BAB 2

LANDASAN TEORI

2.1 Teori Basis Data

Teori – teori yang terkait dengan basis data yaitu:

2.1.1.Pengertian Sistem

Sistem adalah suatu cara untuk mengumpulkan, mengatur, mengendalikan, dan menyebarkan informasi ke seluruh organisasi. (Connolly and Begg, 2002, p27)

Sistem merupakan kumpulan dari elemen – elemen yang terintergrasi dengan maksud umum untuk mencapai suatu tujuan. (McLeod and Schell, 2001, p9)

Sistem secara sederhana dapat diartikan sebagai sebuah kumpulan dari elemen - elemen yang saling berhubungan atau berinteraksi yang membentuk suatu kesatuan. (O'Brien, 2002, p8)

Jadi, berdasarkan pendapat – pendapat di atas dapat disimpulkan bahwa sistem adalah kumpulan dari elemen – elemen yang mengatur, mengendalikan, dan menyebarkan informasi untuk mencapai suatu tujuan dari sebuah organisasi.

2.1.2.Pengertian data

Data adalah fakta – fakta yang telah diketahui dan dapat dikumpulkan serta dapat disimpan dalam media komputer yang secara relative mempunyai arti bagi pengguna. (Hoffer, Prescott, and McFadden, 2005, p5)

Data juga merupakan fakta – fakta atau observasi yang mentah, biasanya mengenai kejadian atau transaksi. (James A. O'Brien, 2002, p13)

Data merupakan sumber yang mesti dikontrol dan ditangani. (Whitten, Bently, Dittman, 2002, p475)

Jadi, dari fakta-fakta diatas dapat disimpulkan bahwa data adalah fakta – fakta mengenai kejadian atau transaksi yang berguna bagi pengguna.

2.1.3.Sistem Basis Data

Sistem basis data pada dasarnya adalah sistem penyimpanan *record* yang terkomputerisasi dimana tujuan sebenarnya adalah menyimpan informasi dan membuat informasi tersebut selalu tersedia pada saat dibutuhkan (Connolly and Begg, 2002, p4).

Sistem basis data merupakan suatu *record* terkomputerisasi yang mempunyai tujuan menyediakan informasi pada saat dibutuhkan. (Date, 2000, p5).

2.1.4.Pengertian basis data

Menurut Connolly & Begg, saat ini *database* sudah menjadi bagian penting dalam hidup sehari-hari yang seringkali penggunaannya tidak kita sadari. Sebuah *database* merupakan kumpulan data terelasi, sedangkan DBMS (*Data Management System*) merupakan *software* yang mengatur dan mengendalikan akses ke *database*. *Database* ialah sebuah bagian dari hubungan data logikal dan sebuah deskripsi dari data yang di desain untuk menemukan informasi yang dibutuhkan dari suatu organisasi. *Database* tidak hanya memegang data operasional organisasi tetapi *database* juga mendeskripsikan data tersebut. Sebagai contoh, sebuah *database* juga didefinisikan sebagai sebuah kumpulan gambaran diri sendiri dari rekaman-rekaman integrasi. Deskripsi dari data diketahui sebagai *system catalog* (atau *data dictionary* atau *meta data*). (Connolly and Begg, 2005, p15)

2.2 DBMS (Database Management System)

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

DBMS adalah perangkat lunak yang digunakan untuk menangani akses ke basis data seperti menambah, menghapus, mengambil, dan meng-*update record*. (Date, 2000, p43) Secara konseptual, cara kerja DBMS adalah sebagai berikut :

- Pengguna melakukan permintaan akses menggunakan bahasa *query* seperti SQL
- DBMS menangkap permintaan itu dan menganalisanya
- DBMS menginspeksi permintaan tersebut dari skema eksternal
- DBMS melaksanakan operasi yang diperlukan pada basis data yang disimpan

DBMS merupakan sebuah *software* yang berinteraksi dengan program aplikasi *user* dan *database*. Sebuah DBMS menyediakan beberapa fasilitas sebagai berikut :

- Mengijinkan *user* untuk mendefinisikan *database*, biasanya meliputi sebuah *Data Definition Language* (DDL). DDL mengijinkan *user* untuk menspesifikasikan tipe-tipe data dan struktus-struktur dan pembatas antara data yang disimpan di dalam *database*.
- Mengijinkan *user* untuk *insert, update, delete* dan memperbaiki data dari *database*. Biasanya meliputi *Data Manipulation Language* (DML). *Data Descriptions* mengijinkan DML untuk menyediakan sebuah fasilitas penyelidikan umum untuk data, ini disebut dengan *query language*.
- Menyediakan akses terkontrol ke dalam *database*. Contohnya:
  - ✓ Sistem kemanan

    Digunakan untuk menjaga akses *user* yang tidak terautorisasi *database*.
  - ✓ Integrasi *database*

    Digunakan untuk memelihara konsistensi dari penyimpanan data.
  - ✓ Ketepatan sistem terkontrol

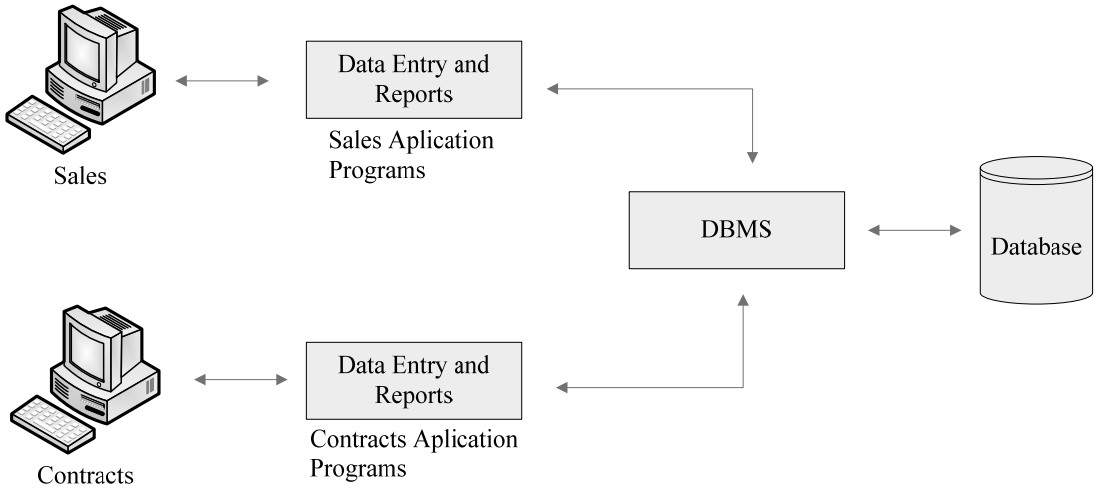
Ketepatan sistem kontrol mengijinkan akses terhubung dari *database.*

- ✓ Pemulihan sistem terkontrol

  Pemulihan sistem kontrol mengembalikan *database*.

- ✓ Katalog akses user

  Katalog akses *user* berisikan deskripsi-deskripsi dari data di dalam sebuah *database*.



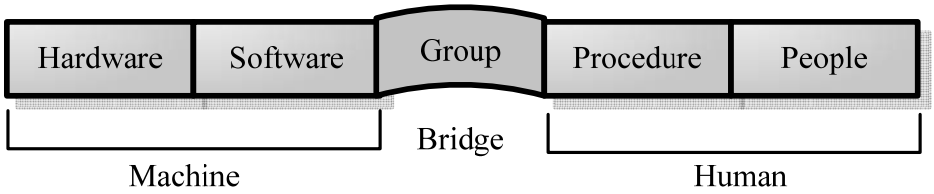
Gambar 2.1 : proses database

Komponen dari lingkungan DBMS terdiri dari :

- *Hardware*

  DBMS dan aplikasi-aplikasi memerlukan *hardware* untuk dapat berjalan. *Hardware* dapat bergerak dari sebuah *single personal computer*, ke *single mainframe*, juga ke sebuah jaringan dari komputer. *Hardware-hardware* tertentu tergantung dari syarat-syarat

organisasi dan kebutuhan dari DBMS itu sendiri. Beberapa DBMS berjalan hanya di *hardware* tertentu atau di *operating system*. Sebuah DBMS memerlukan jumlah minimum dari memori utama dan *disk space* untuk dapat berjalan, tetapi minimum konfigurasi tidak perlu memberikan performa yang cocok.



Gambar 2.2 : database environment

- *Software*

  Komponen *software* DBMS terdiri dari *software* DBMS itu sendiri dan program aplikasi, bersama-sama dengan sistem operasi, termasuk jaringan *software* jika DBMS sedang digunakan oleh sebuah jaringan.

- Data

  Data merupakan komponen terpenting dari DBMS yang menggambarkan sudut pandang *end user* sebagai jembatan penghubung kompenen mesin dan pengguna. Struktur dari *database* ini disebut dengan skema.

- Prosedur

Prosedur ialah instruksi dan aturan yang mengatur desain dan pengguna dari *database*. Pengguna dari sistem dan staff yang mengatur *database* memerlukan prosedur dokumen tentang bagaimana menggunakan atau menjalankan sistem. Prosedur ini terdiri dari instruksi tentang bagaimana untuk :

- ✓ Masuk ke dalam DBMS
- ✓ Menggunakan fasilitas DBMS tertentu atau program aplikasi
- ✓ Memulai dan mengakhiri DBMS
- ✓ Membuat salinan dari *database*
- ✓ Menangani kegagalan *software* dan *hardware*
- ✓ Mengubah struktur dari tabel, mengatur persilangan *database* di penyimpanan, meningkatkan performa, atau menyimpan data di penyimpanan kedua.

- Manusia

Komponen manusia terdiri dari : (Connolly, 2005, p22-24)

a. *Data Administrator* (DA)

Data Administrator bertanggung jawab atas manajemen sumber daya data termasuk perencanaan *database*, pengembangan

dan standar pemeliharaannya, peraturan dan prosedur, dan perancangan *database* secara konseptual/logis.

b. *Database Administrator* (DBA)

DBA bertanggung jawab atas realisasi fisik dari *database*, termasuk perancangan *database* fisik dan implementasi, kemanan dan kontrol integritas, perawatan sistem operasional, dan meyakinkan kinerja aplikasi yang memuaskan untuk pengguna.

c. *Application Developer*

Application Developer bertanggung jawab atas *database* setelah diimplementasikan, program aplikasi yang menyediakan fungsionalitas yang diperlukan untuk *end-users* harus diimplementasikan.

d. *End – users*

End – users adalah *client database* yang telah dirancang dan diimplementasikan, serta dipelihara agar dapat menyediakan kebutuhan-kebutuhan informasi mereka. *End – users* dapat diklasifikasikan berdasarkan cara mereka menggunakan sistem, yaitu :

- *Naïve users*

  Naïve users ialah *user* yang tidak mengetahui sama sekali mengenai DBMS.

- *Sophisticated users*

  Sophisticated users ialah *user* yang sudah mengenal struktus DBMS dengan baik dan fasilitas-fasilitas yang ditawarkan oleh DBMS.

Keuntungan dari DBMS ialah :

- Kontrol dari perulangan data
- Konsistensi data
- Informasi lebih lanjut dari data yang sama
- Membagi data
- Meningkatkan integritas data
- Meningkatkan kemanan
- Penegakkan standarisasi
- Keseimbangan dari persyaratan yang bertentangan
- Meningkatkan aksesibilitas data dan respon data
- Meningkatkan produktivitas
- Meningkatkan pemeliharaan melalui data independen
- Meningkatkan ketepatan
- Meningkatkan layanan *backup* dan *recovery*

Kerugian dari DBMS ialah :

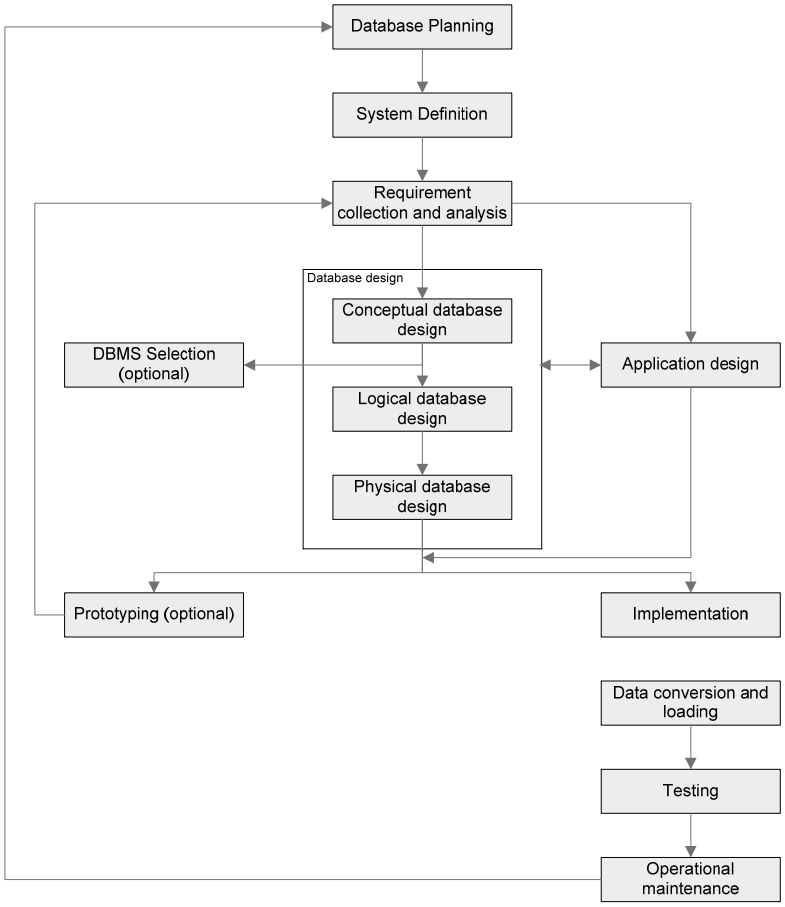
- Biaya pemrosesan data yang sangat tinggi
- Kebutuhan *software* dan *hardware* yang bertambah
- Memperoleh perangkat lunak yang mahal
- Tingginya dampak dari kegagalan

2.3 Siklus Hidup Aplikasi Basis Data

Basis data merupakan komponen dasar dari sebuah sistem informasi, yang perkembangan dan penggunaanya harus dilihat dari segi kebutuhan yang lebih luas dari sebuah organisasi.

Perancangan basis data merupakan proses membuat rancangan sebuah basis data yang akan mendukung operasi dan tujuan perusahaan.

Berikut ini adalah tahap – tahap siklus hidup aplikasi basis data : (Connolly and Begg, 2002, p272)



Gambar 2.3 : Siklus Hidup Aplikasi Basis Data

2.3.1.Database planning

Database planning merupakan kegiatan pengaturan yang memungkinkan tahapan – tahapan aplikasi basis data direalisasikan seefektif dan seefisien mungkin. Metodologi yang digunakan pada tahap ini ada dua, yaitu:

- *Database planning – mission statement*

  Mission statement mendefinisikan tujuan utama dari aplikasi basis data. *Mission statement* membantu untuk menjelaskan tujuan

projek basis data dan mengarahkan lebih jelas pembuatan yang efektif dan efisien dari aplikasi basis data yang diperlukan.

- *Database planning – mission objectives*

  Mission objectives harus mengidentifikasi tugas tertentu yang harus didukung oleh basis data. Asumsinya adalah jika basis data mendukung *Mission objectives* maka *mission statement* dapat ditemukan. *Mission objectives* dan *mission statement* dapat disertai dangan beberapa informasi tambahan yang spesifik, istilahnya, pekerjaan harus dilakukan, sumber untuk mengerjakannya, dan uang untuk membayarnya.

2.3.2.System definition

System definition menggambarkan ruang lingkup dan batasan aplikasi basis data dan *user view* utama. *User view* mendefinisikan apa yang diperlukan aplikasi basis data dari sudut pandang peranan pekerjaan tertentu (seperti *Manager* atau *Supervisor*) atau area aplikasi perusahaan (seperti *Marketing,* HRD, inventori). Sebuah aplikasi berbasis data dapat memiliki satu atau lebih *user view.* Mengenali *user view* merupakan aspek terpenting dalam mengembangkan aplikasi basis data karena membantu untuk memastikan tidak ada pengguna utama basis data yang terlupakan ketika mengembangkan kebutuhan untuk aplikasi yang baru. *User view* juga membantu dalam mengembangkan aplikasi basis data

yang cukup kompleks dengan menyertakan persyaratan yang dipisahkan menjadi bagian – bagian yang dapat diatur.

2.3.3.Requirement collection and analysis

Merupakan proses mengumpulkan dan menganalisa informasi tentang bagian organisasi yang didukung oleh aplikasi basis data, dan menggunakan informasi ini untuk mengenali kebutuhan pengguna pada sistem baru. Informasi dikumpulkan untuk masing – masing *user view* utama meliputi :

- Penjelasan data yang digunakan atau dihasilkan
- Detil bagaimana data digunakan atau dihasilkan
- Persyaratan tambahan lainnya untuk aplikasi basis data baru.

2.3.4.Database design

Merupakan proses membuat rancangan sebuah basis data yang akan mendukung operasi dan tujuan dari perusahaan.

2.3.5.DBMS selection

Merupakan pemilihan DBMS yang cocok untuk mendukung aplikasi basis data. Langkah utama untuk memilih DBMS yaitu:

- Menjelaskan ketentuan dari studi pembelajaran
- Daftar dua atau tiga produk
- Mengevaluasi produk

- Memberikan masukan untuk pilihan dan membuat laporan.

2.3.6.Application design

Merancang tampilan dan program aplikasi yang menggunakan dan memproses basis data.

2.3.7.Prototyping

Prototyping adalah bentuk kerangka kerja dari alikasi basis data. Tujuan utama dari pengembangan *prototype* aplikasi basis data adalah memungkinkan pengguna menggunakan *prototype* untuk mengidentifikasi fitur dari sistem berjalan dengan baik atau tidak, dan apabila memungkinkan untuk membuat peningkatan atau bahkan fitur-fitur baru ke dalam aplikasi basis data. Dengan cara ini, kita dapat mengetahui dengan baik kebutuhan untuk pengguna dan pengembang sistem dan bisa mengevaluasi kemungkinan yang terjadi dari sebagian rancangan sistem. Ada 2 strategi *prototype* yang digunakan saat ini :

- *Requirement prototyping*

  Menggunakan sebuah *prototype* untuk menentukan syarat dari aplikasi basis data yang diusulkan dan ketika persyaratan tersebut dapat dipenuhi, *prototype* tersebut tidak lagi digunakan.

- *Evolutionary*

Digunakan untuk tujuan yang sama, namun perbedaan yang menonjol yaitu *prototype* dikembangkann menjadi aplikasi basis data.

2.3.8.Implementation

Implementasi adalah realisasi fisikal dari basis data dan rancangan aplikasi. Implementasi basis data dapat dicapai dengan menggunakan *data definition language* (DDL) dari DBMS yang dipilih atau dengan *Graphical User Interface* (GUI), yang menyediakan fungsi yang sama ketika menyembunyikan pernyataan DDL tingkat rendah. Program aplikasi yang diimplementasikan menggunakan bahasa generasi 3 atau bahasa generasi 4 (3GL atau 4GL).

2.3.9.Data convertion and loading

Memindahkan data yang ada ke dalam basis data baru dan mengubah aplikasi yang ada untuk dijalankan pada basis data yang baru. Tahapan ini hanya dibutuhkan ketika basis data sistem baru menggantikan yang lama.

2.3.10. Testing

Testing merupakan proses menjalankan program aplikasi dengan tujuan menemukan *error*. Jika testing berjalan dengan baik, maka akan

menemukan kesalahan dengan program aplikasi dan mungkin juga dengan struktur basis data. Testing menunjukkan basis data dan program aplikasi terlihat berjalan sesuai dengan spesifikasi dan kinerja memuaskan.

2.3.11. Operational Maintenance

Merupakan proses mengawasi dan menjaga sistem setelah dilakukan instalasi. Pada tahap ini, proses yang dilakukan adalah :

- Mengawasi kinerja sistem. Jika kinerja sistem mengalami penurunan, *tuning* atau pengaturan ulang basis data mungkin diperlukan.
- Menjaga dan meng-*update* aplikasi basis data (ketika dibutuhkan). Keperluan baru tergabung dalam aplikasi basis data melalui tahapan siklus sebelumnya.
- Ketika aplikasi basis data beroperasi seluruhnya, pengawasan terjadi untuk memastikan kinerja sistem tetap pada level yang pantas.
- Proses pengawasan terus berlanjut sepanjang siklus aplikasi basis data dan pada waktunya mungkin akan terjadi pengaturan ulang basis data untuk mengikuti perubahan yang terjadi.

2.4 Model Relational

Model *relation* adalah gabungan dari banyak tabel yang ada di dalam *database* dan membuat sebuah *relation* atau hubungan di dalam tabel antara baris

dan kolom. Untuk membuat data di model relational memerlukan *relation schema* dan *relation instance.*

- *Relation Schema* adalah nama dari *relation* diambil bedasarkan nama dari *attribute* dan domain.
- *Relation instance* adalah sama seperti *relation schema* tapi dalam *relation instance* nama *relation*-nya juga ditentukan bedasarkan *distinct name*.

Model *relation* memiliki *key* yang berguna untuk menandai maupun untuk membatasi suatu *attribute* diantaranya

- *Supper key, attribute* yang unik dan digunakan sebagai identitas antar *entity*.
- *Candidate key, attribute* unik yang ada dalam setiap *entity*.
- *Primary key*, merupakan *candidate key* yang unik dan terpilih sebagai *primary key* dalam setiap *attribute*.
- *Alternate key*, merupakan *candidate key* yang tidak terpilih sebagian *primary key*.
- *Foreign key*, *attribute* yang memiliki kesamaan nama dengan *primary key* dalam *entity* lainnya dan keduanya saling berhubungan.

2.5 Perancangan Konseptual, Logikal dan Fisikal Basis Data

2.5.1 Perancangan Basis Data Konseptual

Proses membangun model informasi yang digunakan dalam sebuah organisasi, bebas dari semua pertimbangan fisik (Connolly and Begg, 2002, p419). Pertimbangan fisik yang dimaksud meliputi DBMS yang akan digunakan, program aplikasi, bahasa pemrograman, *platform* perangkat keras, unjuk kerja, dan pertimbangan fisik lainnya.

Perancangan basis data konseptual ini terdiri dari langkah-langkah sebagai berikut:

Langkah 1 : Membangun data model konseptual lokal untuk setiap view

1.1 Mengidentifikasi tipe identitas

Tipe entitas dapat dikenali dengan mengidentifikasi kata benda atau frase kata benda pada spesifikasi kebutuhan pengguna, objek besar seperti orang (*people*), benda (*thing*), atau konsep (*concept*). Alternatif lain adalah dengan mencari objek yang keberadaannya bebas.

1.2 Mengidentifikasi tipe relasi

Tahapan ini bertujuan untuk mengidentifikasi *relationship* penting yang ada antara tipe entitas-tipe entitas yang telah diidentifikasi sebelumnya. Tipe *relationship* diidentifikasikan dengan mencari kata kerja atau suatu kata berhubungan dengan kata kerja. Misalnya :

- *Staff* mengatur *property*
- *PrivateOwner* mempunyai *property*

1.3 Mengidentifikasi dan menghubungkan *attribute* dengan entitas atau tipe entitas

Tujuannya untuk menghubungkan *attribute* dengan entitas dan tipe *relationship* yang tepat. Atribut yang dimiliki oleh setiap entitas dan *relationship* harus memenuhi karakteristik *attribute* yaitu *simple/composite attribute, single/multi-valued attribute,* dan *derived attribute*.

1.4 Menentukan *domain attribute*

Domain adalah sekumpulan nilai dimana satu atau lebih *attribute* memperoleh nilainya (Connolly and Begg, 2002, p430).

1.5 Menentukan *candidate* dan *primary key* dari *attribute*

Candidate key adalah set atribut minimal dari sebuah entitas yang secara unik mengidentifikasi setiap kemunculan dari entitas tersebut. *Candidate key* dapat diidentifikasikan lebih dari satu, tetapi harus dipilih satu sebagai *primary key*, sedangkan *candidate key* yang lain disebut *alternate key*.

Berikut adalah acuan dalam menentukan *primary key* dari *candidate key*:

- *Candidate key* dengan set *attribute* minimal.
- *Candidate key* dengan nilai yang berubah paling sedikit.

- *Candidate key* dengan karakter paling sedikit.
- *Candidate key* dengan isi maksimum yang paling sedikit.
- *Candidate key* yang termudah digunakan dari sudut pandang *user*.

1.6 Mempertimbangkan pennggguna konsep model yang lebih tinggi (*enhanced modeling concepts*)/optional

Tahap ini bersifat optional, apakah akan digunakan pengembangan dari model entitas dengan menggunakan *enhanced modeling concepts*, seperti spesialisasi/generalisasi.

1.7 Memeriksa redudansi data model

Tahap ini memeriksa model data konseptual lokal apakah terjadi duplikasi atau tidak dengan dua langkah, antara lain:

- Memeriksa kembali *one-to-one relationship* (1:1)

  Kemungkinan ada dua entitas yang menggambarkan objek yang sama dalam organisasi. Maka, kedua entitas tersebut harus digabungkan.

- Menghilangkan relasi yang redundan

  Suatu *relationship* menjadi redundan jika informasi yang sama dihasilkan melalui *relationship* yang lainnya. Untuk meminimalkan data model maka *relationship* yang redundan harus dihilangkan.

1.8 Memvalidasi model data konseptual lokal dengan transaksi pengguna

Memeriksa model yang telah dihasilkan apakah mendukung transaksi pada *view* (Connolly and Begg, 2002, p435).

Pemeriksaan ini dapat menggunakan dua langkah yaitu:

- Mendeskripsi transaksi
- Menggunakan jalur transaksi

1.9 Meninjau model data konseptual lokal dengan pengguna

Langkah ini dilakukan dengan tujuan untuk memastikan bahwa data model merupakan representasi yang benar bagi setiap pandangan.

2.5.2 Perancangan Basis Data Logikal

Adapun tujuan dari *model logical data* menurut adalah untuk memproses pembuatan suatu model informasi yang digunakan di dalam suatu organisasi berdasarkan model data yang spesifik, tetapi tidak tergantung pada suatu DBMS dan perangkat keras lainnya Connolly (2002, p281).

Perancangan basis data logikal ini terdiri dari langkah-langkah sebagai berikut:

Langkah 2 : Membuat dan memvalidasi model logical data lokal untuk setiap bagian

2.1. Menghilangkan bagian yang tidak sesuai dengan model relasi (langkah pilihan)

Langkah-langkah ini meliputi :

- Menghilangkan *many-to-many* (*.*) tipe relasi *binary*
- Menghilangkan *many-to-many* (*.*) tipe relasi rekursif
- Menghilangkan tipe relasi komplek
- Menghilangkan *multi valued attribute*

2.2. Menurunkan relasi untuk model data logikal lokal

Membuat suatu relasi untuk model data logikal lokal yang merepresentasikan suatu entitas, relasinya dan juga *attribute* yang telah diidentifikasikan. Adapun pendeskripsian bagaimana relasi dapat diturunkan dari struktur data model yang ada sekarang, antara lain:

- Tipe *strong entity*
- Tipe *weak entity*
- *One-to-many* (1.*) tipe relasi *binary*
- *One-to-one* (1.1) tipe relasi *binary*
- *One-to-one* (1.1) tipe relasi rekursif
- *Superclass* atau *subclass* tipe relasi
- *Many-to-many* tipe relasi *binary*
- Tipe relasi komplek

- *Attribute multi value*

2.3. Memvalidasi relasi dengan normalisasi

Memvalidasi relasi dalam model data logikal lokal dengan menggunakan teknik dari normalisasi. Tujuan dari normalisasi adalah sebagai berikut:

- Menghilangkan kumpulan relasi dari *inserting*, *updating*, dan *delete* dependensi yang tidak diharapkan.
- Mengurangi kebutuhan restrukturasi kumpulan relasi dan meningkatkan *life spam* program aplikasi.
- Membuat model relasional lebih informatif.

2.4. Memvalidasi relasi dengan transaksi pengguna

Memastikan bahwa relasi di dalam model data logikal lokal mendukung transaksi yang diminta pengguna. Pada langkah ini, pengecekan bahwa relasi yang dibuat di langkah sebelumnya juga mendukung transaksi ini benar dan pastikan juga bahwa tidak ada kesalahan dalam relasi yang dibuat.

2.5. Menentukan batasan-batasan integritas

Terdiri dari 5 jenis yaitu:

- Data yang diperlukan
- Batasan *domain attribute*
- Integritas entitas
- Integritas referensial

- Batasan dari perusahaan

2.6. Meninjau model data logikal dengan pengguna

Membuat dokumentasi yang mendeskripsikan model data logikal lokal sebagai representasi yang sesuai dengan keadaan sebenarnya.

Langkah 3 : Membangun dan memvalidasi model data logikal global

3.1 Menggabungkan model data logikal lokal ke model global

Beberapa tugas dari pendekatan ini adalah sebagai berikut:

- Memeriksa nama dan isi suatu entitas atau relasi dan *candidate key*.
- Memeriksa nama dan isi dari suatu relasi dan *foreign key.*
- Menggabungkan entitas atau relasi dari model data logikal lokal.
- Memasukkan (tanpa penggabungan) entitas atau relasi untuk setiap model data lokal.
- Menggabungkan relasi atau *foreign key* dari model data lokal.
- Memasukkan (tanpa penggabungan) relasi atau *foreign key* pada setiap model data lokal.
- Mencek untuk entitas, relasi, dan *foreign key* yang hilang.
- Mencek untuk *foreign key*.
- Mencek untuk bahasa integritas.

- Membuat global ERD atau relasi diagram
- Meng-*update* dokumentasi

3.2 Memvalidasi model logikal data global

Memvalidasi relasi yang dibuat dari model data logikal global dengan menggunakan teknik dari normalisasi dan juga memastikan bahwa relasi yang dibuat mendukung transaksi.

3.3 Mengecek kemungkinan pengembangan di masa depan

Menentukan bagian mana yang kelihatan akan berubah ke masa depannya dan juga memperhatikan supaya model data logikal global dapat mengakomodasi perubahan tersebut.

3.4 Mereview model logikal data global dengan pengguna

Memastikan bahwa model logikal data global itu memang merepresentasikan *enterprise* yang ada. Hasil akhir dari perancangan logikal *database* adalah merancang suatu model informasi berdasarkan spesifik model yang ada (seperti model relasional), tetapi tidak tergantung terhadap suatu DBMS dan perangkat keras lainnya. Logikal basis data merancang suatu map untuk setiap lokal konseptual data. Jika terdapat lebih dari satu pandangan pemakai, maka model data logikal lokal akan dikombinasikan menjadi suatu model logikal data global yang merepresentasikan semua pandangan pengguna dari suatu perusahaan.

2.5.3 Perancangan Basis Data Fisikal

Proses memproduksi sebuah deskripsi dari implementansi basis data dalam *secondary storage*, yang menjelaskan relasi dasar, organisasi *file*, dan membuat indeks untuk mendapatkan akses yang efisien ke data, serta setiap *integrity constraint* yang saling berhubungan dan juga pengukuran kemanan.

Perancangan basis data fisikal ini terdiri dari langkah-langkah sebagai berikut:

Langkah 4 : Menerjemahkan model data logikal global untuk DBMS target

4.1 Mendesain relasi dasar (*base relation*)

Memutuskan bagaimana merepresentasikan relasi-relasi yang telah diidentifikasikan pada model data logikal global pada DBMS yang akan dipakai

4.2 Mendesain representasi dari data yang diturunkan

Memutuskan bagaimana merepresentasikan *derived attribute* dalam model data logikal global pada DBMS yang akan dipakai.

4.3 Mendesain *enterprise constraint*

Menentukan *enterprise constraint* untuk target DBMS.

Langkah 5 : Mendesain representasi fisikal

5.1 Menganilis transaksi

Memahami fungsi-fungsi dari transaksi yang akan dijalankan pada basis data dan menganalisi transaksi yang penting.

5.2 Memilih organisasi *file* yang akan digunakan

Menentukan organisasi *file* yang efisien. Ada 5 tipe organisasi *file*, yaitu :

- *Heap*
- *Hash*
- *Indexed Sequential Acces Method (ISAM)*
- *B-tree*
- *Clusters*

5.3 Memilih indeks yang digunakan

Memutuskan apakah dengan menggunakan indeks akan meningkatkan performa sistem.

5.4 Memperkirakan kapasitas *disk* yang diperlukan

Memperkirakan *disk storage* yang diperlukan untuk menggunakan sistem basis data, *disk storage* yang dimaksud adalah *secondary storage.*

Langkah 6 : Mendesain *user view*

Mendesain *user view* yang telah diidentifikasikan.

Langkah 7 : Mendesain mekanisme keamanan

Membatasi akses basis data oleh pengguna yang tidak berhak dan menspesifikasikan pengguna terhadap basis data yang dapat diakses.

2.6 ER (Entity Relationship)

ER adalah permodelan data yang menggunakan beberapa notasi untuk menggambarkan data yang berhubungan dengan *entity* dan *relationship* yang di deskripsikan oleh data tersebut. (Whitten et. Al., 2004, p295-307)

Konsep dasar model data ERD yaitu :

a. *Entities*

Entities adalah sebuah *class* dari orang, tempat, objek, kejadian, atau konsep mengenai apa yang diperlukan untuk mengambil dan menyimpan data. Kategori dari *entity* adalah sebagai berikut :

- Orang : manajer, *supplier*, pelanggan, dan karyawan.
- Tempat : ruangan, kantor cabang, dan bangunan.
- Objek : produk, mesin, dan bahan dasar.
- Kejadian : *invoice*, pemesanan, dan penjualan.
- Konsep : stok, dana, *account*, dan kualifikasi.

b. *Attributes*

Attributes adalah karakteristik dari sebuah *entity*.

- *Domain*

  Nilai dari tiap *attribute* didefinisikan ke dalam tiga *property*, yaitu:

  ✓ Tipe data

  Properti dari *attribute* yang mengidentifikasikan tipe data yang dapat disimpan ke dalam *attribute*.

  ✓ *Domain*

  Properti dari *attribute* yang mendefinisikan nilai apa yang boleh diambil oleh suatu *attribute*.

  ✓ *Default Value*

  Suatu nilai yang akan disimpan apabila nilai tidak dispesifikasikan oleh pengguna.

- *Identification*

  Dengan banyaknya *instance* yang dimiliki oleh suatu *entity* maka diperlukan juga suatu *key* yang unik untuk mengidentifikasikan setiap *instance* berdasarkan data dari *attribute*. Yang dimaksud dengan *key* adalah suatu *attribute* atau sekumpulan *attribute* yang mengasumsikan nilai yang unik dari setiap bagian dari *entity* dan sering kali disebut sebagai *identifier*.

  Candidad key adalah salah satu *key* yang memiliki kemungkinan untuk dijadikan *primary key*.

  Primary key adalah sebuah *candidate key* yang unik dan mengidentifikasikan sebuah bagian dari *entity*.

Alternate key adalah *candidate key* yang tidak dijadikan *primary key*.

c. *Relationship*

Relationship adalah hubungan antara satu atau lebih *entity*. *Cardinality* adalah jumlah minimum dari keberadaan suatu *entity* yang mungkin direlasikan ke *entity* lain. *Degree* adalah sejumlah *entity* yang berpartisipasi dalam sebuah *relationship. Foreign key* adalah sebuah *primary key* yang digunakan oleh *entity* lain untuk mengidentifikasi instansi dari sebuah *relationship*.

d. *Generalization*

Generalization adalah sebuah konsep dimana *attribute-attribute* yang umum bagi beberapa tipe dari *entity* dikelompokkan ke dalam *entity* mereka masing-masing.

2.7 Normalisasi

Normalisasi adalah suatu teknik untuk menghasilkan himpunan relasi dengan *property* yang diinginkan bedasarkan kebutuhan – kebutuhan data suatu organisasi (Connolly,2002,p376).

Proses normalisasi dimulai dengan memindahkan data sumber ke bentuk tabel dengan format baris dan kolom. Tabel ini berbentuk tidak normal dan disebut dengan *normalized table* (Connolly,2002,p388).

Unnormalized form (UNF) adalah tabel yang terdiri dari satu atau lebih kelompok yang berulang atau *repeating group* (Connolly,2002,p387). *Repeating group* adalah sebuah atribut atau himpunan *attribute* di dalam tabel yang memiliki lebih dari satu nilai untuk sebuah *primary key* pada tabel tersebut (Connolly,2002,p388).

Tingkatan normalisasi yang digunakan sebagai landasan skripsi adalah:

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

   Suatu data dapat dikatakan *unnormalized*, jika di dalamnya mengandung kelompok berulang, sehingga untuk membentuk normalisasi pertama (1NF) *repeating group* harus dihilangkan. Untuk menjadi 1NF maka grup yang berulang dihilangkan degan mengisi pada bagian yang kosong dengan data yang seharusnya pada suatu bentuk *record*.

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

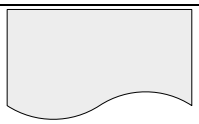
   Dapat dihasilkan dengan melihat apakah ada *attribute* atau bukan *primary key* yang merupakan fungsi dari sebagian *primary key* (*partial dependence)*. Dalam normalisasi kedua (2NF) setiap *attribute* yang tergantung parsial ini harus dipisahkan dengan mengikutsertakan determinannya. Bentuk normal diperoleh bila setiap *attribute* bukan bagian *primary key* dari suatu tabel sepenuhnya merupakan fungsi (*fungsional depence)* dari *primary key* tersebut.

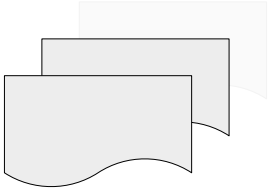
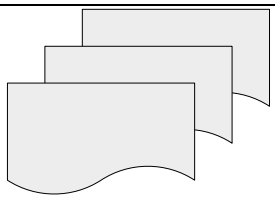
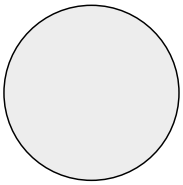
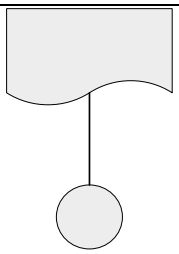
3. *Third Normal Form (3NF)*

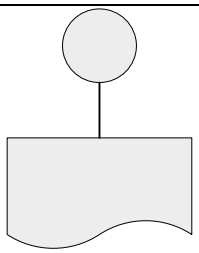
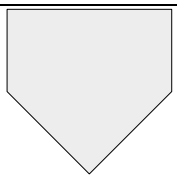
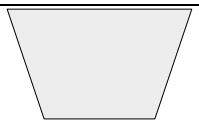
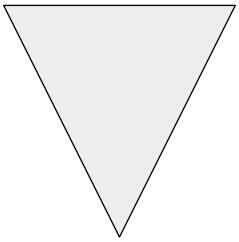
Pengujian terhadap 3NF dilakukan dengan cara melihat apakah terdapat *attribute* bukan *key* tergantung fungsional terhadap *attribute* bukan *key* yang lain (disebut ketergantungan transitif atau *transitive dependence)*. Dengan cara yang sama, maka setiap ketergantungan transitif dipisahkan. 3NF sudah cukup bagus dalam arti bahwa anomali yang dikandungnya sudah sedemikian minimum (hampir tidak ada).

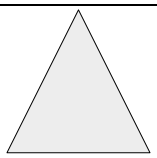
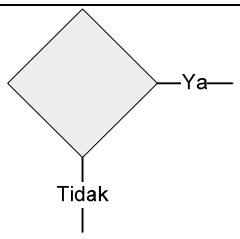
2.8 Diagram Aliran Dokumen (DAD)

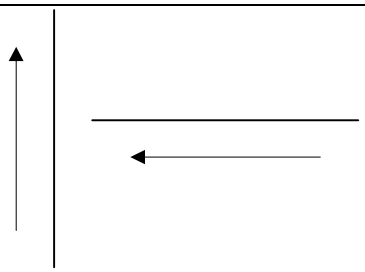
Proses-proses bisnis perusahaan dapat dijelaskan dengan menggunakan bagan alir dokumen (Mulyadi, 2001, p60). Berikut ini symbol-simbol standard DAD dengan maknanya masing-masing antara lain :

	**Dokumen.** Simbol ini digunakan untuk menggambarkan semua jenis dokumen, yang merupakan formulir yang digunakan untuk merekam data terjadinya suatu transaksi. Nama dokumen dicantumkan ditengah simbol. Bagan alir harus menunjukkan dengan jelas darimana suatu dokumen masuk ke dalam sistem dan kemana dokumen keluar dari sistem.

	**Dokumen dan tembusanny**a. Simbol ini digunakan untuk menggambarkan dokumen asli dan tembusannya. Nomor lembar dokumen dicantumkan di sudut kanan atas.
	**Berbagai dokumen**. Simbol ini digunakan untuk menggambarkan berbagai jenis dokumen yang digabungkan bersama di dalam satu paket. Nama dokumen dituliskan di dalam masing-masing simbol dan nomor lembar dokumen dicantumkan di sudut kanan atas simbol dokumen yang bersangkutan.
	**Penghubung pada halaman yang sama (*on-page connector*).** Simbol penghubung untuk memungkinkan aliran dokumen berhenti di suatu lokasi ada halaman tertentu dan kembali berjalan di lokasi lain pada halaman yang sama.
	**Akhir arus dokumen** dan mngarahkan pembaca ke simbol penghubung halaman yang sama yang bernomor seperti yang tercantum di dalam simbol tersebut.

	**Awal arus dokumen** yang berasal dari simbol penghubung halaman yang sama, yang bernomor seperti yang tercantum di dalam simbol tersebut.
	**Penghubung halaman yang berbeda (*off – page connector)*.** Jika untuk menggambarkan bagan alir suatu suatu sistem diperlukan lebih dari satu halaman, simbol ini harus digunakan untuk menunjukan kemana dan bagaimana bagan alir terkait satu dengan lainnya. Nomor yang tercantum pada halaman tertentu terkait dengan bagan alir yang tercantum pada halaman tertentu terkait dengan bagan alir yang tercantum pada halaman yang lain.
	**Kegiatan manual.** Simbol ini digunakan untuk menggambarkan kegiatan manual. Uraian singkat kegiatan manual dicantumkan di dalam simbol ini.
	**Arsip sementara.** Simbol ini digunakan untuk menunjukan tempat penumpanan dokumen, seperti alamari arsip dan kotak arsip. Terdapat dua tipe arsip dokumen : arsip sementara dan

	arsip permanen. Arsip sementara adalah penyimpanan dokumen yang dokumennya akan diambil kembali dari arsip tersebut di masa yang akan datang untuk keperluan pengolahan lebih lanjut terhadap dokumen tersebut. Untuk menunjukan urutan pengarsipan dokumen digunakan simbol berikut ini: A = menurut abjad N= menurut nomor urut T = kronologis, menurut tanggal
	**Arsip permanen.** Simbol ini digunakan untuk menggambarkan arsip permanen yang merupakan tempat penyimpanan dokumen yang tidak akan diproses lagi dalam sistem yang bersangkutan.
	**Keputusan.** Simbol ini menggambarkan keputusan yang harus dibuat dalam proses pengolahan data. Keputusan yang dibuat ditulis di dalam simbol.

	**Garis alir (*flow line*).** Simbol ini menggambarkan arah proses pengolahan data. Anak panah tidak digambarkan jika arus dokumen mengarah ke bawah dan ke kanan. Jika arus dokumen mengalir ke atas atau ke kiri, anak panah perlu dicantumkan.
	**Mulai / berakhir (*terminal*).** Simbol ini untuk menggambarkan awal dan akhir suatu sistem.

2.9 Web Internet

2.9.1 Web

Web adalah aplikasi yang menyangkut banyak komputer. Web menyedikan fasilitas seperti menyimpan informasi, menemukan dan mengambil informasi, menyimpan dan mengeksekusi program komputer, meng-*input* dan memanipulasi informasi, dan yang paling utama adalah siapa saja yang dapat menggunakan fitur-fitur tersebut. Web menyediakan fasilitas tersebut dengan menggunakan jaringan komputer dunia luas yang disebut internet.

2.9.2 Internet

Internet merupakan sebuah jaringan komputer yang bersifat global. Diperkenalkan pada tahun 1960-an dengan dana yang disediakan oleh Departerment Pertahanan Amerika Serikat. Pada awalnya, internet dirancang untuk menghubungkan sistem komputer utama dari beberapa Universitas dan organisasi penelitian. Saat ini, internet dapat diakses lebih dari 1 miliar komputer dan peralatan komputer pengendali secara luas (Deitel & Deitel, 2003, p12).

Keuntungan-keuntunan yang di dapatkan melalui penggunaan internet antara lain :

- Membuat pekerjaan menjadi lebih mudah
- Membuat informasi secara instan dan mudah diakses secara luas
- Memungkinkan individu dan bisnis lokal kecil untuk memperkenalkan bisnisnya kepada dunia
- Mengubah cara suatu bisnis dijalankan
- Masyarakat luas dapat mencari harga terbaik dari produk atau layanan secara *virtual*
- Komunitas dengan hobi tertentu dapat saling berhubungan

TCP/IP (*Transmission Control Protocol/Internet Protocol*) adalah protokol komuniaksi utama yang dipergunakan di seluruh jaringan internet. Permintaan melalui HTTP dienkapsulasi dengan TCP/IP agar dapat digunakan melalui WWW.

2.10 Teori Peristilahan Pekerjaan

2.10.1. Kepegawaian

Sistem kepegawaian adalah suatu sistem atau cara pengelolaan dalam bidang kepegawaian menyangkut semua aspek yang ada dalam sistem kepegawaian mulai dari cara penerimaan, pengangkatan, kenaikan golongan, penggajian karyawan dan sebagainya. (Wursanto, 1987, p34)

2.10.2. Klasifikasi Jabatan

Klasifikasi Jabatan adalah suatu kegiatan penggolongan jabatan-jabatan berdasarkan macam tugas yang dilakukan berikut cara-cara yang diperlukan untuk memangku jabatan tersebut atau suatu kegiatan penyusunan secara teratur dari jabatan-jabatan dalam beberapa golongan atau tingakatan untuk dapat diketahui derajat tiap-tiap jabatan. (Wursanto, 1987, p75)

2.10.3. Mutasi

Salah satu tindak lanjut yang dilakukan atas hasil penilaian prestasi karyawan adalah mutasi karyawan. Mutasi adalah suatu perubahan posisi/jabatan/tempat/pekerjaan yang dilakukan baik secara horizontal maupun vertikal dalam suatu organisasi. Tujuan mutasi (Faustino, 1995, p197) adalah :

- Menciptakan produktivitas karyawan
- Menciptakan keseimbangan antara tenaga kerja dengan komposisi pekerjaan atau jabatan
- Memperluas pengetahuan karyawan
- Memberikan perangsangan agar karyawan dapat meningkatkan karier
- Menghilangkan rasa bosan terhadap pekerjaan
- Melaksanakan hukuman atas pelanggaran karyawan
- Memberikan imbalan terhadap pekerjaan karyawan
- Sebagai alat pendorong agar semangat kerja meningkat
- Menyesuaikan pekerjaan dengan kondisi fisik karyawan
- Mengatasi perselisihan sesama karyawan
- Sebagai tindakan pengamanan yang lebih baik

2.10.4. Cuti

Jenis cuti yang merupakan hak dari setiap karyawan, yaitu :

- Cuti tahunan
- Ijin kepentingan keluarga
- Cuti melahirkan
- Cuti menjalankan ibadah agama
- Cuti diluar tanggungan perusahaan

2.10.5. Kedisiplinan

Menurut asal katanya istilah disiplin berasal dari kata *disciple* yang berarti murid, *disciplinary* mengenai kepatuhan, kata itu berubah menjadi kata *discipline* yang mempunyai arti kepatuhan atau hal yang menyangkut tata tertib. Dengan demikian, maka disiplin merupakan suatu kepatuhan kepada aturan-aturan, norma-norma, patokan-patokan, hukum, dan tata tertib yang berlaku. (Wursanto, 1987, p145)

2.10.6. Prestasi Kerja

Dalam prestasi kerja pengangkatan seorang karyawan (kenaikan pangkat) didasarkan atas kecakapan dan prestasi yang dicapai oleh seorang karyawan yang bersangkutan. (Wursanto, 1987, p36)