BAB 2

LANDASAN TEORI

2.1 Teori-teori Umum

2.1.1 Pengertian Analisis Sistem

Menurut Mcleod (2001, p88), analisis sistem adalah penelitian suatu sistem yang telah ada dengan tujuan untuk merancang sistem baru atau diperbaharui. Ada enam tahap menganalisis sistem:

1. Mengumumkan penelitian sistem.

   Ketika perusahaan menerapkan sistem baru, manajemen bekerja sama dengan pekerja perihal sistem baru tersebut.

2. Mengorganisasikan tim proyek.

3. Mendefinisikan kebutuhan informasi.

   Melalui wawancara perorangan, pengamatan, pencarian catatan dan *survey*.

4. Mendefinisikan kriteria kinerja sistem

   Setelah kebutuhan informasi manajer didefinisikan, langkah selanjutnya adalah menspesifikasi secara tepat apa yang harus dicapai oleh sistem.

5. Menyiapkan usulan rancangan

   Analisis sistem memberikan kesempatan bagi para manajer untuk membuat keputusan terusan atau hentikan untuk kedua kalinya.

6. Menyetujui atau menolak rancangan proyek

   Manajer dan komite pengarah sistem informasi manajemen mengevaluasi usulan rancangan dan menentukan apakah memberi persetujuan atau tidak.

2.1.2 Pengertian Perancangan Sistem

Menurut Mcleod (2001, p238), perancangan sistem adalah penentuan proses dan data yang diperlukan oleh sistem baru, jika sistem itu berbasis komputer, perancangan dapat menyertakan spesifikasi peralatan yang akan digunakan.

Tahap perancangan sistem:

1. Menyiapkan rancangan sistem yang terinci.

   Analis bekerjasama dengan pemakai dan mendokumentasikan rancangan sistem baru dengan alat-alat yang dijelaskan dalam modul teknis.

2. Mengidentifikasikan berbagai alternatif sistem.

   Analis harus mengidentifikasikan konfigurasi peralatan komputer yang akan memberikan hasil terbaik bagi sistem untuk menyelesaikan pemrosesan.

3. Mengevaluasi berbagai alternatif konfigurasi sistem.

   Analis bekerjasama dengan manajer mengevaluasi berbagai alternatif. Alternatif yang dipilih adalah yang paling memungkinkan subsistem memenuhi kriteria kinerja, dengan kendala-kendala yang ada.

4. Memilih konfigurasi terbaik.

   Analis mengevaluasi konfigurasi subsistem dan menyesuaikan dengan kombinasi peralatan sehingga semua subsistem menjadi satu konfigurasi tunggal. Setelah selesai analis membuat rekomendasi kepada manajer untuk disetujui.

5. Menyiapkan usulan penerapan.

   Analis menyiapkan ikhtisar tugas-tugas penerapan yang harus dilakukan.

6. Menyetujui atau menolak penerapan sistem.

   Jika keuntungan yang diharapkan dari sistem melebihi biayanya, penerapan akan disetujui.

Dari pendapat-pendapat tersebut di atas, maka dapat disimpulkan bahwa perancangan sistem merupakan proses penterjemahan kebutuhan pemakai informasi ke dalam suatu rancangan untuk memenuhi kebutuhan pemakai dan memberi gambaran yang lebih jelas untuk dijadikan pertimbangan.

2.1.3 Basis Data

Menurut Connolly dan Begg (2005, p15), basis data adalah sekumpulan dari data – data logikal yang berhubungan dan penjelasan dari data tersebut, yang dibuat untuk memenuhi informasi yang dibutuhkan oleh perusahaan, dan menurut Atzeni, Paolo… [et al], basis data adalah koleksi data yang digunakan untuk merepresentasikan informasi yang menarik ke dalam sistem informasi.

Database merupakan kumpulan *file-file* yang saling berelasi, relasi tersebut biasa ditujukan dengan kunci dari setiap *file* yang ada. Kegunaan dari *database* adalah:

1. Menghilangkan *redundancy* data
2. Keterbatasan akses data
3. Meningkatkan keamanan
4. *Multiple User*
5. Independensi data (kebebasan data)

2.1.4 Sistem Basis Data

Sistem merupakan kumpulan elemen-elemen yang terintegrasi dengan maksud umum untuk mencapai suatu tujuan (Mcleod, 2001, p12), dan menurut C.J. Date (2002, p5), sistem basis data adalah sistem penyimpanan *record* yang terkomputerisasi dimana tujuan sebenarnya adalah untuk menyimpan informasi dan memperbolehkan *user* untuk menelusuri kembali dan mengubah informasi tersebut sesuai kebutuhan.

2.1.5 *Database Management System* (DBMS)

Database Management System (DBMS) adalah sistem piranti lunak yang digunakan untuk membuat, merawat, dan menyediakan kontrol akses untuk pengguna basis data. DBMS menyediakan metode sistematik untuk pembuatan, pengubahan, penyimpanan, dan pengambilan data pada basis data. DBMS juga menyediakan fasilitas untuk mengontrol akses data, memperkuat integritas data, mengatur *concurrency control* dan menyimpan sebuah basis data (Hoffer, 2005, p7).

Menurut Connolly dan Begg (2005, p16) *Database Management System* (DBMS) adalah sebuah sistem *software* yang memungkinkan *user* untuk mendefinisikan, membuat, memelihara, dan mengatur akses ke *database*.

2.1.5.1 Fasilitas-fasilitas DBMS

DBMS berinteraksi dengan program aplikasi *user* dan *database*. DBMS menyediakan fungsi-fungsi sebagai berikut :

1. *Data Definition Language* (DDL)

   Menurut Connolly dan Begg (2005, p40), DDL (*Data Definition Language*) adalah suatu bahasa yang memungkinkan *Database Administrator* atau pengguna untuk mendefinisikan, menerangkan dan memberi nama entiti, atribut dan hubungan yang dibutuhkan untuk aplikasi. DDL berfungsi untuk menjalankan perintah (mengubah suatu data yang dapat berguna bagi pengguna).

   Beberapa *statement* DDL (Connolly dan Begg,2005, p169) :

   a. Create Table

   Untuk membuat tabel dengan mengidentifikasikan tipe data untuk tiap kolom.

   b. Alter Table

   Untuk menambah atau membuang kolom dan constrain.

   c. Drop Table

   Untuk membuang atau menghapus tabel beserta semua data yang terkait di dalamnya.

   d. Create Index

   Untuk membuat index pada suatu tabel.

   e. Drop Index

   Untuk membuang atau menghapus index yang telah dibuat sebelumnya.

2. *Data Manipulation Language* (DML), digunakan untuk menyisipkan, memperbaharui, memanggil data dari basis data.

   Menurut Connolly dan Begg (2005, p40), DML (*Data Manipulation Language*) adalah suatu bahasa yang menyediakan kumpulan operasi yang akan diinginkan untuk mendukung operasi manipulasi data utama pada data yang diperoleh dalam basis data. Menyediakan operasi dasar manipulasi data pada data yang ada dalam basis data, yaitu :

   - Penyisipan data baru ke dalam basis data (*insertion*).
   - Mengubah atau memodifikasi data yang disimpan di dalam basis data (*modify*).
   - Pemanggilan data yang ada dalam basis data (*retrieve*).
   - Menghapus data dari basis data (*delete*).

   Menurut Connolly dan Begg (2005, p41-42), kita dapat membedakan DML menjadi 2 tipe yang berbeda yaitu :

   - Prosedural DML

     Prosedural DML adalah suatu bahasa yang memungkinkan pengguna (umumnya programmer) untuk memberi instruksi ke sistem mengenai data apa yang dibutuhkan dan bagaimana cara pemanggilannya (*retrieve*). Artinya pengguna harus menjelaskan operasi pengaksesan data yang akan digunakan dengan menggunakan prosedur yang ada untuk mendapatkan informasi yang dibutuhkan.

- Non-prosedural DML

  Non-prosedural DML adalah bahasa yang memungkinkan pengguna untuk menentukan data apa yang dibutuhkan dengan menyebutkan spesifikasinya tanpa menspesifikasikan bagaimana cara mendapatkannya.

3. Menyediakan kontrol akses ke basis data dengan menyediakan :
   a. Sistem keamanan yang mencegah akses ilegal ke basis data.
   b. Sistem integritas yang memelihara keakuratan data.
   c. Sistem pengendalian persetujuan yang mengijinkan pembagian akses ke basis data.
   d. Sistem pengendalian pemulihan yang memulihkan basis data ke keadaaan sebelumnya yang dikarenakan oleh kegagalan *software* atau *hardware*.
   e. Katalog pengaksesan *user* yang berisi penjelasan data di basis data.

2.1.5.2 Komponen-komponen DBMS

DBMS memiliki lima komponen yang penting, yaitu :

A. *Hardware*, meliputi *single personal computer*, *single mainframe*, dan jaringan komputer bergantung pada kebutuhan organisasi dan DBMS yang digunakan.

B. *Software*, meliputi *software* DBMS itu sendiri dan program aplikasi, dan sistem operasi, termasuk *software* jaringan jika DBMS sedang digunakan melalui jaringan.

C. Data, merupakan komponen yang paling penting dalam lingkungan DBMS. Data bertindak sebagai jembatan antara komponen mesin dan komponen manusia, yang terdiri dari data operasional dan meta data.

D. *Procedure* (prosedur), mengacu pada instruksi dan peraturan yang mengatur perancangan dan penggunaan basis data bagi pengguna sistem dan staf yang menangani basis data mengenai bagaimana menggunakan atau menjalankan suatu sistem.

E. *People* (manusia), terdiri dari :

1. *Data Administrator* (DA), merupakan seseorang yang bertanggung jawab untuk pengaturan sumber daya data meliputi perencanaan basis data, pengembangan dan pemeliharaan standarisasi, kebijakan dan prosedur, dan perancangan basis data konseptual/logikal.
2. *Database Administrator* (DBA), bertanggung jawab terhadap realisasi fisik dari basis data, meliputi perancangan basis data fisikal dan implementasi, pengaturan integritas dan keamanan, pemeliharaan dari sistem operasional dan memastikan pencapaian yang memuaskan dari aplikasi terhadap user.
3. *Database designers*
   a. *Logical database designer* fokus dengan pengidentifikasian data (entiti dan atribut), hubungan antara data, dan batasan data yang akan disimpan dalam basis data.
   b. *Physical database designer* memutuskan bagaimana perencanaan basis data logikal direalisasikan secara fisik.

4. *Application developers*, bertanggung jawab untuk mengimplementasikan program aplikasi yang menyediakan fungsionalitas yang dibutuhkan oleh *end-user* setelah basis data diimplementasikan.
5. *End Users*, merupakan klien dari basis data yang didesain dan diimplementasikan, dan dipelihara untuk melayani kebutuhan informasi mereka. *End users* dapat dikelompokkan menurut cara mereka menggunakan sistem, yaitu :
    a. *Naive users* adalah *user* yang mengakses basis data melalui program aplikasi spesial yang mencoba untuk membuat operasi sesederhana mungkin sehingga tidak perlu mengetahui segala sesuatu mengenai basis data atau DBMS.
    b. *Sophisticated users* adalah *user* yang lebih mengenal struktur basis data dan fasilitas yang ditawarkan oleh DBMS. Beberapa *sophisticated users* bahkan dapat menulis program aplikasi untuk kepentingan mereka sendiri.

2.1.5.3 Keuntungan dan Kerugian DBMS

DBMS memiliki keuntungan dan kerugian (Connolly dan Begg, 2005, p26-30) Keuntungan DBMS :

1. Kendali terhadap pengulangan data (*data redudancy*).
2. Data yang konsisten.
3. Semakin banyak informasi yang didapat dari data yang sama.
4. Data yang digunakan bersama-sama (*sharing of data*).

5. Meningkatkan integritas data.
6. Meningkatkan keamanan data.
7. Penetapan standard.
8. Penekanan biaya.
9. Mempermudah pengaksesan dan respon data
10. Meningkatkan produktivitas
11. Meningkatkan pengelolaan dengan adanya *data independence*
12. Meningkatkan *concurrency*
13. Meningkatkan layanan *backup* dan *recovery*

Kerugian DBMS :

1. Kompleksitas.
2. *Size* / ukuran besar.
3. Biaya dari suatu DBMS.
4. Biaya penambahan perangkat keras.
5. Biaya konversi.
6. Kinerja.
7. Imbas kegagalan yang besar.

2.1.6 *Structured Query Language* (SQL)

Menurut Connolly dan Begg (2005, p113), SQL merupakan bahasa yang dirancang untuk menggunakan relasi untuk mengubah masukan menjadi keluaran yang diharapkan.

Menurut O’Brien (2003, p148), SQL adalah bahasa query yang ditemukan di berbagai paket manajemen database.

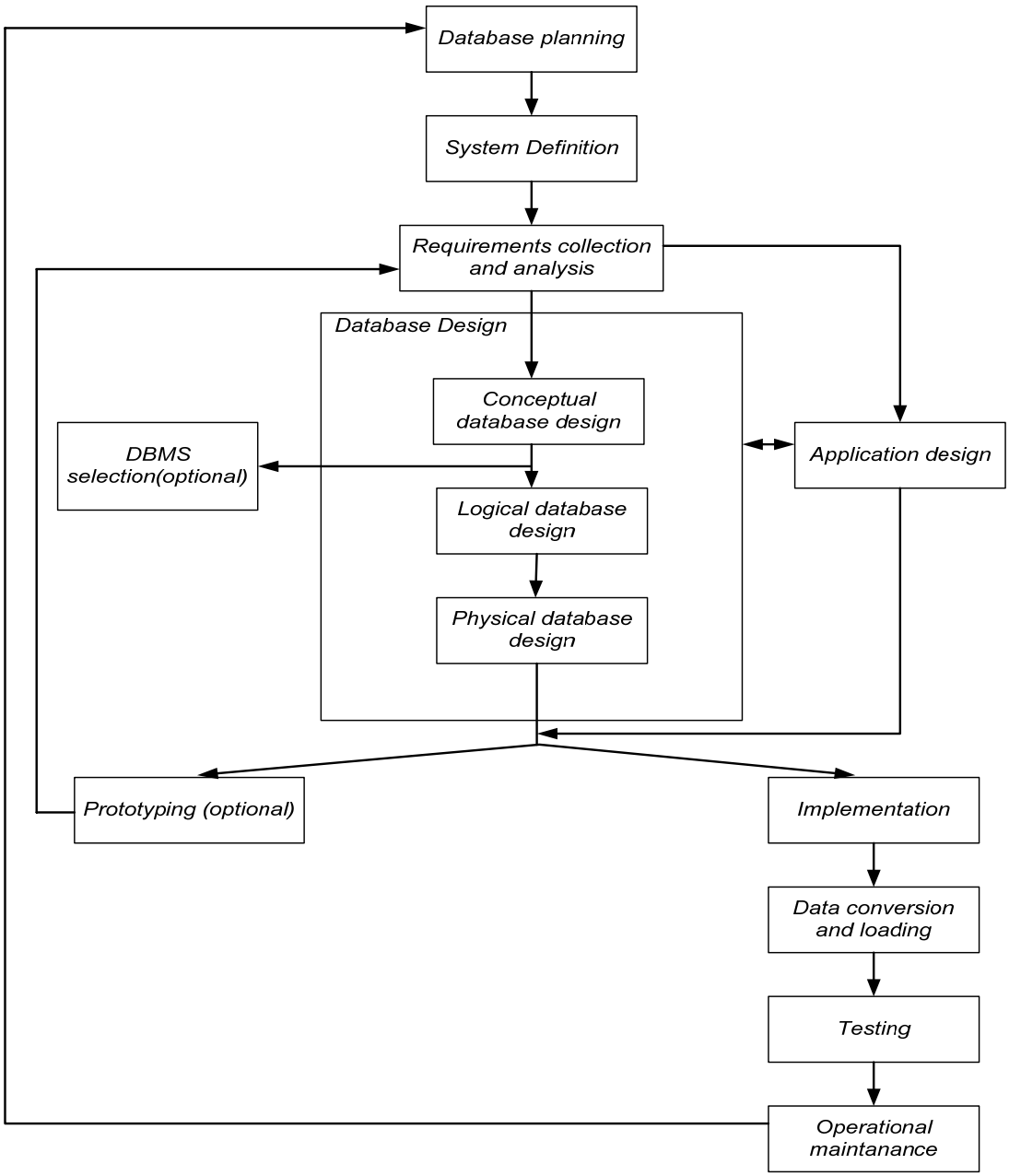
SQL dimaksudkan untuk memenuhi keputusan berikut :

a. Membuat database dan struktur relasi;
b. Melakukan tugas dasar manajemen data, seperti pemasukan, modifikasi dan penghapusan data dari relasi;
c. Melakukan *query* sederhana dan kompleks.

Standar SQL memiliki dua komponen :

a. *Data Definition Language* (DDL) untuk menetapkan struktur database dan mengontrol akses ke data;
b. *Data Manipulation Language* (DML) untuk mendapatkan kembali (*retrieve*) dan memperbaharui data.

2.1.7 Siklus Hidup Pengembangan Sistem Basis Data



Gambar 2.1 *Database Lifecycle* (Sumber: Connolly, 2005, p284)

2.1.7.1 *Database Planning* (Perencanaan Basis Data)

Berdasarkan Connolly dan Begg (2005, p285), perencanaan basis data merupakan aktivitas-aktivitas manajemen yang memungkinkan tahap-tahap dalam aplikasi basis data direalisasikan seefisien dan seefektif mungkin. Perencanaan basis data harus diintegrasikan dengan keseluruhan sistem informasi suatu organisasi.

Ada 3 (tiga) persoalan pokok yang terlibat dalam perumusan suatu strategi sistem informasi:

- Identifikasi rencana, sasaran (*goals*) dan tujuan perusahaan dengan penentuan kebutuhan sistem informasi.
- Evaluasi sistem informasi yang sedang berjalan untuk menentukan kelebihan dan kekurangan yang ada.
- Penilaian terhadap peluang IT apakah mampu menghasilkan keuntungan yang kompetitif.

Tahapan ini bertujuan untuk mendefinisikan *mission statement* dan *mission objective* :

- ***Mission statement.***

  Mission statement ini menjelaskan tujuan utama aplikasi basis data, juga membantu menjelaskan tujuan proyek basis data, dan menyediakan alur yang lebih jelas dalam pembuatan aplikasi basis data yang dibutuhkan secara efektif dan efisien.

- ***Mission objectives.***

  Setiap *Mission objective* akan menjelaskan suatu tugas tertentu yang harus didukung oleh basis data, dengan asumsi jika basis data mendukung *mission objective*, maka *mission statement*nya juga akan sesuai.

2.1.7.2 *System Definition* (Definisi Sistem)

System definition menggambarkan ruang lingkup dan batasan dari aplikasi basis data dan pandangan pengguna (*user view*) yang utama (Connolly dan Begg, 2005, p286). Sebelum merancang aplikasi basis data, maka harus mengidentifikasikan batasan dari sistem dan bagaimana sistem tersebut berinteraksi dengan bagian lain dari informasi perusahaan.

User view menggambarkan apa yang dibutuhkan oleh aplikasi basis data dari sudut pandang jabatan tertentu, seperti manajer atau pengawas, maupun dari sudut pandang area aplikasi perusahaan, seperti pemasaran, personalia, atau pengawasan persediaan, dalam hubungannya dengan data yang akan disimpan dan transaksi yang akan dijalankan terhadap data itu (Connolly dan Begg, 2005, p287).

2.1.7.3 *Requirement Collection and Analysis* (Pengumpulan dan Analisis Kebutuhan)

Pengumpulan dan analisis kebutuhan adalah proses pengumpulan dan analisis informasi tentang bagian perusahaan yang didukung oleh aplikasi basis data dan yang menggunakan informasi ini untuk mengidentifikasikan

kebutuhan – kebutuhan *user* dari sistem yang baru (Connolly dan Begg, 2005, p286 – p288).

Pengumpulan kebutuhan atau *fact-finding* memiliki 5 jenis teknik yang penggunaannya sesuai kebutuhan :

1) Memeriksa dokumentasi dan formulir

   Pemahaman terhadap jalannya sistem akan cepat diperoleh dengan memeriksa dokumen-dokumen, formulir, laporan, dan file yang berhubungan dengan sistem yang sedang berjalan.

2) Wawancara

   Bertujuan untuk mengumpulkan fakta-fakta, memeriksa kebenaran fakta yang ada dan mengklarifikasinya, membangkitkan semangat, melibatkan pengguna akhir, mengidentifikasi kebutuhan-kebutuhan, dan mengumpulkan ide-ide dan pendapat (Connolly dan Begg, 2005, p317). Teknik ini memerlukan kemampuan komunikasi yang baik untuk menghadapi pengguna yang memiliki nilai, prioritas, pendapat, motivasi, dan kepribadian yang beragam.

   Keuntungan menggunakan teknik ini menurut Connolly dan Begg (2005, p318) antara lain :

   - Memungkinkan orang yang diwawancara untuk menanggapi pertanyaan dengan bebas dan terbuka
   - Memungkinkan orang yang diwawancara merasa bahwa ia merupakan bagian dari proyek

- Memungkinkan pewawancara untuk menindaklanjuti komentar-komentar menarik yang dibuat oleh orang yang diwawancara
- Memungkinkan pewawancara untuk mengubah atau menyusun kembali pertanyaan selama kegiatan wawancara
- Memungkinkan pewawancara untuk mengamati bahasa tubuh orang yang diwawancara

Kerugian teknik ini (Connolly dan Begg, 2005, p318) yaitu :

- Sangat memakan waktu dan biaya sehingga menjadi tidak praktis
- Keberhasilannya tergantung pada kemampuan komunikasi pewawancara
- Keberhasilannya tergantung pada keinginan orang yang diwawancara untuk ikut serta dalam wawancara

3) Mengamati Operasional Perusahaan

Memungkinkan untuk ikut serta atau mengamati seseorang dalam melakukan kegiatan untuk mempelajari sistem. Salah satu faktor pengamatan dapat berhasil adalah dengan mencari informasi sebanyak mungkin tentang aktivitas yang akan diamati serta orang yang melakukan aktivitas tersebut.

Keuntungan menggunakan teknik ini antara lain :

- Validasi fakta dan data dapat diperiksa
- Pengamat dapat melihat dengan jelas apa yang dikerjakan
- Pengamat juga dapat memperoleh data yang menjelaskan lingkungan fisik dari tugas yang diberikan

- Pengamat dapat membuat pengukuran kerja

Kerugian teknik ini yaitu :

- Sangat memakan waktu dan biaya, sehingga menjadi tidak praktis
- Beberapa tugas tidak selalu dilakukan dengan cara seperti pada saat pengamatan
- Dapat terlewatkan dalam mengamati tugas – tugas yang melibatkan tingkat kesulitan yang lain

4) Penelitian

Jurnal komputer, buku – buku referensi, dan internet merupakan sumber informasi yang baik yang menyediakan informasi bagaimana orang lain memecahkan masalah yang serupa.

Keuntungan menggunakan teknik ini antara lain :

- Dapat menghemat waktu jika solusinya telah tersedia
- Peneliti dapat mengamati cara orang lain memecahkan masalah yang sama atau menemui kebutuhan serupa
- Membuat para peneliti selalu *up-to-date* dengan perkembangan baru

Kerugian teknik ini yaitu :

- Membutuhkan akses ke sumber informasi yang tepat
- Dapat saja tidak membantu memecahkan masalah karena tidak didokumentasikan

5) Kuesioner

Kuesioner adalah suatu dokumen dengan tujuan khusus yang memungkinkan fakta – fakta dikumpulkan dari banyak orang sambil

menjaga kontrol terhadap tanggapan yang diberikan (Connolly dan Begg, 2002, p320).

Keuntungan menggunakan teknik ini antara lain :

- Orang dapat melengkapi dan mengembalikan kuesioner pada waktu yang sebaik-baiknya
- Tidak mahal untuk mengumpulkan data dari banyak orang
- Responden lebih mudah untuk memberikan jawaban yang benar, karena jawaban yang diberikan dapat dijaga kerahasiaannya
- Tanggapan dapat ditabulasikan dan dianalisa dengan cepat

Kerugian teknik ini yaitu :

- Jumlah responden dapat saja rendah, sekitar 5% sampai 10%
- Kuesioner dapat saja dikembalikan dengan tidak lengkap
- Tidak menyediakan kesempatan untuk mengubah pertanyaan yang salah diartikan
- Tidak dapat mengamati dan menganalisa bahasa tubuh responden
- Memakan waktu untuk menyiapkan kuesioner

2.1.7.4 *Database Design* (Perancangan Basis Data)

Perancangan basis data adalah suatu proses menciptakan perancangan untuk basis data yang akan mendukung keseluruhan operasi dan tujuan – tujuan perusahaan (Connolly dan Begg, 2005, p291).

Terdapat dua pendekatan yang dapat digunakan untuk merancang basis data, yaitu :

1. *Bottom up Approach*

   Pendekatan *bottom up* dimulai dari atribut awal (*entity* dan *relationship*) yang dianalisa asosiasi antar atribut, kemudian dibentuk *relation* yang mewakili tipe dari *entity* dan *relationship* antar *entity*. Pendekatan ini sesuai untuk perancangan basis data yang sederhana dengan jumlah atribut sedikit.

2. *Top down Approach*

   Pendekatan *Top down* dimulai dengan pengembangan *data model* yang terdiri dari sedikit atau banyak *entity* dan *relationship*, kemudian melakukan perbaikan *top down* untuk mengidentifikasikan *lower-level entity*, *relationship*, dan asosiasi antar atribut. Pendekatan ini digambarkan dengan *entity relationship* (ER) *model*, yang dimulai dari identifikasi *entity* dan *relationship* antar *entity*.

Dalam perancangan basis data terdapat tiga tahapan yang diperlukan agar mendapatkan sebuah *database* yang diinginkan, menurut Connoly (2005, p293), setiap konsep tersebut memiliki langkah yang perlu ditempuh antara lain :

1. **Conceptual Database Design**, merupakan proses membangun sebuah model informasi yang digunakan dalam sebuah perusahaan, bebas dari semua pertimbangan fisik.

   Langkah 1 Membangun model data konseptual lokal untuk setiap *view*, bertujuan untuk membangun sebuah model data konseptual lokal dari sebuah perusahaan untuk setiap *view* yang spesifik.

Langkah 1.1 Mengidentifikasi tipe entiti, bertujuan untuk mengidentifikasi tipe-tipe entiti yang utama yang dibutuhkan pada *view*

Langkah 1.2 Mengidentifikasi tipe relasi, bertujuan untuk mengidentifikasi relasi penting yang terdapat di antara tipe-tipe entiti yang telah teridentifikasi.

Langkah 1.3 Identifikasi dan menghubungkan atribut sesuai dengan tipe entiti dan relasi, bertujuan untuk menghubungkan atribut dengan tipe-tipe entiti atau relasi yang sesuai.

Langkah 1.4 Menentukan domain atribut, bertujuan untuk menentukan domain untuk atribut dalam model data konseptual lokal.

Langkah 1.5 Menentukan atribut *candicate* dan *primary key*, bertujuan untuk mengidentifikasikan *candidate key* untuk setiap tipe entiti dan, jika terdapat lebih dari satu *candidate key*, memilih satu untuk dijadikan *primary key*.

Langkah 1.6 Mempertimbangkan penggunaan konsep permodelan *enhanced* (opsional), bertujuan untuk mempertimbangkan penggunaan konsep permodelan *enhanced*, seperti spesialisasi/generalisasi, agregasi dan komposisi

Langkah 1.7 Memeriksa redundansi pada model, bertujuan untuk memeriksa adanya redundansi pada model.

Langkah 1.8 Validasikan model konseptual lokal pada transaksi *user*, bertujuan untuk memastikan bahwa model konseptual lokal mendukung transaksi yang dibutuhkan oleh *view*.

Langkah 1.9 Meninjau model data konseptual lokal dengan *user*, bertujuan untuk meninjau model data konseptualdata model dengan *user* untuk memastikan bahwa model sudah mewakili *view*.

2. **Logical Database Design**, merupakan proses pembangunan model informasi yang digunakan dalam perusahaan berdasarkan sebuah model data spesifik, tetapi bebas dari DBMS tertentu dan pertimbangan fisik lainnya. (Connoly dan Begg, 2005, p294)

   Langkah 2 Membangun dan memvalidasikan model data logikal untuk setiap *view*, bertujuan untuk membangun sebuah model data logikal lokal dari sebuah model data konseptual lokal yang mewakili sebuah *view* tertenti dari perusahaan dan kemudian untuk memvalidasi model ini untuk memastikan secara struktur model ini benar (menggunakan teknik normalisasi) dan memastikan model ini mendukung transaksi yang dibutuhkan.

   Langkah 2.1 Menghilangkan fitur yang tidak kompetibel dengan model relasional (opsional), bertujuan untuk menyempurnakan model data konseptual lokal dengan menghilangkan fitur yang tidak kompetibel dengan model relasional. Fitur yang tidak kompatibel tersebut antara lain :

   1. Tipe relasi *many-to-many* (*:*) *binary*
   2. Tipe relasi *many-to-many* (*:*) rekrusif
   3. Tipe relasi komplek
   4. Atribut *multi-valued*

Langkah 2.2 Mendapatkan relasi untuk model data logikal lokal, bertujuan untuk membuat relasi model data logikal lokal yang menampilkan entiti, relasi dan atribut yang telah diidentifikasikan.

Langkah 2.3 Memvalidasikan relasi dengan menggunakan normalisasi, bertujuan untuk memvalidasikan relasi dalam model data logikal lokal menggunakan teknik normalisasi.

Langkah 2.4 Memvalidasikan relasi pada transaksi *user*, bertujuan untuk memastikan bahwa relasi dalam model data logikal lokal mendukung transaksi yang dibutuhkan dalam *view*.

Langkah 2.5 Menetapkan integritas *constraints*, bertujuan utnuk membatasi integritas *constraints* yang terdapat dalam *view*.

Langkah 2.6 Meninjau model data logikal dengan *user*, bertujuan untuk memastikan bahwa model data logikal lokal dan dokumentasi pendukung yang menjelaskan model telah mewakili *view*.

Langkah 3 Membangun dan memvalidasikan model data logikal global, bertujuan untuk menggabungkan masing-masing model data logikal lokal menjadi sebuah model data logikal global yang menggambarkan perusahaan.

Langkah 3.1 Menggabungkan model data logikal lokal menjadi model global, bertujuan untuk menggabungkan model data logikal lokal yang ada menjadi sebuah model data logikal global perusahaan. Beberapa hal yang dilakukan dalam langkah ini antara lain :

1. Membahas nama dan isi dari entiti atau relasi dan *candidate key* masing-masing.

2. Membahas nama dan isi dari relasi atau *foreign key*.
3. Menggabungkan entiti atau relasi dari model data lokal.
4. Tambahkan (tanpa menggabungkan) entiti/relasi yang unik ke setiap model data lokal.
5. Menggabungkan relasi/*foreign key* yang unik ke setiap model data lokal.
6. Tambahkan (tanpa menggabungkan) rekasi/*foreign key* ke setiap model data lokal.
7. Memeriksa entiti/relasi dan relasi/*foreign key* yang hilang.
8. Memeriksa *foreign key*.
9. Memeriksa integritas *constraints*.
10. Menggambar diagram relasi/ ERD global.
11. Memperbarui dokumentasi.

Langkah 3.2 Memvalidasi model data logikal global, bertujuan untuk memvalidasikan relasi yang dibentuk dari model data lokal logikal global menggunakan teknik normalisasi dan memastikan model data tersebut mendukung transaksi yang dibutuhkan, bila perlu.

Langkah 3.2 Memeriksa kemungkinan perkembangan, bertujuan untuk memastikan apakah ada perubahan yang signifikan pada suatu saat nanti dan memperkirakan apakah model data logikal global dapat mengakomodasi perubahan ini.

Langkah 3.3 Meninjau model data logikal global dengan para *users*, bertujuan untuk memastikan bahwa model data logikal global ini mewakili perusahaan.

3. **Physical Database Design**, merupakan proses menghasilkan sebuah deskripsi dari implementasi database di *secondary storage*; yang mendeskripsikan relasi dasar, *file* organisasi, dan index yang digunakan agar dapat mengakses data secara efisien, dan semua hubungan integritas *constraints* dan tingkat keamanan. (Connoly dan Begg, 2005, p294)

   Langkah 4 Menerjemahkan model data logikal global ke dalam DBMS tujuan, bertujuan untuk menghasilkan sebuah skema relasi *database* dari model data logikal global yang diimplementasikan di DBMS tujuan.

   Langkah 4.1 Mendesain relasi dasar, bertujuan untuk menentukan bagaimana menampilkan relasi dasar yang diidentifikasikan dalam model data logikal global di dalam DBMS tujuan.

   Langkah 4.2 Mendesain berdasarkan data yang diperoleh, bertujuan untuk menentukan bagaimana menampilkan data yang diperoleh di model data logikal global dalam target DBMS.

   Langkah 4.3 Mendesain batasan *constraints*, bertujuan untuk mendesain batasan *constraints* untuk DBMS tujuan.

   Langkah 5 Mendesain *physical representation*, bertujuan untuk menentukan organisasi *file* yang optimal untuk disimpan dalam relasi dasar dan index yang dibutuhkan untuk menghasilkan performa yang baik, yang mana relasi dan *tuples* akan disimpan dalam *secondary storage*.

Langkah 5.1 Menganalisa transaksi, bertujuan untuk memahami fungsionalitas transaksi yang akan dijalankan pada *database* dan untuk menganalisa transaksi penting.

Langkah 5.2 Memilih *file* organisasi, bertujuan untuk menentukan *file* organisasi untuk setiap relasi dasar.

Langkah 5.3 Memilih index, bertujuan untuk menentukan apakah penambahan index akan meningkatkan performa sistem.

Langkah 5.4 Memperkirakan kebutuhan *disk space*, bertujuan untuk memperkirakan kapasitas *disk space* yang akan dibutuhkan *database*.

Langkah 6 Mendesain *user view*, bertujuan untuk mendesain *user view* yang teridentifikasi selama pengumpulan kebutuhan dan analisis pada bagian siklus hidup aplikasi relasional *database*.

Langkah 7 Mendesain mekanisme keamanan, bertujuan untuk mendesain tingkat keamanan untuk database secara terspesifikasi bagi *users*.

Langkah 8 Mempertimbangkan pengenalan redundansi terkontrol.

Langkah 9 Memantau dan menyetel sistem operasional.

2.1.7.5 DBMS *Selection* (Pemilihan DBMS)

Pemilihan DBMS yang tepat untuk mendukung aplikasi basis data. Bagaimanapun pemilihan DBMS bisa dilakukan pada setiap waktu sebelum melakukan desain logical yang menyajikan informasi cukup mengenai kebutuhan sistem seperti *performance*, *security*, *integrity constraint*. Walaupun pemilihan DBMS mungkin jarang tetapi ketika kebutuhan

perusahaan sedang diperluas atau sistem yang berjalan digantikan, mungkin menjadi perlu kadang-kadang untuk mengevaluasi produk DBMS yang baru. Suatu penelitian sederhana dalam melakukan pemilihan DBMS adalah dengan mencocokkan DBMS sesuai kebutuhan. Tahap-tahap utama untuk memilih DBMS :

1. Mendefinisikan terminology studi referensi
2. Mendaftar dua atau tiga produk
3. Evaluasi produk
4. Rekomendasi pilihan dan laporan produk

2.1.7.6 *Application Design* (Desain Aplikasi)

Merupakan perancangan user interface dan program aplikasi yang menggunakan dan memproses basis data. Ada 2 (dua) aspek penting dalam perancangan aplikasi, yakni:

- *Transaction Design* (Perancangan Transaksi)

  Transaksi merupakan sebuah aksi, atau serangkaian aksi yang dilakukan oleh seorang pengguna atau program aplikasi yang mengakses atau mengubah isi dari basis data.

  Tujuan dari perancangan transaksi adalah untuk menetapkan dan mendokumentasikan karakteristik tingkat tinggi dari transaksi yang dibutuhkan pada basis data, yang termasuk:

  1. Data yang digunakan dalam transaksi
  2. Karakteristik fungsional dari transaksi
  3. Keluaran (*output*) dari transaksi

4. Kepentingan pengguna
5. Nilai yang diharapkan dari pemakaian

Perancangan ini harus dilakukan lebih awal dalam proses perancangan untuk memastikan bahwa basis data yang diimplementasikan mampu mendukung semua transaksi yang dibutuhkan. Ada 3 (tiga) jenis transaksi, yaitu:

1. *Retrival transactions*, digunakan untuk mendapatkan kembali data untuk ditampilkan di layar atau dalam laporan.
2. *Update transactions*, digunakan untuk menambah data, menghapus data lama, atau memodifikasi data yang ada dalam basis data.
3. *Mixed Transactions*, melibatkan *retrieval* (pemanggilan) dan *update* (perubahan) data atau kombinasi antara keduanya.

- *User Interface Design* (Perancangan Antarmuka)

  Interaksi Manusia dan Komputer menurut Schneiderman (1998, pp 74-75), delapan aturan dalam membuat antarmuka adalah :

  1. Usahakan untuk selalu konsisten

     Aplikasi yang dirancang harus konsisten, baik dalam tampilan, susunan menu, teks dan warna

  2. Memungkinkan pemakai untuk menggunakan *shortcut*

     Aplikasi yang dirancang sebaiknya mempunyai fasilitas *shortcut* bagi *user* untuk lebih menjelajahi aplikasi

  3. Memberikan umpan balik yang informatif

Sebuah aplikasi harus dapat memberikan navigasi ataupun informasi mengenai tujuan dari sebuah *link*, sehingga akan dapat mengurangi kesalahan yang mungkin dilakukan *user*

4. Merancang untuk menghasilkan keadaan akhir

   Aksi – aksi yang ada seharusnya diorganisasikan untuk mempunyai suatu permulaan, pertengahan, dan tahap akhir, seperti sebuah cerita pendek yang bagus. Dengan adanya umpan balik yang informatif pada tahap akhir dari suatu *form* akan memberitahukan *user* sehingga mereka dapat mengetahui kapan mereka dapat berpindah ke *form* berikutnya.

5. Memberikan penanganan kesalahan yang sederhana

   Jika terjadi kesalahan, maka aplikasi mampu memberikan petunjuk sederhana dan praktis dalam menanganinya. Misalnya disediakan fasilitas *help*

6. Mengizinkan pembalikan aksi dengan mudah

   Aplikasi harus menyediakan fasilitas bagi *user* ntuk kembali ke menu dengan mudah.

7. Mendukung pemakai menguasai sistem (*internal locus of system*)

   Memberikan user memutuskan apa yang diperlukan dan kemudian sistem menyediakan informasi yang dibutuhkan.

8. Mengurangi ingatan jangka pendek (*rule of thumb*)

   Aplikasi harus memudahkan user dalam mengingat hal – hal penting. Misalnya dengan kombinasi kode – kode, maka kode

tersebut diusahakan mudah diingat dengan kemampuan berpikir manusia. Kode jangan terlalu panjang.

Sebelum mengimplementasikan suatu *form* atau laporan, ada perlunya merancang *layout* (tampilan) terlebih dahulu. Berikut pedoman yang berguna dalam perancangan laporan:

1. Judul yang berarti, diusahakan pemberian nama suatu *form* cukup jelas menerangkan kegunaan dari suatu *form* / laporan. Informasi yang disampaikan sebuah judul harus jelas dan terhindar dari ambigu.
2. Instruksi yang dapat dipahami, menggunakan terminologi yang lazim dalam menyampaikan instruksi kepada pengguna. Instruksi harus diuraikan dengan singkat dan jelas, dan ketika membutuhkan informasi lebih lanjut, layar bantuan harus tersedia. Instruksi harus ditulis dengan tata bahasa yang baku dengan menggunakan pola standar.
3. Pengelompokan logik dan pengurutan *field*, *field* yang berhubungan harus ditempatkan secara bersama dalam suatu *form* / laporan. Pengurutan *field* harus logis dan konsisten.
4. Tampilan permohonan *layout form* / laporan secara visual, *form* / laporan harus menarik perhatian bagi pengguna. Hindari area *form* yang kosong terlalu banyak atau field yang berlebihan.
5. Nama field yang lazim, agar tidak menimbulkan kebingungan bagi pengguna.
6. Terminologi dan singkatan yang digunakan harus konsisten.
7. Penggunaan warna secara konsisten dan berarti.

8. Ruang yang tampak dan batasan untuk *field* pemasukan data, seorang pengguna harus secara *visual* menyadari jumlah ruang yang tersedia untuk setiap *field*.
9. Pergerakan kursor yang baik, seorang pengguna harus dengan mudah mengenal operasi yang dibutuhkan untuk menggerakkan kursor di seluruh *form* / laporan.
10. Perbaikan kesalahan untuk karakter individual dan *field* secara keseluruhan.
11. Pesan kesalahan untuk nilai yang tidak dapat diterima sistem.
12. *Field-field* yang bersifat pilihan harus ditandai dengan jelas.
13. Pesan-pesan untuk *field* yang bersifat menjelaskan.

    Ketika suatu pengguna menempatkan kursor pada suatu *field*, informasi tentang *field* tersebut seharusnya muncul dalam suatu posisi yang teratur pada layar.
14. Tanda Penyelesaian

    Indikator yang menjelaskan bahwa suatu proses telah selesai dilaksanakan.Begitu juga ketika proses pengisian dalam field pada suatu *form* lengkap.

2.1.7.7 *Prototyping* (Prototipe)

Dalam tahapan prototyping dilakukan pembuatan suatu model kerja dari aplikasi basis data. Suatu *prototype* adalah model yang bekerja yang tidak mempunyai semua fitur-fitur yang diperlukan atau menyediakan

semua fungsionaliti dari sistem terakhir. Tujuan utama dari pengembangan suatu aplikasi basis data *prototype* yaitu :

1. Untuk mengidentifikasi fitur dari sistem yang berjalan dengan baik atau tidak.
2. Untuk memberikan perbaikan-perbaikan atau penambahan fitur baru
3. Untuk klarifikasi kebutuhan *user.*
4. Untuk evaluasi feasibilitas (kemungkinan yang akan terjadi) dari desain sistem khusus.

Terdapat 2 macam strategi *prototyping* yang digunakan saat ini, yaitu :

1. *Requirements prototyping*, menggunakan suatu *prototype* untuk menentukan kebutuhan-kebutuhan dari aplikasi basis data yang diusulkan dan suatu waktu kebutuhan-kebutuhan tersebut lengkap *prototype* dibuang.
2. *Evolutionary prototyping*, digunakan untuk tujuan yang sama, perbedaan yang penting adalah bahwa *prototype* tidak dibuang tetapi dengan perkembangan yang lebih jauh menjadi aplikasi basis data yang digunakan.

2.1.7.8 *Implementation* (Implementasi)

Implementasi merupakan realisasi fisik dari perancangan basis data dan aplikasi. Pada penyelesaian tingkat-tingkat perancangan (dimana mungkin atau tidak melibatkan *prototyping*), sekarang kita dalam posisi mengimplementasi basis data dan program aplikasi. Implementasi basis data dicapai dengan menggunakan *Data Definition Language* (DDL) dari

DBMS yang dipilih atau *Graphical User Interface* (GUI), dimana menyediakan fungsionaliti yang sama ketika menyembunyikan pernyataan DDL tingkat-rendah.

Pernyataan DDL tersebut digunakan untuk membuat struktur basis data dan file basis data kosong. Program aplikasi diimplementasikan dengan menggunakan bahasa generasi ketiga atau keempat (3GL atau 4GL). Bagian dari program aplikasi ini adalah transaksi basis data, dimana diimplementasikan dengan menggunakan *Data Manipulation Language* (DML) dari DBMS sasaran, yang mungkin disimpan dalam sekumpulan bahasa pemograman, seperti *Visual Basic*. Juga mengimplementasikan komponen-komponen lainnya dari perancangan aplikasi seperti layar menu, *form* pemasukan data, dan laporan.

2.1.7.9 *Data Conversion and Loading* (Konversi dan Pemuatan Data)

Data Conversion and Loading adalah suatu proses menstransfer data yang ada ke dalam basis data baru dan mengubah aplikasi yang ada untuk dijalankan dalam basis data baru (Connolly dan Begg, 2005, p305). Tahap ini hanya dibutuhkan ketika sistem basis data baru menggantikan sistem yang lama.

2.1.7.10 *Testing* (Pengujian)

Testing (pengujian) adalah suatu proses eksekusi program aplikasi dengan tujuan untuk menemukan kesalahan. Sebelum diterapkan dalam suatu sistem, basis data harus dilakukan pengujian terlebih dahulu.

2.1.7.11 *Operational Maintenance* (Pemeliharaan Operasional)

Merupakan proses pengawasan dan pertahanan sistem berikut instalasi. Pada langkah sebelumnya, aplikasi basis data telah diimplementasikan dan diuji sepenuhnya. Sekarang sistem memasuki langkah perawatan, yang melibatkan aktivitas-aktivitas berikut:

1. Pengawasan performa sistem, jika performa menurun maka memerlukan perbaikan atau pengaturan ulang basis data.
2. Pemeliharaan dan pembaharuan aplikasi basis data. Kebutuhan baru disertakan dalam aplikasi basis data melalui tahapan sebelumnya dari siklus hidup.

2.1.8 Normalisasi

Menurut Connolly dan Begg (2005, p388), normalisasi adalah suatu teknik untuk menghasilkan sekumpulan relasi dengan sifat-sifat yang diinginkan, memenuhi kebutuhan data pada perusahaan. Proses normalisasi, yaitu :

1. Suatu teknik formal untuk menganalisa relasi berdasarkan *primary key* dan *functional dependencies* antar atribut.
2. Dieksekusi dalam beberapa langkah. Setiap langkah mengacu ke bentuk normal tertentu sesuai dengan sifat yang dimilikinya.
3. Setelah normalisasi diproses, relasi menjadi secara bertahap lebih terbatas/kuat bentuk formatnya dan juga mengurangi tindakan update yang anomali.

Berikut proses normalisasi

1. ***Unnormalized Form* (UNF)**

   UNF merupakan suatu tabel yang berisikan satu atau lebih grup yang berulang (*repeating-groups*) (Connolly dan Begg, 2005, P403).

2. ***First Normal Form* (1NF)**

   Menurut Connolly dan Begg (2005,p403), aturan normalisasi pertama (1NF) dapat dikatakan bahwa sebuah relasi dimana setiap baris dan kolom hanya berisi satu nilai.

   Suatu data dikatakan *unormalized* (UNF), jika didalamnya mengandung kelompok yang berulang (*repeating group*), sehingga untuk membentuk normalisasi pertama *repeating group* harus dihilangkan. Nilai dari setiap atribut adalah tunggal. Kondisi ini dapat diperoleh dengan melakukan eliminasi terjadinya data ganda (*repeating group*). Namun pada kondisi pertama ini kemungkinan masih terjadi adanya data rangkap.

3. ***Second Normal Form* (2NF)**

   Menurut Connolly dan Begg (2005,p407), aturan normalisasi kedua (2NF) dapat dikatakan bahwa sebuah relasi dalam bentuk normal pertama dan setiap atribut bukan primary key yang tergantung secara fungsional kepada *primary key*.

   Pengujian bentuk normal kedua dapat dihasilkan dengan melihat apakah ada atribut bukan *primary key* yang merupakan fungsi dari sebagian *primary key* (*partial dependency*).

4. ***Third Normal Form* (3NF)**

Menurut Connolly dan Begg (2005, p408) aturan normalisasi ketiga (3NF) adalah sebuah relasi dalam bentuk normal pertama dan kedua dan setiap atribut yang bukan *primary key* yang bergantung secara transitif kepada *primary key*.

Pengujian terhadap 3NF dilakukan dengan cara melihat apakah terdapat atribut yang bukan *key* tergantung fungsional terhadap atribut bukan *key* lainnya (disebut ketergantungan transitif atau *transitive dependence*). Dengan cara yang sama, maka setiap ketergantungan transitif dipisahkan. 3NF sudah cukup baik dalam arti anomali (data yang berulang) yang dikandungnya sudah sedemikian minimum.

5. ***Boyce-Codd Normal Form* (BCNF)**

Menurut Connolly dan Begg (2005, p419) aturan normalisasi *boyce-codd* (BCNF) adalah sebuah relasi jika dan hanya jika setiap determinan adalah *candidate key*.

Pengujian terhadap BCNF dilakukan dengan cara mengidentifikasi semua determinan dan memastikan bahwa semuanya adalah *candidate key*. Determinan adalah sebuah atau sekelompok atribut dimana beberapa atribut lain bergantung terhadapnya.

6. ***Fourth Normal Form* (4NF)**

4NF adalah sebuah relasi yang terdapat dalam BCNF dan tidak memiliki ketergantungan *multi-valued* yang berarti (Connolly dan Begg, 2005, p 428).

7. ***Fifth Normal Form* (5NF)**

5NF adalah sebuah relasi yang tidak memiliki *join dependency* (Connolly dan Begg, 2005, p430). Normalisasi kelima (atau yang juga dikenal juga dengan *project-join normal form* (PJNF)) memastikan bahwa 5NF tidak memiliki *join dependency*.

2.1.9 Keamanan

Menurut Connolly dan Begg (2005, p542), "*Database security* adalah mekanisme untuk memproteksi *basis data* melawan ancaman dari luar maupun dalam". *Database security* mencakup *hardware*, *software*, *user*, dan data. Untuk menjalankan sistem keamanan secara efektif, diperlukan beberapa kontrol yang sesuai, yang didefinisikan pada *mission* objektifitas spesifik dari sistem. *Database security* termasuk dalam situasi sebagai berikut:

- Pencurian dan memanipulasi data
- Kehilangan kerahasiaan data
- Kehilangan *privacy*
- Kehilangan *integrity*

2.2 Teori-teori Khusus

2.2.1 Teori Pemasaran, Pemesanan, dan Penjualan

2.2.1.1 Pemasaran

Pemasaran adalah sebuah proses dalam memuaskan kebutuhan dan keinginan manusia. Jadi, segala kegiatan dalam hubungannya dalam pemuasan kebutuhan dan keinginan manusia merupakan bagian dari konsep pemasaran. Pemasaran dimulai dengan pemenuhan kebutuhan manusia yang kemudian bertumbuh menjadi keinginan manusia. Proses dalam pemenuhan kebutuhan dan keinginan manusia inilah yang menjadi konsep pemasaran. Mulai dari pemenuhan produk (product), penetapan harga (price), pengiriman barang (place), dan mempromosikan barang (promotion). Seseorang yang bekerja dibidang pemasaran disebut pemasar. Pemasar ini sebaiknya memiliki pengetahuan dalam konsep dan prinsip pemasaran agar kegiatan pemasaran dapat tercapai sesuai dengan kebutuhan dan keinginan manusia terutama pihak konsumen yang dituju.

Menurut Jill dan Matthew Ellsworth (1997, p xxxiii), dalam pemasaran terdapat empat prinsip dasar yang terdiri 4 P

- *Product* (produk)
- *Price* (harga)
- *Place* (tempat) – termasuk di dalamnya adalah distribusi.
- *Promotion* (promosi)

Metode ini yang dikenal dengan Bauran Pemasaran (Marketing Mix).

Dalam perkembangannya, untuk layanan jasa dikenal juga istilah 7 P dimana 4 P pertama adalah Product, Price, Place, dan Promotion. Untuk 3 P

yang selanjutnya adalah Bukti Fisik (Physical Evidence), Proses (Process) dan Orang (People).

Pemasaran lebih dipandang sebagai seni daripada ilmu, maka seorang ahli pemasaran tergantung lebih banyak pada ketrampilan pertimbangan dalam membuat kebijakan daripada berorientasi pada ilmu tertentu.

Pandangan ahli ekonomi terhadap pemasaran adalah dalam menciptakan waktu, tempat dimana produk diperlukan atau diinginkan lalu menyerahkan produk tersebut untuk memuaskan kebutuhan dan keinginan konsumen (konsep pemasaran).

Metode pemasaran klasik seperti 4P di atas berlaku juga untuk pemasaran internet, meskipun di internet pemasaran dilakukan dengan banyak metode lain yang sangat sulit diimplementasikan diluar dunia internet.

2.2.1.2 Pemesanan

Pemesanan adalah permintaan produk yang dilakukan oleh *customer*. Data yang diperlukan yaitu : kode pemesanan yang unik, tanggal pemesanan, data pelanggan, kode sales, kode barang, nama barang, harga barang, jumlah dan total.

2.2.1.3 Penjualan

Penjualan merupakan satu set *rekursif* dari kegiatan bisnis dan operasi pemrosesan informasi terkait yang dihubungkan dengan penyediaan barang

dan pelayanan pelanggan dan penerimaan pembayaran dari penjualan tersebut. (Romney, 2003, p157).

Penjualan dapat diartikan secara bebas sebagai usaha perseorangan untuk membujuk calon pelanggan untuk membeli barang atau jasa (Storholm, 1982, p14).

Penjualan ada dua macam bila dilihat dari cara pembayarannya :

1. Penjualan Tunai

   Penjualan tunai merupakan penjualan barang yang pembayarannya harus dilakukan secara tunai.

2. Penjualan Kredit

   Penjualan kredit merupakan penjualan barang yang pembayarannya dilakukan kemudian hari. Pembayaran ini dilakukan sesuai dengan syarat-syarat pembayaran yang diberikan atau sesuai dengan kesepakatan kedua belah pihak.

2.2.2 Teori Alat Bantu Analisis dan Perancangan

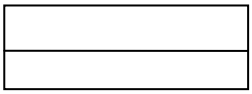
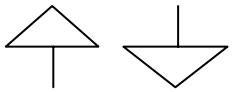
2.2.2.1 *Entity Relationship Diagram* (ERD)

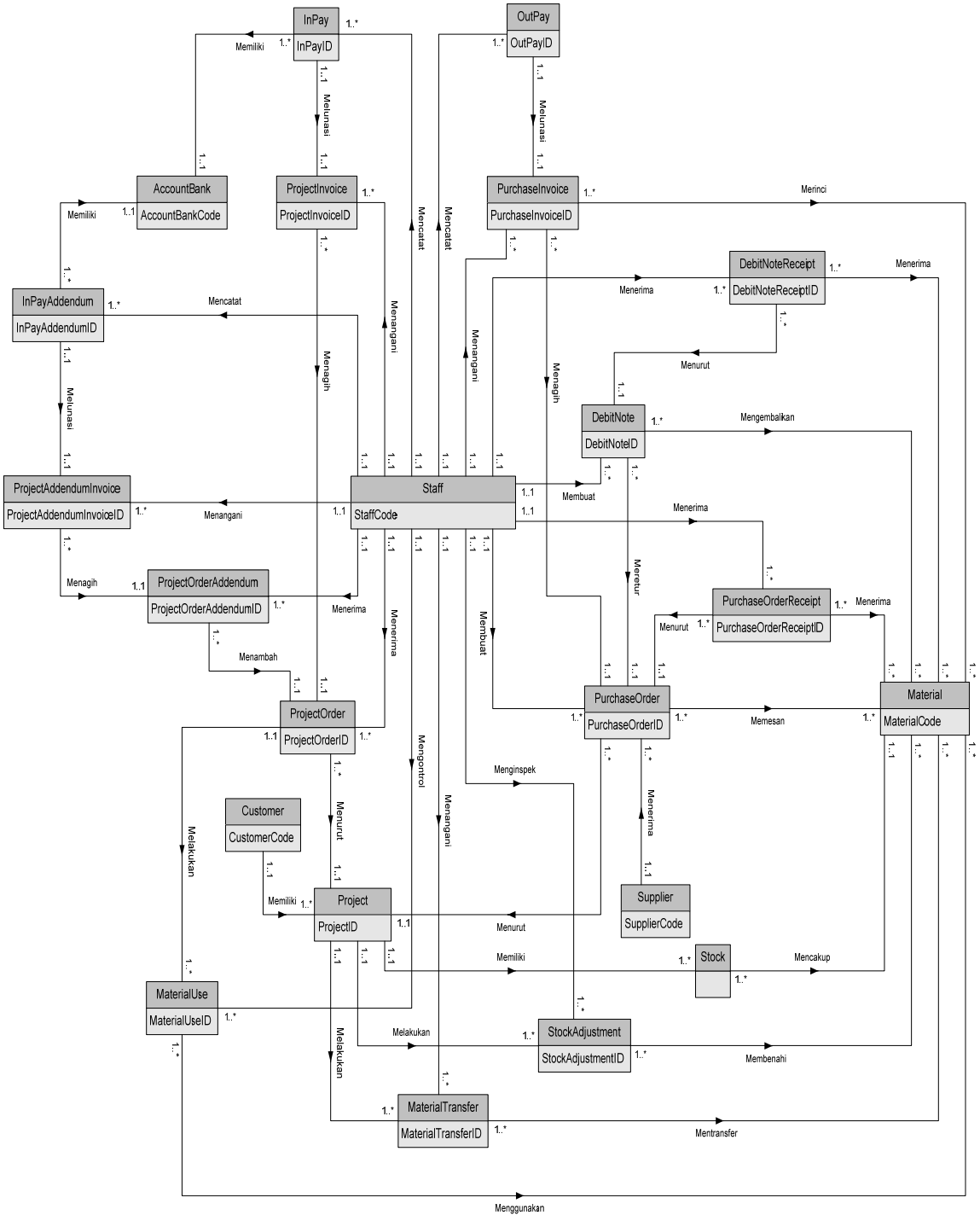
Model *entity relationship* merupakan salah satu model yang dapat memastikan pemahaman yang tepat terhadap data dan bagaimana penggunaannya di dalam suatu organisasi (Connolly dan Begg, 2005, p342). *ER Modeling* merupakan pendekatan *top-down* pada perancangan *database* yang dimulai dengan identifikasi entity dan relasi antar data yang harus direpresentasikan di dalam model, dan kemudian ditambahkan atribut dan setiap *constraint* pada entity, *relationship*, dan atributnya.

Secara umum symbol yang digunakan untuk ER diagram adalah sebagai berikut :

- Persegi panjang mewakili himpunan entitas.
- Ellips mewakili atribut.
- Jajaran genjang mewakili relasi antar entitas.
- Garis penghubung antara entitas dengan relasi, maupun antara relasi dengan himpunan atributnya.

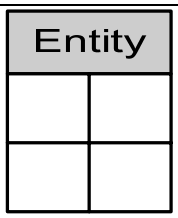
Tabel 2.1 Tabel Simbol ERD konseptual

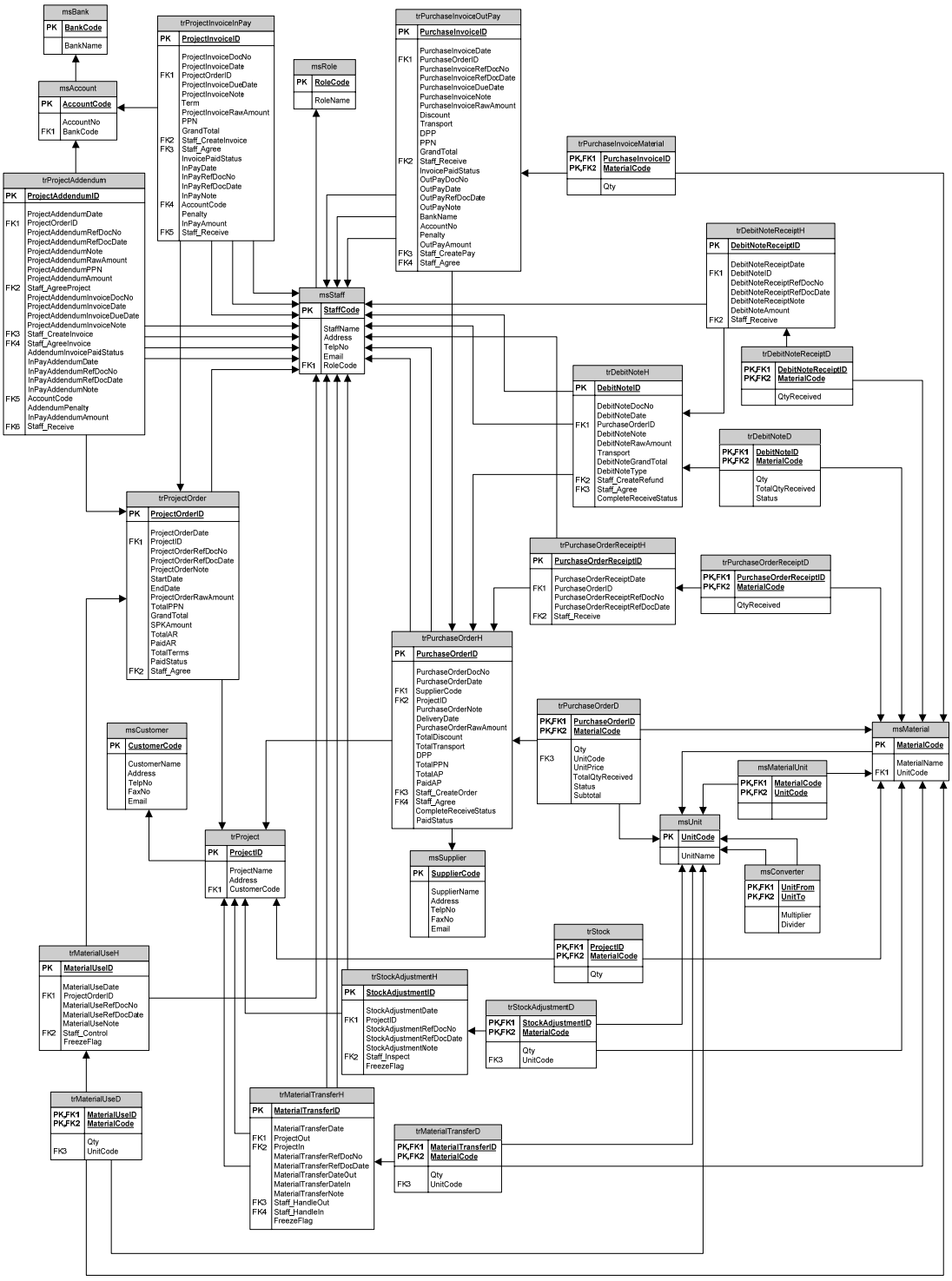
Simbol	Arti
	Menunjukkan entitas
	Menunjukkan arah baca hubungan dari satu entitas ke entitas yang lain
	Menunjukkan satu entitas berhubungan dengan entitas yang lain
	Menunjukkan generalization, aggregation, specialization dan composition



Gambar 2.2 Contoh ERD hasil konseptual(*Arrohman Nasirudin, 2010, Analisis Dengan Diagram Keterhubungan Entitas (ERD), http://arrohman.wordpress.com/2007/09/23/analisis-dengan-diagram-keterhubungan-entitas-entity-relationship-diagramerd/***)**

Tabel 2.2 Tabel Simbol ERD logical

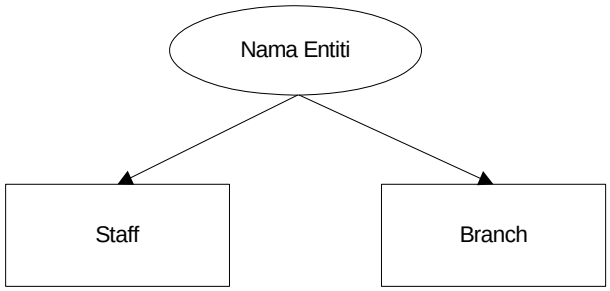
Simbol	Arti
	Menunjukkan entitas
	Menunjukkan arah hubungan dari satu entitas ke entitas yang lain yang bisa menentukan FK suatu entitas jika primary key suatu entitas terdapat pada entitas yang lain.



Gambar 2.3 contoh ERD hasil Logical (*Iffa Mashfufah,2010,Model Design Basis Data ERD (Entity Relationship Diagram), http://pupahhh.wordpress.com/2010/11/28/model-design-basis-data-erd-entity-relationship-diagram/***)**

2.2.2.1.1 *Entity Types*

Konsep dasar dari model ER adalah *entity types* yaitu kumpulan dari objek-objek dengan sifat (*property*) yang sama, yang diidentifikasi oleh *enterprise* karena keberadaannya yang bebas (*independent existence*) (Connolly dan Begg, 2005, p343). Keberadaan objek-objeknya dapat berupa fisik, seperti entiti pegawai, rumah dan pelanggan, maupun berupa abstrak, seperti entiti penjualan, pembelian dan peminjaman. Entiti *occurences*, yaitu pengidentifikasian objek yang unik dari sebuah tipe entiti.



Gambar 2.4 Diagramatik yang merepresentasikan tipe entiti Staff dan Branch

Menurut Connolly dan Begg (2005, p354), tipe entiti dapat diklasifikasikan menjadi :

a. Tipe entiti kuat, yaitu entiti yang keberadaannya tidak bergantung pada tipe entiti lainnya.
b. Tipe entiti lemah, yaitu tipe entiti yang keberadaannya bergantung pada tipe entiti lainnya.

2.2.2.1.2 *Relationship Types*

Relationship type adalah kumpulan keterhubungan yang mempunyai arti antara tipe entiti yang ada. *Relationship occurencess*, yaitu keterhubungan yang diidentifikasikan secara unik yang meliputi keberadaan tiap tipe entiti yang berpartisipasi (Connolly dan Begg, 2005, p346).

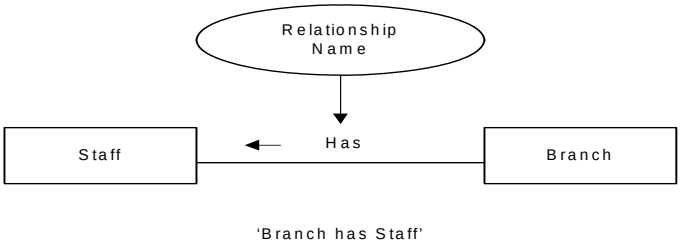
Tipe *relationship* digambarkan dengan sebuah garis yang menghubungkan entiti-entiti yang saling berhubungan. Sebuah relationship hanya dinamai dalam satu arah dan sebuah simbol panah ditempatkan di samping nama untuk menunjukkan arah yang tepat bagi pembaca untuk menginterpretasikan nama *relationship*.

2.2.2.1.3 *Degree of Relationship*

Derajat relationship yaitu jumlah tipe entitas yang ada dalam suatu *relationship* (Connolly dan Begg, 2005, p346). Derajat *relationship* terdiri dari :

1. *Binary Relationship*

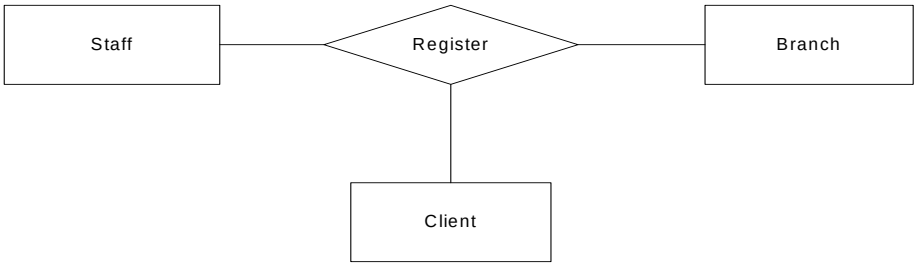
   Keterhubungan antar dua tipe entitas. Contoh *binary relationship* antara *Staff* dengan *Branch* yang disebut *Has.*



Gambar 2.5 Contoh *Binary Relationship*

2. *Ternary Relationship*

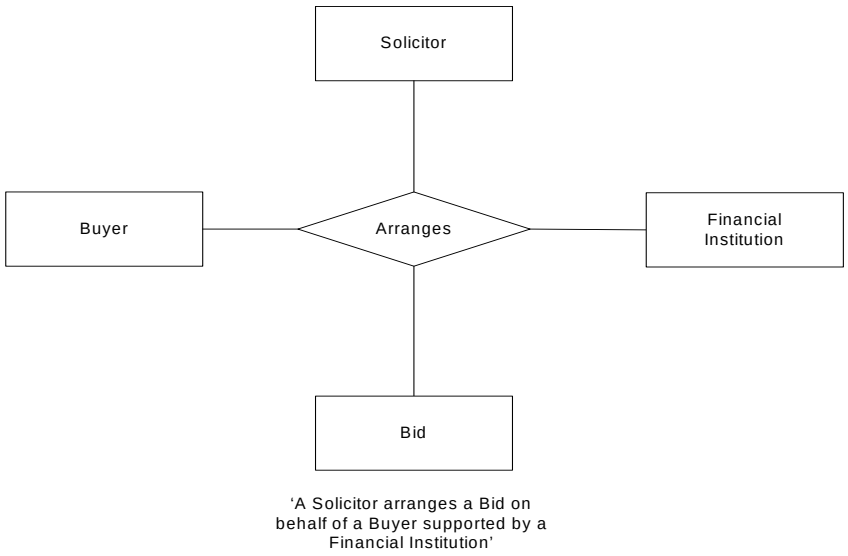
   Keterhubungan antar tiga tipe entitas. Contoh *Ternary Relationship* yang dinamakan *Registers*. Relasi ini melibatkan tiga tipe entity yaitu *Staff, Branch* dan *Client. Relationship* ini menggambarkan *Staff* mendaftarkan C*lient* pada *Branch.*



Gambar 2.6 Contoh *Ternary Relationship*

3. *Quaternary Relationship*

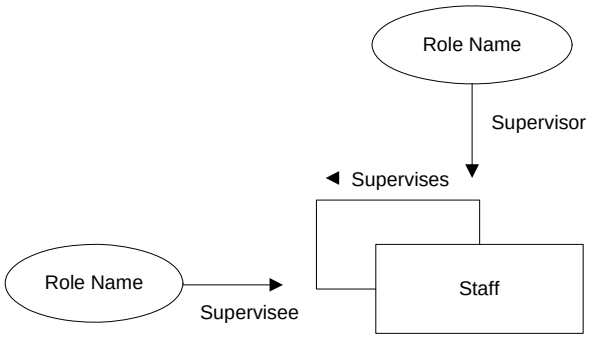
   Keterhubungan antar empat tipe entitas. Contoh *Quaternary Relationship* yang dinamakan *Arranges.* Relasi ini melibatkan 4 entity yaitu *Buyer, Solicitor, Financial Intstuttion* dan *Bid*. Relasi ini menggambarkan *Buyer*, diberi masukan oleh *Solicitor*, dan didukung oleh *Financial Institution*, melakukan penawaran (*Bid*).



Gambar 2.7 Contoh *Quaternary Relationship*

4. *Recursive Relationship*

Keterhubungan antar satu tipe entitas, dimana tipe entitas tersebut berpartisipasi lebih dari satu kali dengan peran yang berbeda. *Relationship* dapat diberikan *role names* untuk meng-identifikasikan keterkaitan tipe entitas dalam relationship. Contoh entitas *Staff* yang berperan menjadi *Supervisor* dan *Staff* yang di-*Supervisor-i.*



Gambar 2.8 Contoh *Recursive Relationship*

2.2.2.1.4 *Attributes* (Atribut)

Atribut merupakan sifat-sifat (*property*) dari sebuah entiti atau tipe *relationship*. Atribut domain adalah himpunan nilai yang diperbolehkan untuk satu atau lebih atribut (Connolly dan Begg, 2005, p350).

Macam-macam atribut :

a. *Simple attribute*, yaitu atribut yang terdiri dari satu komponen tunggal dengan keberadaan yang independen dan tidak dapat dibagi menjadi yang lebih kecil lagi. Dikenal juga dengan nama *atomic attribute*.
b. *Composite attribute*, yaitu atribut yang terdiri dari beberapa komponen, dimana masing-masing komponen memiliki keberadaan yang independen.
c. *Single-value attribute*, yaitu atribut yang mempunyai nilai tunggal untuk setiap kejadian dari tipe entiti.
d. *Multi-value attribute*, yaitu atribut yang mempunyai beberapa nilai untuk setiap kejadian dari tipe entiti.
e. *Derived attribute*, yaitu atribut yang memiliki nilai yang dihasilkan dari satu atau beberapa atribut lainnya yang berhubungan, dan tidak harus berasal dari tipe entiti yang sama.

2.2.2.1.5 *Keys*

Penentuan kunci (*key*) merupakan hal yang paling esensial pada basis data relasional. Kunci bukan hanya sebagai metode untuk

mengakses suatu baris tertentu, tetapi sekaligus juga menjadi pengenal unik dalam suatu tabel. Akan tetapi perlu juga diketahui bahwa tidak semua kunci dapat menjadi pengenal yang unik karena terdapat beberapa istilah kunci. Kunci dapat berupa sebuah atribut atau gabungan dari beberapa atribut (Connolly dan Begg, 2005, p352). Berikut ini adalah penjelasan jenis-jenis kunci yang digunakan dalam perancangan :

a. *Candidate key*, yaitu jumlah minimal atribut-atribut yang dapat mengidentifikasi setiap kejadian/*record* dari tipe entiti secara unik.
b. *Primary key*, yaitu *candidate key* yang dipilih untuk mengidentifikasikan setiap kejadian/*record* dari suatu tipe entiti secara unik.
c. *Composite key*, yaitu *candidate key* yang terdiri dari dua atau lebih atribut.
d. *Alternate key*, adalah setiap *candidate key* yang tidak terpilih menjadi *primary key* atau biasa disebut dengan *secondary key*.
e. *Foreign key*, adalah sebuah *primary key* pada sebuah entity yang digunakan pada entity lainnya untuk mengidentifikasi sebuah *relationship*.

2.2.2.1.6 *Structural Constraints*

Batasan utama pada *relationship* disebut *multiplicity*, yaitu jumlah (atau *range*) dari kejadian yang mungkin terjadi pada suatu tipe entiti yang terhubung ke satu kejadian dari tipe entiti lain yang berhubungan

melalui suatu *relationship*. *Relationship* yang paling umum adalah *binary relationship*. Macam-macam *binary relationship* yaitu :

a. *One-to-one* (1:1)

   Setiap relasi menggambarkan hubungan antara sebuah *entity occurrences* pada entiti yang satu dengan sebuah *entity occurences* pada entiti yang lainnya yang ikut serta dalam relasi tersebut.

b. *One-to-many* (1:*)

   Setiap relasi menggambarkan hubungan antara sebuah *entity occurence* pada entiti yang satu atau lebih *entity occurence* pada entiti yang lainnya yang ikut serta dalam relasi tersebut.

c. *Many-to-many* (*:*)

   Setiap relasi menggambarkan hubungan antara satu atau lebih *entity occurence* pada entity yang satu atau lebih entity lainnya yang ikut serta dalam relasi tersebut

d. *Multiplicity* untuk *complex relationship*

   Setiap relasi menggambarkan jumlah atau *range* dari kejadian yang mungkin dari suatu tipe entity dalam n-*ary relationship* ketika nilai entity yang lain (n-1) diketahui.

2.2.2.1.7 *Cardinality* dan *Participation Constraints*

Multiplicity terdiri atas dua *constraint* yang berbeda, yaitu :

1. *Cardinality*

   Cardinality adalah nilai maksimum dari relasi *occurrences* yang mungkin terjadi untuk sebuah entiti yang ikut serta pada suatu tipe relasi (Connolly dan Begg, 2005, p363).

2. *Participation*

   Menurut Connolly dan Begg (2005, p363), *participation* menentukan apakah semua atau beberapa *entity occurrence* yang ikut serta dalam sebuah relasi. *Participation constraint* dibagi menjadi :

   a. *Mandatory participation*, melibatkan semua *entity occurence* pada relasi tertentu.
   b. *Optional participation*, melibatkan beberapa *entity occurence* pada relasi tertentu.

2.2.2.2 *Flowchart*

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. Flowchart menolong analis dan programmer untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.

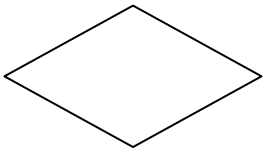
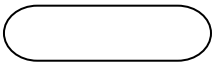
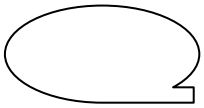
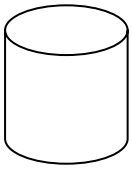
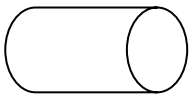
Flowchart biasanya mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut.

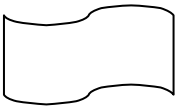
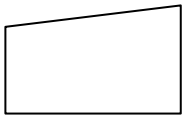
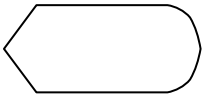
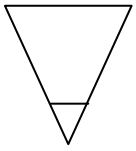
2.2.2.2.1 Elemen – Elemen Flowchart

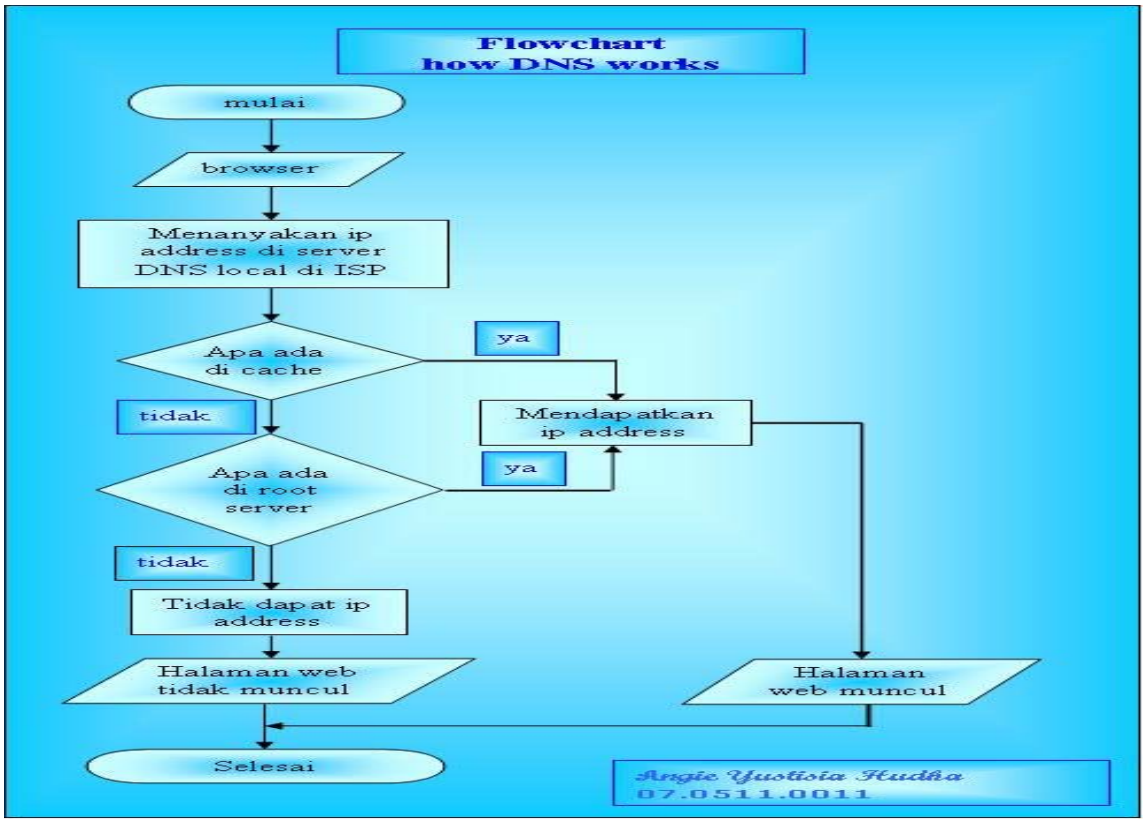
Flowchart terdiri dari 4 buah elemen dasar: sumber data dan tujuan data, aliran data, proses transformasi, dan penyimpanan data. Berikut adalah simbol – simbol yang digunakan dalam membuat Flowchart :

Tabel 2.3 Tabel Simbol *Flowchart*

Simbol	Nama	Deskripsi
	Input / Output	Merepresentasikan Input data atau Output data yang diproses atau informasi.
	Proses	Mempresentasikan operasi
	Connector	Keluar ke atau masuk dari bagian lain flowchart khususnya halaman yang sama
	Arrow	Merepresentasikan alur kerja

	Decision	Keputusan dalam program
	Preparation	Pemberian harga awal
	Terminal Points	Awal / akhir flowchart
	Punched card	Input / outuput yang menggunakan kartu berlubang
	Document	I/O dalam format yang dicetak
	Magnetic Tape	I/O yang menggunakan pita magnetik
	Magnetic Disk	I/O yang menggunakan disk magnetik
	Magnetic Drum	I/O yang menggunakan drum magnetik

	On-line Storage	I/O yang menggunakan penyimpanan akses langsung
	Punched Tape	I/O yang menggunakan pita kertas berlubang
	Manual Input	Input yang dimasukkan secara manual dari keyboard
	Display	Output yang ditampilkan pada terminal
	Manual Operation	Operasi Manual
	Off-line Storage	Penyimpanan yang tidak dapat diakses oleh komputer secara langsung

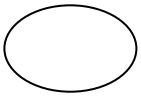
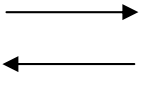
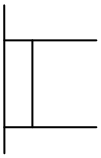
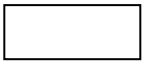


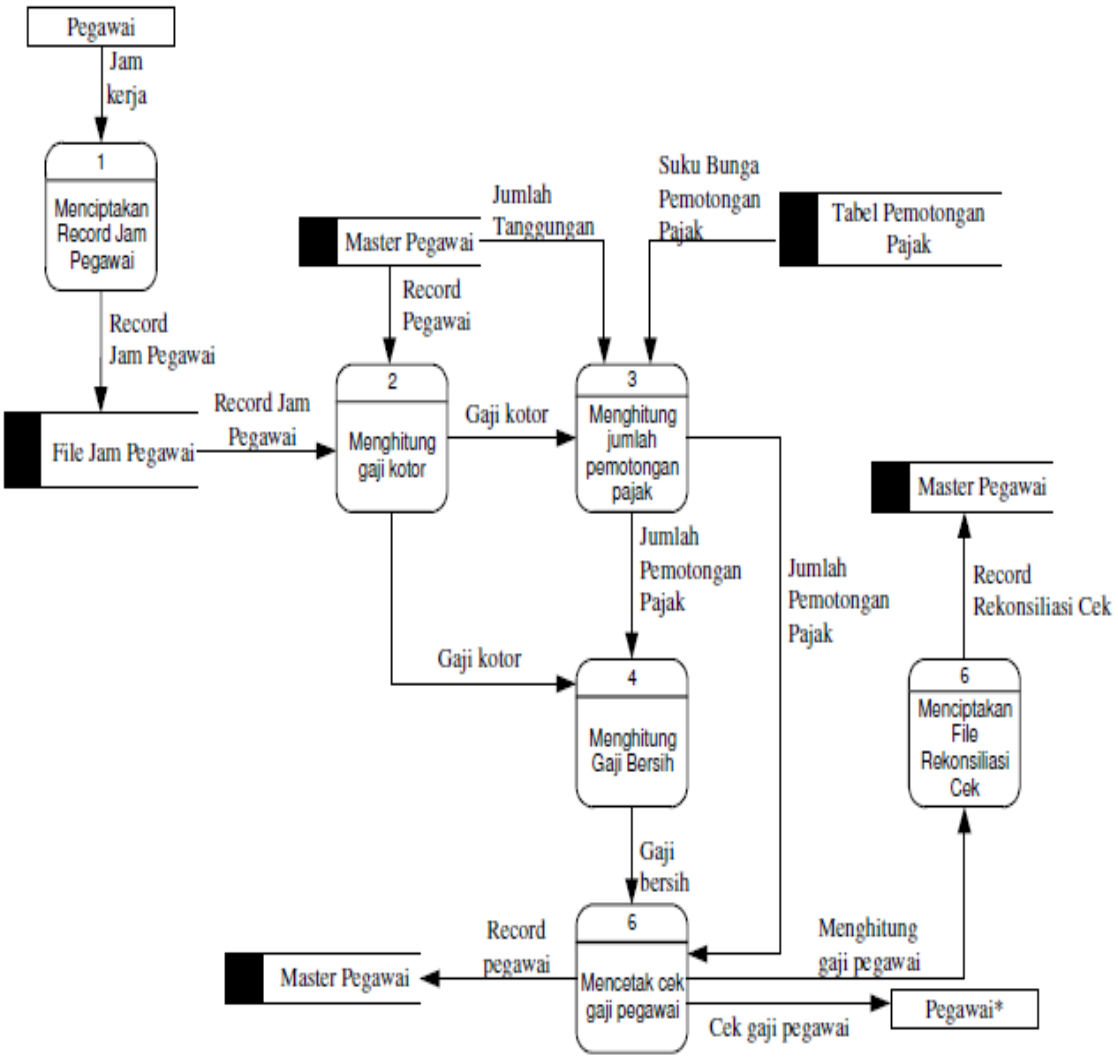
Gambar 2.9 Contoh Flowchart *(Duniangie, 2008, Cara Kerja DNS Server, http://duniangie.wordpress.com/page/2/)*

2.2.2.3 *Data Flow Diagram* (DFD)

Menurut McLeod (2001, p401), *Data Flow Diagram* adalah suatu gambaran grafis dari suatu sistem yang menggunakan sejumlah bentuk. Bentuk simbol untuk menggambarkan bagaimana data mengalir melalui suatu proses yang saling berkaitan. Walau nama diagram ini menekankan pada data, situasi justru sebaliknya, penekanannya ada pada proses.

Tabel 2.4 Tabel Simbol DFD

Simbol	**Keterangan**
	Proses adalah sesuatu untuk mengubah input menjadi output. Tiap simbol proses diidentifikasikan dengan tabel. Teknik pembuatan tabel yang paling umum adalah dengan menggunakan kata kerja atau obyek, tetapi dapat juga menggunakan sistem atau program komputer.
	Arus data terdiri dari sekelompok elemen data yang dapat digambarkan sebagai garis lurus atau lengkung.
	Penyimpanan data (*data store*) adalah suatu tempat penampungan data. Proses dapat memasukkan atau mengambil data dari *data store*.
	Terminator (*external entities*) digunakan untuk menggambarkan elemen - elemen lingkungan yang berada di luar sistem, yang menandai titik - titik berakhirnya sistem. Dapat berupa sistem lain, orang atau organisasi.



Gambar 2.10 Contoh DFD (*Islam-Download.net, 2010, Data Flow Diagram (DFD),* *http://islam-download.net/contoh-contoh/contoh-dfd.html***)**

2.2.2.4 *State Transition Diagram* (STD)

State Transition Diagram (STD) adalah sebuah *modeling tool* yang menggambarkan ketergantungan waktu pada sistem *real time* dan *human interface* pada sistem *online.*

Notasi yang paling penting dari STD adalah :

1. *State*

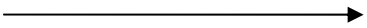
   Merupakan kumpulan keadaan atau atribut-atribut yang mencirikan benda atau orang pada waktu, keadaan dan kondisi tertentu.

Gambar 2.11 Notasi State

2. *Transition* (Perubahan) *State*

   Perubahan state ditandakan dengan tanda panah.

Kondisi

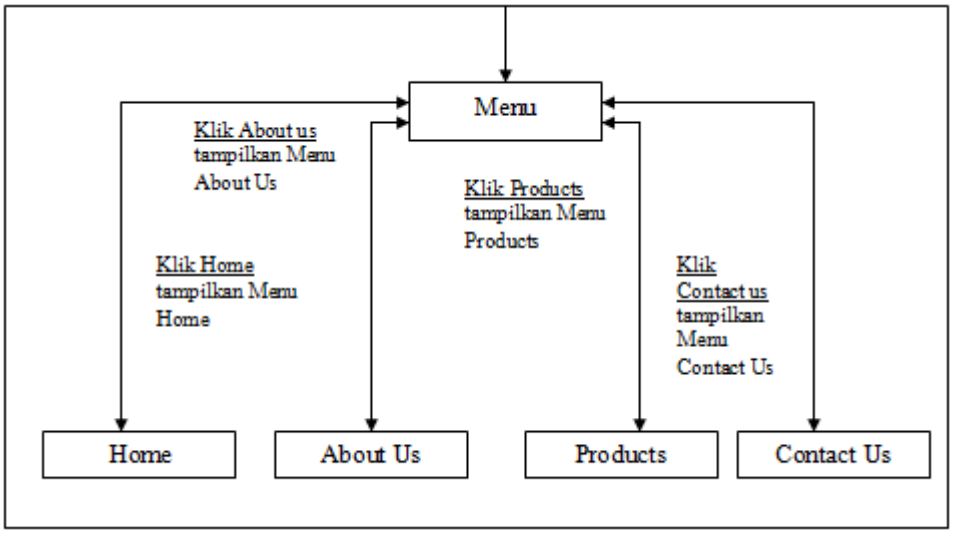


Aksi

Gambar 2.12 Notasi Transition State

Ada 2 hal yang perlu ditambahkan untuk melengkapi STD, yaitu :

1. Kondisi adalah keadaan lingkungan luar yang dapat dideteksi oleh sistem dan menyebabkan perubahan *state*. Kondisi dapat berupa sinyal, *interrupt* dan lainnya.
2. Aksi adalah apa yang dilakukan sistem jika ada perubahan *state.* Aksi dapat menghasilkan keluaran, tampilan pesan pada layar pengguna, membuat kalkulasi, dan lainnya.



Gambar 2.13 Contoh STD

2.2.3 Teori *Internet*

Menurut *Federal Networking Council*, istilah *Internet* menunjuk pada sistem informasi global di mana seluruh komputer terhubung antara satu sama lain di dalam jaringan. *Internet* adalah sebuah perpustakaan terbaik di dunia yang dilengkapi multimedia dan dapat saling berkomunikasi sehingga dapat mempermudah dalam berbagi informasi sehingga sistem ini menjadikan standar dunia akan informasi yang dituntut dengan *hypermedia.*

Menurut Tanenbaum (2003, p50), internet bukanlah sebuah jaringan, melainkan sekumpulan koleksi jaringan yang menggunakan protokol umum tertentu dan menyediakan layanan umum tertentu. Internet ini merupakan sistem yang tidak biasa dimana internet tidak direncanakan dan kontrol oleh siapapun.

Sedang berdasarkan pendapat Jill dan Matthew Ellsworth(1997, p4), internet merupakan jaringan besar yang dibentuk oleh interkoneksi jaringan

komputer dan komputer tunggal di seluruh dunia, lewat saluran telepon, satelit, dan sistem telekomunikasi lainnya.

2.2.3.1 TCP/IP

TCP/IP (Transmision Control Protocol/Internet Protocol) adalah bahasa komunikasi atau protokol dasar internet. TCP/IP dapat juga digunakan sebagai suatu protokol komunikasi dalam suatu jaringan pribadi (baik *intranet* ataupun *extranet*). Ketika anda disediakan dengan akses langsung kepada *internet*, komputer anda dilengkapi dengan salinan program TCP/IP sama halnya dengan komputer lain di mana anda mengirimkan pesan ke atau menerima pesan dari, juga mempunyai satu salinan TCP/IP.

2.2.3.2 *Hypertext Transfer Protocol* (HTTP)

Berdasarkan pendapat Kurose dan Ross(2003, p89), HTTP atau *HyperText Transfer Protocol*, aplikasi protokol layer Web, merupakan jantung dari Web itu sendiri. HTTP diimplementasikan dalam dua program : program *client* dan program *server*. Program *client* dan program *server*, dieksekusi pada *end system* yang berbeda, berkomunikasi satu sama lain dengan saling bertukar pesan HTTP.

Sedang menurut Comer(2000, p530), protokol yang digunakan untuk berkomunikasi antara sebuah *browser* dan *Web server* atau antara mesin menengah dengan *Web server* dikenal sebagai *HyperText Transfer Protocol (HTTP).*

World Wide Web

Menurut Jill dan Matthew Ellsworth(1997, p5), Web atau yang juga dikenal juga dengan *World Wide Web* merupakan sistem yang menyebabkan pertukaran data di Internet menjadi mudah dan efisien. Web sendiri terdiri atas dua komponen dasar :

- *Server* Web : sebuah komputer dan *software* yang menyimpan dan mendistribusikan data ke komputer lainnya (yang meminta informasi) melalui Internet.
- *Browser* Web : *software* yang dijalankan pada komputer pemakai (“*client”*) yang meminta informasi dari *server* Web dan menampilkannya sesuai dengan file data itu sendiri.

Sedang menurut pendapat Tanenbaum(2003,p611), *World Wide Web* merupakan sebuah serangkaian *framework* untuk mengakses dokumen terhubung yang tersebar di lebih dari sejuta mesin di seluruh Internet.