daerah yang membatasi antara suatu sistem dengan sistem lainnya atau dengan lingkungan luar dinamakan dengan batasan sistem. Batasan sistem ini memungkinkan sistem dipandang sebagai satu kesatuan dan juga menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan luar sistem (*environment*)

Apapun yang berada di luar batas dari sistem dan mempengaruhi sistem tersebut dinamakan dengan lingkungan luar sistem. Lingkungan luar yang

bersifat menguntungkan wajib dipelihara dan yang merugikan harus dikendalikan agar tidak mengganggu kelangsungan sistem.

4. Penghubung sistem (*interface*)

Media penghubung diperlukan untuk mengalirkan sumber-sumber daya dari sub sistem ke sub sistem lainnya dinamakan dengan penghubung sistem.

5. Masukkan sistem (*input*)

Energi yang dimasukkan ke dalam sistem dinamakan dengan masukan sistem (*input*) dapat berupa perawatan dan masukan sinyal. Perawatan ini berfungsi agar sistem dapat beroperasi dan masukan sinyal adalah energi yang diproses untuk menghasilkan keluaran (*output*).

6. Keluaran sistem (*output*)

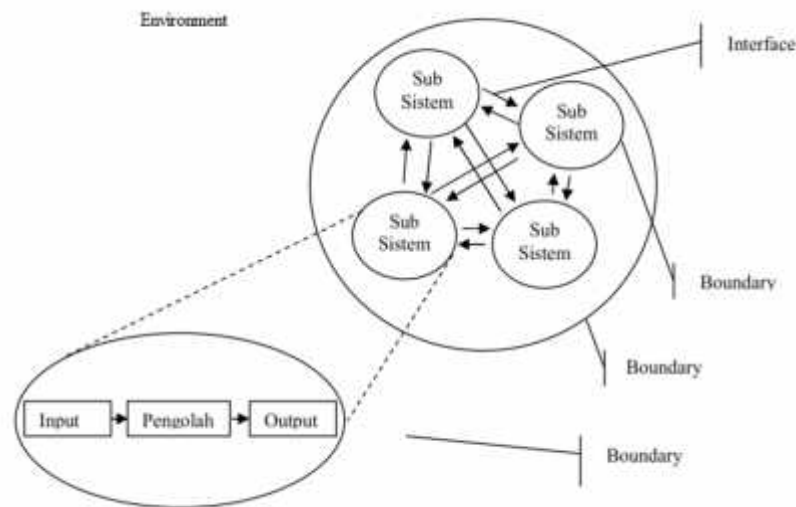
Hasil dari energi yang telah diolah dan diklasifikasikan menjadi keluaran yang berguna dinamakan dengan keluaran sistem (*output*). Informasi merupakan contoh keluaran sistem.

7. Pengolah sistem

Untuk mengolah masukan menjadi keluaran diperlukan suatu pengolah yang dinamakan dengan pengolah sistem.

8. Sasaran sistem

Sistem pasti memiliki tujuan atau sasaran yang sangat menentukan *input* yang dibutuhkan oleh sistem dan keluaran yang dihasilkan.



Sumber: Hutahaeen (2015:5)

Gambar II.1. Karakteristik Sistem

B. Klasifikasi sistem

Sistem merupakan suatu bentuk integrasi antara satu komponen dan komponen lain karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi di dalam sistem tersebut. Oleh karena itu sistem dapat diklasifikasikan dari beberapa sudut pandang. Adapun klasifikasi sistem menurut (Hutahaeen, 2015:6) diuraikan sebagai berikut:

1. Sistem Abstrak dan Sistem Fisik

Sistem abstrak merupakan sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik, misalnya sistem telogi. Sedangkan sistem fisik diartikan sebagai sistem yang nampak secara fisik sehingga setiap mahluk dapat melihatnya, misalnya sistem komputer.

2. Sistem Alamiah dan Sistem Buatan Manusia

Sistem alamiah merupakan sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia, misalnya sistem tata surya, sistem galaksi, sistem

reproduksi dan lain-lain. Sedangkan sistem buatan manusia merupakan sistem yang dirancang oleh manusia. Sistem buatan yang melibatkan interaksi manusia, misalnya sistem akuntansi, sistem informasi, dan lain-lain.

3. Sistem Deterministik dan Sistem Probabilistik

Sistem deterministik merupakan sistem yang beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi bagian-bagiannya dapat dideteksi dengan pasti sehingga keluaran dari sistem dapat diramalkan, misalnya sistem komputer, adalah contoh sistem yang tingkah lakunya dapat dipastikan berdasarkan program-program komputer yang dijalankan. Sedangkan sistem probabilistik merupakan sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas, misalnya sistem manusia.

4. Sistem Terbuka dan Sistem Tertutup

Sistem terbuka merupakan sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Lebih spesifik dikenal juga yang disebut dengan sistem terotomasi, yang merupakan bagian dari sistem buatan manusia dan berinteraksi dengan kontrol oleh satu atau lebih komputer sebagai bagian dari sistem yang digunakan dalam masyarakat modern. Sistem ini menerima masukan dan menghasilkan keluaran untuk subsistem lainnya, misalnya sistem kebudayaan manusia. Sedangkan sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa campur tangan dari pihak luar. Secara teoritis sistem tersebut ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup).

2.1.2. Basis Data

Dalam pembuatan sebuah aplikasi para pembuat aplikasi atau *programmer* menggunakan basis data atau *database* sebagai dasar dalam mengolah data atau mengelola *file-file*.

A. Definisi Basis Data

Pada umumnya basis data merupakan salah satu aspek yang sangat penting dalam sistem informasi, dimana basis data dijadikan sebagai gudang penyimpanan data yang akan diolah lebih lanjut. Basis data yaitu kumpulan data yang disusun secara sistematis didalam komputer menggunakan perangkat lunak untuk menghasilkan informasi.

Menurut Sukanto dan Shalahuddin (2015:43), “sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan”. Sedangkan, menurut Hirin dan Virgi (2011:29) memberikan batasan bahwa “*database* atau basis data yaitu sekumpulan informasi atau data secara sistematis sehingga dapat diperiksa oleh program komputer untuk memperoleh informasi dari basis data tersebut”.

Dapat ditarik kesimpulan bahwa basis data merupakan kumpulan informasi yang disimpan di dalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut.

B. MySQL

Dalam mengolah basis data (*database*) perangkat lunak yang sering digunakan ialah MySQL, dimana MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional yang mempunyai kompatibel dengan berbagai sistem operasi.

Menurut Ahmar (2013:11) “MySQL adalah sistem yang berguna untuk melakukan proses pengaturan koleksi-koleksi struktur data (*database*) baik meliputi proses pembuatan atau proses pengelolaan *database*”. MySQL merupakan *software database* untuk mengelola dan menyimpan data yang jenisnya beraneka ragam dan tipe data relational yang saling berhubungan (Zaki dan SmitDev Community, 2008:94). Sedangkan menurut Hirin dan Virgi (2011:27) “MySQL adalah satu perangkat lunak sistem manajemen basis data (*database*) SQL atau sering disebut dengan DBMS (*Database Management System*)”.

Penulis menyimpulkan bahwa MySQL merupakan *software* sistem basis data (*database*) yang mempunyai hubungan tipe data relational dalam mengelola dan menyimpan data.

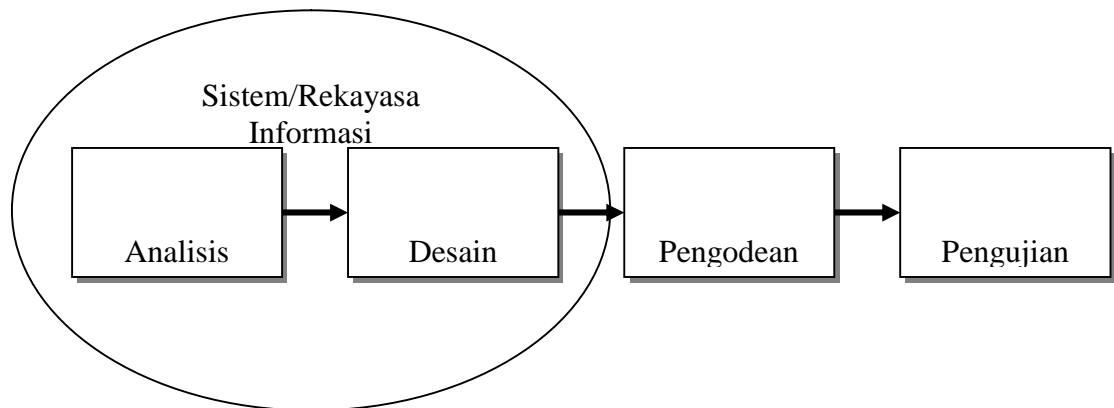
2.1.3. Model Pengembangan Perangkat Lunak

Model pengembangan perangkat lunak digunakan sebagai dasar dalam merancang sebuah sistem informasi atau aplikasi yang bertujuan untuk mengembangkan sistem melalui tahapan-tahapan tertentu. Adapun model pengembangan perangkat lunak yang digunakan penulis yaitu model air terjun (*waterfall*) yang menyediakan pendekatan alur hidup perangkat lunak secara

sekuensial atau terurut dimulai dari analisis, desain, pengodean, dan tahap pendukung (*support*).

Menurut Sukamto dan Shalahuddin (2015:28) mengemukakan bahwa:

Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian dan tahap pendukung (*support*).



Sumber : Sukamto dan Shalahuddin (2015:29)

Gambar II.2. Ilustrasi Model Waterfall

Adapun metode air terjun menurut Sukamto dan Shalahuddin (2015:29) adalah:

1. Analisis Kebutuhan Perangkat Lunak

Proses pengumpulan kebutuhan dilakukan secara insentif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengkodean. Tahap ini

mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. Pembuatan Kode Program

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4. Pengujian

Pengujian fokus pada perangkat lunak secara ad-hoc dari segi *logic* dan fungsional serta memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. Pendukung (*support*) atau Pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.2. Teori Pendukung

Teori pendukung digunakan sebagai alat untuk menggambarkan bentuk logika dari suatu sistem atau aplikasi yang dirancang. Adapun teori pendukung

yang digunakan penulis dalam merancang suatu sistem/aplikasi yang berlandaskan menurut pendapat para ahli dapat dilihat sebagai berikut.

2.2.1. Diagram Alir Data (DAD)

Dalam pembuatan diagram alir data (DAD) para ahli sering menggunakan teknik pemodelan sistem yang berdasarkan pada prosedur-prosedur atau alur dari sebuah sistem. Diagram alir data atau sama dengan *data flow diagram* (DFD) merupakan sebuah prosedur sistem secara keseluruhan dan dapat menguraikan prosedur kedalam bentuk yang lebih rinci.

Menurut Sukamto dan Shalahuddin (2015:70) “DFD tidak sesuai untuk memodelkan sistem yang menggunakan pemrograman berorientasi objek”. Sedangkan menurut Fatta (2009:32) mengemukakan bahwa “*data flow diagram* adalah sebuah teknik grafis yang menggambarkan desain informasi yang diaplikasikan pada saat data bergerak dari *input* menjadi *output*”. Begitu juga menurut McLeod dan Schell (2008:193) mengemukakan bahwa “DFD adalah cara yang sangat alamiah untuk mendokumentasikan proses dan dapat dibuat dalam suatu hierarki untuk menyajikan berbagai tingkat rincian”.

Dari kutipan diatas dapat ditarik kesimpulan bahwa diagram alir data (DAD) atau *data flow diagram* (DFD) merupakan gambaran arus data sistem yang saling berhubungan satu sama lain dengan cara alamiah untuk mendokumentasikan proses dalam menyajikan informasi.

Dalam pembuatan diagram alir data (DAD) Simbol atau lambang yang digunakan (Sukamto dan Shalahuddin, 2015:71) terdiri dari empat buah simbol yaitu:

1. Entitas/Lingkungan Luar (*External Entity*)

Simbol ini digunakan untuk menggambarkan asal atau tujuan data, menunjukkan entitas atau kesatuan yang berhubungan dengan sistem, dapat berupa orang, organisasi, atau sistem lainnya yang akan memberikan input atau menerima input dari sistem atau keduanya digunakan dengan simbol empat persegi panjang.

2. Proses (*Process*)

Simbol ini digunakan untuk proses pengolahan atau transformasi data, menunjukkan kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dan hasil suatu data yang masuk kedalam proses untuk menghasilkan arus data yang akan keluar dari proses, digambarkan dengan simbol lingkaran.

3. Arus Data (*Data Flow*)

Simbol ini digunakan untuk menggambarkan aliran data yang berjalan, menunjukan arus data yang berupa masukan untuk sistem atau hasil dari proses sistem yang mengalir diantara proses (*process*), simpanan data (*data store*) dan entitas (*external entity*) digambarkan dengan arah panah.

4. Simpanan Data (*Data Store*)

Simbol ini digunakan untuk menggambarkan *data flow* yang sudah disimpan, menunjukan suatu tempat penyimpanan data yang dapat berupa suatu file di sistem komputer, arsip atau catatan manual, tabel acuan dan lain-lain digambarkan dengan sepasang garis horizontal.

Tahap dalam pembuatan diagram alir data (Fatta, 2009:32) terdiri dari tiga tingkatan kontruksi yaitu:

1. Diagram Konteks

Diagram ini dibuat untuk menggambarkan sumber serta tujuan data yang akan diproses atau dengan kata lain diagram tersebut untuk menggambarkan sistem secara global dari keseluruhan sistem yang ada.

2. Diagram Nol

Diagram ini dibuat untuk menggambarkan tahap-tahap proses yang akan ada didalam konteks atau penjabaran secara rinci.

3. Diagram Detail

Diagram ini dibuat untuk menggambarkan arus data secara lebih detail dan terperinci dari tahapan proses yang ada dalam diagram.

2.2.2. Kamus Data

Kamus data digunakan sebagai alat komunikasi antara analisis sistem dengan pemakai sistem tentang data yang mengalir di sistem, yaitu tentang data yang masuk ke sistem dan tentang informasi yang dibutuhkan oleh pemakai sistem. Kamus data dibuat bertujuan untuk memberikan informasi kepada para pengembang sistem agar dapat mendesain ulang sistem informasi menjadi spesifikasi *file*.

Menurut McLeod dan Schell (2008:171) menyatakan bahwa kamus data (*data dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Sedangkan menurut Djahir dan Pratita (2015:199) mengemukakan bahwa “kamus data adalah suatu ensiklopedi dari informasi yang berkenaan dengan data organisasi/perusahaan, dan penjelasan ini dikombinasikan kepada komputer

melalui *data description language*-DDL, yang menghasilkan skema”. Sukamto dan Salahuddin (2015:73) mengemukakan bahwa “kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat difahami secara umum (memiliki standar cara penulisan)”.

Kesimpulan menurut para ahli diatas bahwa kamus data adalah kumpulan dari elemen yang tersimpan kedalam basis data yang dikombinasikan melalui perangkat lunak komputer.

Kamus data (Sukamto dan Shalahuddin, 2015:74) berisikan tentang:

1. Nama

Kamus data berisikan nama data yang mengalir di DAD.

2. Digunakan

Kamus data digunakan pada proses-proses terkait aliran data.

3. Deskripsi

Deskripsi disini menguraikan data-data yang mengalir menjadi lebih detail.

4. Informasi tambahan

Kamus data biasa berisikan informasi tambahan seperti tipe data, nilai data, batas nilai data, dan komponen yang membentuk data tersebut.

Kamus data memiliki beberapa simbol untuk menjelaskan informasi (Sukamto dan Shalahuddin, 2015:74) dapat dilihat pada tabel berikut ini.

Tabel II.1. Simbol Kamus Data

Simbol	Keterangan
=	Disusun atau terdiri dari
+	Dan
[]	Baik ... atau
{ } ⁿ	n kali diulang/bernilai banyak
()	Data opsional
...	Batas komentar

Sumber: Sukamto dan Shalahuddin (205:74)

2.2.3. Entity Relationship Diagram (ERD)

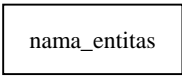
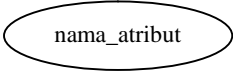
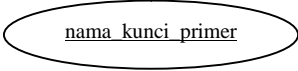
Entity relationship diagram (ERD) merupakan salah satu model untuk menjelaskan hubungan antar data dalam basis data yang mempunyai hubungan antar relasi dan digunakan beberapa notasi dan simbol untuk menggambarkan model struktur data dan hubungan antar data.

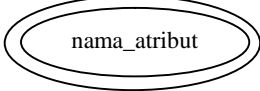
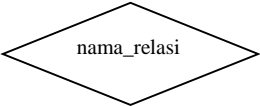
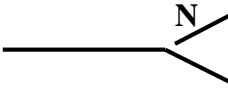
Menurut Sukamto dan Shalahuddin (2015:53) “ERD adalah bentuk paling awal dalam melakukan perancangan basis data relasional. Jika menggunakan OODMBS maka perancangan ERD tidak perlu dilakukan”. Menurut Lubis (2016:31) mengemukakan bahwa “ERD menjadi salah satu pemodelan data konseptual yang paling sering digunakan dalam proses pengembangan basis data bertipe relasional”. Pada dasarnya ERD adalah sebuah diagram konseptual yang memetakan hubungan antar penyimpanan data pada diagram DFD (Wahana Komputer, 2010:30).

Dapat disimpulkan bahwa *entity relationship diagram* (ERD) adalah salah satu bentuk pemodelan basis data dengan diagram konseptual yang mempunyai tipe relasional dalam memetakan hubungan antar data.

Simbol yang terdapat pada *entity relationship diagram* (ERD) sering disebut dengan komponen. Simbol pada *entity relationship diagram* (ERD) menurut Chen dalam buku Sukanto dan Shalahuddin (2015:50) disajikan ke dalam bentuk tabel.

Tabel II.2. Komponen-Komponen *Entity Relationship Diagram* (ERD)

Notasi	Komponen	Keterangan
	Entitas/ <i>entity</i>	Entitas merupakan data inti yang akan disimpan, bakal tabel pada basis data, benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer. Penamaan entitas biasanya lebih ke kata benda dan belum merupakan nama tabel.
	Atribut	<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas.
		<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses <i>record</i> yang diinginkan, biasanya

	Atribut kunci primer	berupa id. Kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat unik (berbeda tanpa ada yang sama).
	Atribut multinilai/ <i>multivalued</i>	<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki lebih dari satu.
	Relasi	Relasi yang menghubungkan antar entitas, biasanya diawali dengan kata kerja.
	Asosiasi/ <i>association</i>	Penghubung antara relasi dan entitas dimana di kedua ujungnya memiliki <i>multiplicity</i> kemungkinan jumlah pemakaian. Kemungkinan jumlah maksimum keterhubungan antara entitas satu dengan entitas yang lain disebut dengan kardinalitas. Misalkan ada kardinalitas 1 ke N atau sering disebut dengan <i>one to many</i> menghubungkan entitas A dan entitas B.

Sumber: Sukamto dan Shalahuddin (2015:50)

2.2.4. *Logical Record Structure (LRS)*

logical record structure (LRS) adalah suatu program untuk merelasikan struktur *record* dari tabel yang memiliki database dengan sebuah sistem yang digambarkan untuk mempermudah logika.

Menurut Iskandar dan Rangkuti (2008:126) mengemukakan bahwa:

LRS terdiri dari *link-link* diantara *tipe record*. *Link* ini menunjukkan arah dari satu *tipe record* lainnya. Banyak *link* dari LRS yang diberi tanda *field-field* yang kelihatan pada kedua *link tipe record*. Penggambaran LRS mulai dengan menggunakan model yang dimengerti.

Hasugian dan Shidiq (2012:608) mengemukakan bahwa LRS adalah “sebuah model sistem yang digambarkan dengan sebuah *diagram-ER* akan mengikuti pola atau aturan permodelan tertentu dalam kaitannya dengan konvensi ke LRS”. LRS merupakan hasil transformasi diagram E-R (ERD) menggunakan aturan aturan tertentu. Aturan-aturan tersebut yaitu: (1) setiap *entity* akan diubah ke dalam bentuk sebuah kotak dengan nama *entity* berada di luar kotak dan atribut berada di dalam kotak, (2) sebuah relasi kadang disatukan dalam sebuah kotak bersama *entity*, kadang dipisah dalam sebuah kotak tersendiri (Ladjamudin, 2013:159).

Kesimpulan *logical record structure* (LRS) adalah sistem yang digambarkan mengikuti aturan permodelan yang merupakan hasil dari transformasi diagram E-R dengan menggunakan aturan-aturan tertentu.

Dalam menguraikan aturan pokok (Ladjamudin, 2013:159) mempengaruhi langkah pentransformasian yaitu kardinalitas. Adapun kardinalitas tersebut (Ladjamudin, 2013:160) yaitu:

1. 1:1 (*one to one*)

Relasi yang terjadi antara suatu *entity* dengan *entity* lainnya yang memiliki hubungan 1:1.

2. 1:M (*one to many*)

Relasi yang terjadi antara suatu *entity* dengan *entity* lainnya yang memiliki hubungan 1:M.

3. M:N (*many to many*)

Relasi yang terjadi antara suatu *entity* dengan *entity* lainnya yang memiliki hubungan M:N. Pada relasi ini biasa digunakan tabel bantuan untuk memecahkan relasi tersebut menjadi 1:1 atau 1:M.

2.2.5. Pengkodean

Pengkodean atau struktur kode yaitu suatu teknik dalam merancang kode program yang telah tersusun dari aturan-aturan yang dibuat dengan berdasarkan elemen-elemen tertentu.

Menurut Jogiyanto (2009:384), “struktur kode adalah suatu bentuk struktur yang berfungsi untuk mengklasifikasikan data, memasukkan data ke dalam komputer dan untuk mengambil bermacam-macam informasi yang berhubungan dengannya”. Sedangkan menurut Shatu (2016:106) mengemukakan bahwa “kode memudahkan proses pengolahan data karena dengan kode, data akan lebih mudah diidentifikasi”.

Beberapa kemungkinan susunan digit (angka), huruf dan karakter-karakter khusus dapat dirancang ke dalam bentuk kode. dalam merancang suatu kode harus diperhatikan beberapa hal (Jogiyanto, 2009:384) yaitu:

1. Mudah diingat

Supaya kode mudah diingat maka dapat dilakukan dengan cara menghubungkan kode tersebut dengan objek yang diwakili dengan kodenya.

2. Unik

Kode harus unik untuk masing-masing item yang diwakilinya.

3. Fleksibel

Kode harus fleksibel sehingga memungkinkan perubahan-perubahan atau penambahan *item* baru dapat tetap diwakili oleh kodenya masing-masing.

4. Efisien

Kode harus sesingkat mungkin, selain mudah diingat juga akan efisien apabila direkam dan disimpan di luar komputer.

5. Konsisten

Kode harus konsisten dengan kode yang telah digunakan.

6. Harus distandarisasi

Kode harus distandarisasi untuk seluruh tingkatan dan departemen dalam organisasi.

7. Hindari penggunaan spasi

Spasi dalam kode sebaiknya dihindarkan karena dapat menyebabkan kesalahan dalam menggunakannya.

8. Hindari karakter yang mirip

Karakter-karakter yang hampir mirip bentuk dan bunyi pengucapannya sebaiknya tidak digunakan dalam kode.

9. Panjang kode harus sama

Masing-masing kode yang sejenis harus memiliki panjang kode yang sama.

Ada beberapa macam tipe dari kode yang dapat digunakan, menurut Jogyanto (2009:386) sebagai berikut:

1. Kode Mnemonik (*Mnemonic Code*)

Kode mnemonik dibuat dengan dasar singkatan atau mengambil sebagian karakter dari *item* yang akan diwakili dengan kode ini.

2. Kode Urut (*Sequential Code*)

Kode yang nilainya urut antara satu kode dengan kode berikutnya.

3. Kode Blok (*Block Code*)

Kode ini mengklasifikasikan *item* ke dalam kelompok blok tertentu yang mencerminkan satu klasifikasi tertentu atas dasar pemakaian maksimum yang diharapkan.

4. Kode Group (*Group Code*)

Kode yang dibentuk berdasarkan *field-field* dan tiap-tiap *field* kode mempunyai arti.

5. Kode Desimal (*Decimal Code*)

Kode ini mengklasifikasikan dalam bentuk sepuluh unit angka desimal dimulai dari angka 0 sampai dengan angka 9 atau dari 00 sampai angka 99 tergantung dari banyaknya kelompok.