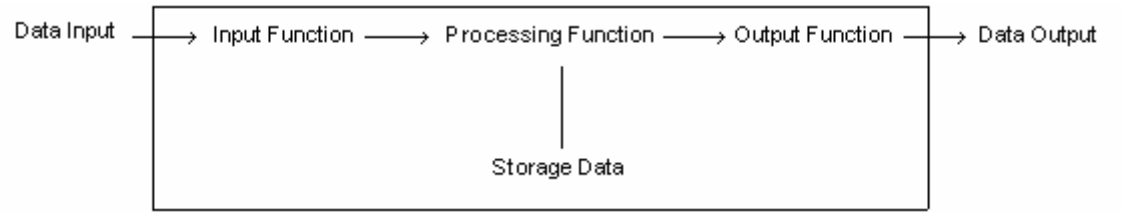
BAB 2

LANDASAN TEORI

2.1 Definisi Sistem Informasi

Sistem Informasi adalah koleksi komponen – komponen yang saling bekerja sama untuk menyediakan informasi untuk membantu operasi dan manajemen suatu organisasi.



Gambar 2.1 Fungsi Sistem Informasi

Fungsi sistem informasi:

Input function : menerima data dari luar sistem

Processing Function : mengatur kalkulasi kuantitas inventori apabila terjadi perubahan dan secara tidak langsung memanipulasi data masukan dan data simpanan

Storage function : menjaga data masukan bersamaan dengan data penyimpanan lain dan mengambil data simpanan apabila diperlukan oleh sistem dan memantau kuantitas inventori

Output function : menghasilkan informasi yang telah diolah bagi pengguna

Komponen sistem informasi :

- *Hardware*

  Terdiri dari komputer , peralatan komunikasi, dan peralatan lain yang digunakan dalam suatu sistem. Banyaknya sistem informasi yang menyertakan lebih dari satu tipe saling terhubung , menggunakan peralatan komunikasi seperti kabel atau kabel *fiber optic* dan *circuit board* spesial. *Hardware* lain seperti kamera digital, *microphone*, dan *circuit board* untuk mengirimkan faks digunakan dalam sistem informasi.

- *Software*

  Terdiri dari instruksi-instruksi yang menyuruh *hardware* untuk melakukan suatu pekerjaan . Komputer tidak dapat berfungsi tanpa *software*, mereka harus mempunyai instruksi – instruksi untuk memberitahu apa yang harus dilakukan.

- *Storage data*

  Terdiri dari semua data yang disimpan di komputer dalam sistem dan digunakan oleh *software* sistem. Komponen penyimpanan data suatu sistem informasi hanya terdiri dari data yang disimpan sistem, bukan data masukan dan atau keluaran. Walaupun data masukan dan atau keluaran mengalir melalui sistem , tetapi mereka bukan bagian dari sistem karena mereka tidak berparisipasi untuk tujuan sistem, data masukan dan atau keluaran walaupun kritis untuk kegunaan sistem informasi, bukanlah bagian dari sistem.

- *Personnels*

  Merupakan orang-orang yang menyuplai data masukan ke sistem, menerima data keluaran dari sistem mengoperasikan *hardware* dari sistem dan menjalankan *software* yang merupakan bagian dari sistem.

- *Procedures*

  Merupakan instruksi – instruksi yang memberitahu orang bagaimana cara menggunakan dan mengoperasikan sistem. Sama seperti *hardware* yang tidak dapat berfungsi tanpa *software*, orang pun tidak tau apa yang harus dilakukan jika tidak ada prosedur yang harus diikuti.

2.2 Konsep Basis Data

Basis data pada saat ini adalah bagian dari kehidupan kita yang seringkali kita tidak sadar telah menggunakannya. Ketika membeli barang dari supermarket, kemungkinan basis data diakses. Kasir menggunakan *bar code* untuk mencari harga barang dari basis data produk. Begitu juga ketika kita membeli menggunakan *credit card*, memesan hotel di agen perjalanan, peminjaman buku, menggunakan internet, dan lain sebagainya. Sebelum adanya aplikasi basis data seperti DBMS, penyimpanan data masih menggunakan penyimpanan di dalam suatu file. Sistem penyimpanan file secara manual bekerja dengan baik bila data yang disimpan kecil. Bagaimanapun, penyimpanan file secara manual akan tidak baik bila kita harus mereferensi silang proses informasi antara file.

Menurut Connolly (2002,p7), *File-Based System* adalah suatu kumpulan dari program aplikasi yang memberikan servis kepada *user* untuk membuat suatu laporan. Setiap program mendefinisikan dan mengatur datanya masing-masing. Kemudian akhirnya ditemui banyaknya keterbatasan dari pendekatan menggunakan file, antara lain:

1. Data yang terpisah dan terisolasi
2. Duplikasi data

3. Ketergantungan data
4. Format file yang tidak kompatibel
5. *Query* yang tetap
6. Tidak dapat mengantisipasi perkembangan program aplikasi

Kemudian akhirnya digunakanlah pendekatan dengan menggunakan basis data dan *Database Management Sistem* (DBMS).

2.2.1 Pengertian Basis Data

Menurut Connolly (2002,p14) basis data adalah suatu kumpulan data logis yang terhubung satu sama lain, dan deskripsi dari suatu data yang dirancang sebagai informasi yang dibutuhkan oleh organisasi. Sedangkan menurut Date (2000,p10), basis data adalah suatu koleksi data yang tetap yang digunakan oleh sistem aplikasi dari suatu perusahaan.

Menurut Mannino(2004,p4), basis data memiliki 3 karakteristik, di antaranya :

1. *Persistent*. Data disimpan dalam tempat penyimpanan yang stabil, seperti disk magnetik. Ketetapan data tersebut tidak berarti bahwa data tersebut ada selamanya. Ketika data sudah tidak relevan lagi, maka data tersebut akan disingkirkan.
2. *Shared*. Suatu basis data dapat mempunyai kegunaan dan pengguna yang banyak. Suatu basis data menyediakan memori umum untuk fungsi-fungsi dalam suatu organisasi. Basis data dapat digunakan oleh lebih dari 1 orang dalam waktu yang bersamaan. Apabila ada dua pengguna yang ingin mengubah bagian yang sama dalam basis data, maka mereka harus saling menunggu. Sebagai contoh, banyak mahasiswa Binus dapat melakukan registrasi KRS secara bersamaan

3. *Interrelated*. Data disimpan dalam sebagai bagian-bagian yang terpisah yang saling terhubung satu dengan yang lainnya.

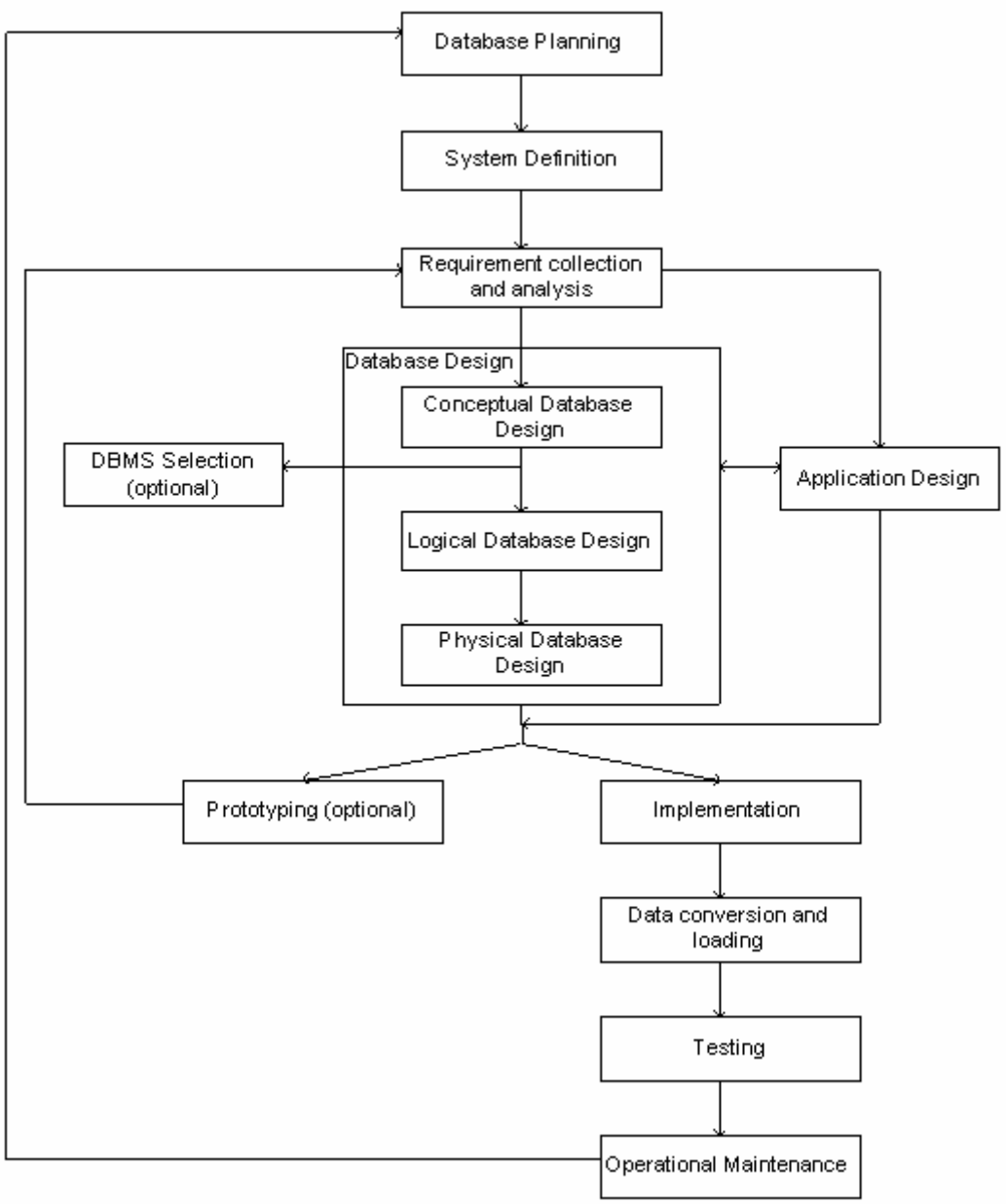
2.2.2 Daur Hidup Aplikasi Basis Data

Tahapan-tahapan perancangan yang digunakan untuk membangun suatu aplikasi basis data menurut Connolly (2002,p272) disebut dengan daur hidup aplikasi basis data (*The Database Application Lifecycle*).

Tahapan daur hidup aplikasi basis data :

1. *Database Planning* : merencanakan bagaimana tahapan-tahapan dari daur hidup ini dapat terealisasi secara efisien dan efektif.
2. *System Definition* : menspesifikasikan jangkauan dan batasan aplikasi basis data, penggunanya, dan area aplikasi.
3. *Requirements Collection and Analysis* : pengumpulan dan analisa kebutuhan penggunanya dan area aplikasi.
4. *Database Design* : perancangan basis data secara konseptual, logikal, dan fisikal.
5. *DBMS Selection (optional)* : memilih DBMS yang sesuai untuk aplikasi basis data.
6. *Application Design* : merancang antar muka dan program aplikasi yang menggunakan dan memproses basis data.
7. *Prototyping (optional)* : membangun model kerja dari aplikasi basis data, yang memperbolehkan perancang atau pengguna untuk memvisualisasikan dan mengevaluasi bagaimana sistem akhirnya akan tampak dan berfungsi.
8. *Implementation* : membuat definisi basis data eksternal, konseptual, dan internal serta program aplikasi.

9. *Data Conversion and loading* : *loading* data dari sistem yang lama ke sistem yang baru.
10. *Testing* : pengetesan aplikasi basis data untuk mencari kesalahan dan divalidasi untuk kebutuhan yang dispesifikasikan oleh pengguna.
11. *Operational Maintenance*: aplikasi data telah diimplementasikan sepenuhnya. Sistem diawasi dan dipelihara secara berkelanjutan. Ketika diperlukan, kebutuhan-kebutuhan baru dimasukan dalam aplikasi basis data melalui tahapan basis data terdahulu.



Gambar 2.2 Daur Hidup Aplikasi Basis Data

2.2.3 Perancangan Basis Data

2.2.3.1 *Data Modelling*

Kriteria *Data Modelling*:

1. *Structural Validity* : konsistensi dengan cara perusahaan mendefinisikan dan mengatur informasi
2. *Simplicity* : mempermudah pengertian oleh profesional Sistem Informasi dan pengguna non-teknikal
3. *Expressibility* : kemampuan untuk membedakan antara data, *relationship* antara data dan *constraint*
4. *Nonredudancy* : menampilkan tiap informasi hanya satu kali, tanpa ada pengulangan
5. *Shareability* : tidak spesifik untuk aplikasi/teknologi tertentu dan dapat digunakan oleh banyak orang
6. *Extensibility* : kemampuan untuk mengembangkan untuk mendukung kebutuhan baru dengan efek minimal pada pengguna yang sudah ada
7. *Integrity* : konsistensi cara perusahaan menggunakan dan mengatur informasi
8. *Diagrammatic representation*: kemampuan untuk menampilkan sebuah model dengan menggunakan diagram yang mudah dimengerti

2.2.3.2 Tahap-Tahap Perancangan Basis Data

Menurut Connolly (2002,p279), perancangan basis data adalah proses membuat suatu desain untuk sebuah basis data yang mendukung kegiatan operasional suatu perusahaan.

Tahapan proses perancangan :

- ***Conceptual Database Design***

  1. Membuat model data konseptual lokal untuk setiap *view*
     - 1.1. Identifikasi tipe entiti
     - 1.2. Identifikasi tipe *relationship*
     - 1.3. Identifikasi dan asosiasikan atribut dengan entiti dan *relationship*
     - 1.4. Menentukan domain atribut
     - 1.5. Menentukan atribut *candidate key* dan *primary key*
     - 1.6. Pertimbangkan penggunaan konsep *enhanced modelling*
     - 1.7. Cek model untuk redudansi
     - 1.8. Validasikan model konseptual lokal terhadap transaksi *user*
     - 1.9. *Review* model data konseptual lokal dengan *user*

- ***Logical Database Design***

  2. Membuat dan memvalidasikan model data logikal lokal untuk setiap *view*
     - 2.1. Menghilangkan fitur-fitur yang tidak kompatibel dengan model relasional (*optional*)
     - 2.2. Mendapatkan relasi untuk model data logikal lokal

2.3. Validasi relasi dengan menggunakan proses normalisasi

2.4. Validasi relasi terhadap transaksi user

2.5. Menetapkan *integrity constraints*

2.6. *Review* model data logikal lokal dengan *user*

3. Membuat dan memvalidasikan model data logikal global

3.1. Gabungkan semua model data logikal lokal ke dalam model global

3.2. Validasi model data logikal global

3.3. Cek untuk perkembangan di masa depan

3.4. *Review* model data logikal global dengan *user*

- ***Physical Database Design***

4. Menerjemahkan model data logikal global untuk DBMS yang ditargetkan

4.1. Desain relasi dasar

4.2. Mendesain representasi *der1ived data*

4.3. Desain *enterprise constraints*

5. Desain representasi fisikal

5.1. Analisa transaksi

5.2. Pilih organisasi file

5.3. Pilih indeks

5.4. Perkirakan kebutuhan *disk space*

6. Desain *user view*

7. Desain mekanisme keamanan

8. Pertimbangkan pengenalan redudansi terkontrol

9. Memonitor dan memelihara sistem operasional

2.2.4 *Database Management System* (DBMS)

Definisi DBMS menurut Connolly (2002,p16) adalah suatu sistem perangkat lunak yang bisa mendefinisikan, membuat, memelihara, dan mengontrol akses ke basis data. Biasanya, suatu DBMS mempunyai fasilitas seperti berikut ini:

- Terdapat fasilitas untuk mendefinisikan basis data, biasanya menggunakan suatu *Data Definition Language* (DDL). Suatu DDL memberikan fasilitas kepada pengguna untuk menspesifikasikan tipe data dan strukturnya dan batasan aturan mengenai data yang bisa disimpan ke dalam basis data tersebut.
- Terdapat fasilitas yang memperbolehkan *user* untuk menambah, mengedit, menghapus data, dan mendapatkan kembali data. Biasanya dengan menggunakan suatu *Data Manipulation Language* (DML). Biasanya ada suatu fasilitas untuk melayangi mengaksesan data yang disebut sebagai bahasa Query. Bahasa Query yang paling diakui adalah Structured Query Language (SQL), yang secara de fakto merupakan standar bagi DBMS.
- Terdapat fasilitas untuk mengontrol akses ke basis data, sebagai contoh:
  - Suatu sistem keamanan yang mencegah pengguna yang tidak punya otoritas untuk mengakses data
  - Suatu sistem terintegrasi yang mana memelihara konsistensi penyimpanan data

- Suatu sistem kontrol yang mana memperbolehkan akses ke basis data
- Suatu sistem kontrol pengembalian data yang mana dapat mengembalikan data pada keadaan sebelumnya apabila terjadi kegagalan perangkat keras atau perangkat lunak
- Terdapat suatu katalog yang dapat diakses oleh *user* yang mana mendeskripsikan data di dalam basis data tersebut

2.2.5 Sistem Basis Data

Suatu sistem basis data menurut Date (1999,p5) adalah sistem yang terkomputerisasi yang semua tujuannya untuk menyimpan informasi dan membolehkan pengguna untuk mengambil data kembali dan memperbaharui informasi yang diperlukan.

Menurut Subekti (1997,p3), berbagai keuntungan dan kerugian yang diperoleh dari hasil penerapan manajemen basis data pada suatu perusahaan meliputi beberapa hal berikut:

Keuntungan:

- Kontrol terpusat data operasional
- Redundansi data dapat dikurangi dan dikontrol
- Ketidakkonsistenan data dapat dihindarkan
- Data dapat dipakai bersama (*sharing*)
- Penerapan standarisasi
- Penerapan pembatasan keamanan data (*security*)
- Integritas data dapat dipelihara

- Kebutuhan yang berbeda dapat diselaraskan
- Independensi data/program

Kerugian:

- Mahal, karena membutuhkan biaya yang lebih besar untuk perangkat keras, perangkat lunak, dan personil yang lebih berkualitas
- Kompleks, karena kemampuan perangkat lunak yang lebih besar, menjadi terlihat lebih rumit dan penguasaan yang lebih tinggi antara lain untuk kebutuhan-kebutuhan:
  - Sistem Administrasi
  - Prosedur *Recovery*
  - Prosedur *Backup*
  - Penataan Keamanan Data

2.2.6 *Entity-Relationship*

Entity-Relationship Modelling adalah pendekatan *top-down* terhadap desain basis data yang dimulai dengan mengidentifikasi data penting yang disebut sebagai entiti dan relasi antar data yang disebut sebagai *relationships* yang akan direpresentasikan dalam model. Kemudian akan ditambahkan detil-detil seperti informasi yang diperlukan mengenai entiti dan *relationships* tersebut yang disebut sebagai atribut.

Menurut Kroenke (2002,p52), ERD terdiri atas 4 elemen yaitu: entiti, atribut, *identifier*, dan *relationship*.

- Entiti adalah sekumpulan obyek dengan properti sama yang mempunyai keberadaan yang independen.

- *Relationship* adalah suatu set asosiasi antara satu atau lebih entiti yang berpartisipasi.
- Atribut adalah properti dari suatu entiti atau *relationship*. Atribut memegang nilai yang akan menjelaskan setiap kejadian entiti.

Ada beberapa jenis atribut, antara lain :

- *Simple Attributes.* Atribut yang dibentuk dari komponen sederhana yang memiliki keberadaan yang independen.
- *Composite Attributes.* Atribut yang dibentuk dari komponen jamak, yang masing-masingnya memiliki keberadaan yang independen.
- *Single-valued Attributes.* Atribut yang memiliki hanya 1 nilai untuk setiap kejadian dari suatu entiti.
- *Multi-valued Attributes.* Atribut yang memiliki lebih dari 1 nilai untuk setiap kejadian dari suatu entiti.
- *Derived Attributes.* Atribut yang memiliki nilai yang berasal dari nilai atribut lain yang berelasi.

2.2.7 *Relational Model*

2.2.7.1 *Relational Data Structure*

Relation (relasi) : suatu tabel yang terdiri dari kolom dan baris

Attribute (atribut) : kolom yang mempunyai nama yang terdapat dalam suatu relasi. Contoh : relasi (tabel) mahasiswa mempunyai atribut NIM, nama, alamat, dan seterusnya

Tuple : baris dalam suatu relasi. *Tuple* sendiri merupakan suatu *occurence* (kejadian) yang terdapat dalam suatu relasi

Degree (derajat) : jumlah atribut yang dimiliki oleh suatu relasi

Cardinality : jumlah *tuple* yang dimiliki oleh suatu relasi

2.2.7.2 *Relational Keys*

Fungsi dari kunci relasional adalah untuk mengidentifikasi setiap *tuple* dalam suatu relasi secara unik**.** Ada beberapa jenis kunci relasional, antara lain:

- *Superkey*

  Satu atau sekumpulan atribut yang mengidentifikasi *tuple* dalam suatu relasi secara unik

- *Candidate key*

  Suatu *superkey* yang tidak memiliki subset *superkey* lain dalam suatu relasi. Ada dua properti *candidate key*, yaitu *uniqueness* dan *irreducibility*

- *Primary key*

  Suatu *candidate key* yang telah terpilih untuk mengidentifikasikan suatu *tuple* dalam suatu relasi secara unik

- *Alternate key*

  Candidate key yang tidak terpilih untuk mengidentifikasikan suatu *tuple* dalam suatu relasi secara unik

- *Foreign key*

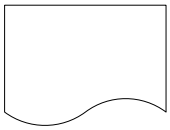
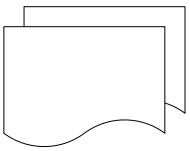
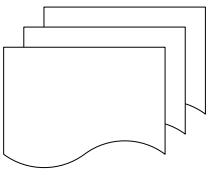
  Satu atau kumpulan atribut dalam suatu relasi yang mirip atau sama dengan *candidate key* yang terdapat dalam satu atau beberapa relasi lain

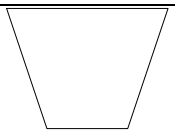
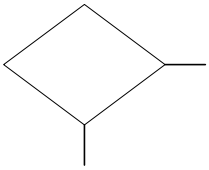
- *Composite key*

  Candidate key yang mempunyai lebih dari satu atribut

2.2.8 Diagram Alir

Diagram alir adalah salah satu metode untuk menggambarkan proses kerja dalam suatu perusahaan atau organisasi. Berikut ini adalah simbol-simbol standar diagram alir dengan maknanya masing-masing:

Simbol	Nama	Keterangan
	Dokumen	Menggambarkan semua jenis dokumen, yang merupakan formulir yang digunakan untuk merekam data terjadinya suatu transaksi
	Dokumen dan tembusannya	Menggambarkan dokumen asli dan tembusannya. Nomor lembar dokumen dicantumkan di sudut kanan atas
	Berbagai dokumen	Menggambarkan berbagai jenis dokumen yang digabungkan bersama di dalam suatu paket. Nama dokumen dituliskan di dalam masing-masing simbol dan nomor lembar dokumen dicantumkan disudut kanan atas simbol dokumen yang bersangkutan
	Penghubung pada halaman yang berbeda	Menunjukkan kemana dan bagaimana bagan alir terkait satu dengan lainnya. Nomor yang tercantum dalam simbol penghubung

	(*off-page connector*)	menunjukkan bagaimana bagan alir yang tercantum pada halaman yang lain
	Penghubung pada halaman yang sama (*on-page connector*)	Karena keterbatasan ruang halaman kertas untuk menggambar, maka diperlukan penghubung untuk memungkinkan aliran dokumen berhenti di suatu lokasi pada halaman tertentu dan kembali berjalan di lokasi lain pada halaman yang sama. Dengan memperlihatkan nomor yang tercantum di dalam simbol penghubung pada halaman yang sama, dapat diketahui aliran dokumen dalam sistem yang digambarkan dalam bagan alir
	Kegiatan manual	Uraian singkat kegiatan manual dicantumkan di dalam simbol ini
	Arsip permanen	Menggambarkan arsip permanen yang merupakan tempat penyimpanan dokumen yang tidak akan diproses lagi dalam sistem yang bersangkutan
	Mulai/berakhir (*terminal*)	Menggambarkan awal dan akhir suatu sistem
	Keputusan	Menggambarkan keputusan yang harus dibuat dalam proses pengolahan data. Keputusan yang dibuat ditulis dalam simbol.

2.2.9 *Data Definition Language* (DDL)

Menurut Martina (2003,p58), DDL merupakan bagian dari sistem manajemen basis data, dipakai untuk mendefinisikan dan mengatur semua atribut dan properti dari sebuah basis data. DDL digunakan untuk mendefinisikan basis data, tabel, dan *view*.

Sedangkan menurut Subekti (1997,p20), DDL adalah bahasa yang dipakai untuk menjelaskan obyek dari basis data seperti terlihat oleh pengguna (DBA, *Programmer*, Pengguna akhir). DDL dipakai untuk mendefinisikan kerangka basis data (berorientasi pada tipe dari obyek basis data).

1. *Create Table*

   Pernyataan *create table* digunakan untuk membuat tabel dengan mengidentifikasikan tipe data untuk tiap kolom.

   Bentuk umum:

   CREATE TABLE Table_name

   (Column_name DataType[NULL | NOT NULL]

   [, Column_name DataType [NULL | NOT NULL]] ...)

2. *Alter Table*

   Pernyataan *alter table* digunakan dapat dipakai untuk menambah atau membuang kolom dengan *constraint*.

   Bentuk umum:

   ALTER TABLE Table_name

   [ADD Column_name DataType[NULL | NOT NULL]]

   [DROP Column_name DataType [RESTRICT | CASCADE]]

   [ADD Constrain_name]

[DROP Constrain_name [RESTRICT | CASCADE]]

3. *Drop Table*

   Pernyataan *drop table* digunakan untuk membuang/menghapus tabel beserta semua data yang terkait di dalamnya

   Bentuk umum:

   DROP TABLE Table_name;

4. *Create Index*

   Pernyataan *create index* digunakan untuk membuat indeks pada suatu tabel

   Bentuk umum:

   CREATE [UNIQUE] INDEX index_name

   ON table_name

   (Column_name [, Column_name]...)

5. *Drop Index*

   Pernyataan *drop index* digunakan untuk membuang atau menghapus indeks yang telah dibuat sebelumnya.

   Bentuk umum:

   DROP INDEX Index_name

2.2.10 *Data Manipulation Language* (DML)

DML menurut Subekti (1997,p21) adalah bahasa yang dipakai untuk memanipulasi (memasukkan, mengubah, menghapus) obyek data dari basis data. DML dipakai untuk operasi terhadap isi basis data, jadi berorientasi pada *occurence* basis data.

Beberapa bentuk operasi basis data relasional:

- SELECT : mencari record (baris-baris) dari basis data. Selain itu, dapat juga untuk mencari record dari beberapa tabel.

  SELECT [DISTINCT | ALL] {* | [ekpresiKolom [AS namaBaru]] [, ...]}

  FROM nama_tabel [alias] [...]

  [WHERE kondisi]

  [GROUP BY daftarKolom] [HAVING kondisi]

  [ORDER BY daftarKolom]

  select nim, nama from mhs where kdj = "KA"

- INSERT : menambahkan tuple (rekord) dari *working-area* ke suatu relasi.

  INSERT

  INTO nama_tabel [(nama_field) [, nama_field) [, (...)]]]

  VALUE (literal [, literal [,.....]] ;

  Atau

  INSERT

  INTO nama_tabel [(nama_field) [, nama_field) [, (...)]]]

  Subquery ;

  insert into mhs value ("109", "johny", "L", "01-MAR-99","KA")

- DELETE: menghapus *tuple* (rekord) dari suatu relasi

  DELETE FROM nama-tabel

  [WHERE kondisi] ;

  delete daf_nilai where nim = "101" and kode_mk = "X01"

- UPDATE: mengubah isi *field* pada *tuple* tertentu

  UPDATE nama_tabel

  SET nama-field = ekspresi skalar

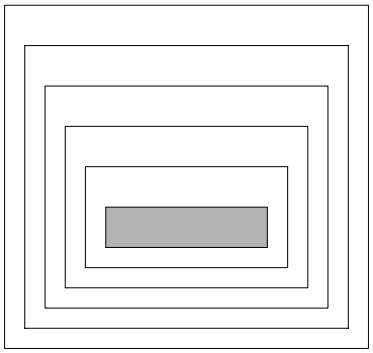
[, nama-field = ekspresi skalar] ...

[WHERE kondisi] ;

update mhs where nim = "101" set kode_pos = "12450"

2.2.11 Normalisasi

Normalisasi adalah pendekatan *bottom-up* dalam perancangan basis data dengan menganalisa *relationship* antar atribut. Merupakan teknik yang digunakan untuk menganalisa relasi berdasarkan *primary key* (atau *candidate keys*) dan ketergantungan fungsional.



Gambar 2.3 Level Normalisasi

Proses normalisasi sendiri terdiri dari tahapan-tahapan sebagai berikut :

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

   Proses pertama adalah dengan mentransfer data dari sumber ke dalam suatu format tabel yang terdiri dari baris dan kolom.

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

   Suatu relasi dimana persimpangan dari tiap baris dan kolom (*field*) hanya berisi 1 nilai saja dan tidak berulang. Hal ini dapat dilakukan dengan mengidentifikasi

dan menghilangkan *repeating groups* dari tabel *unnormalized*, yang merupakan hasil dari UNF. *Repeating groups* adalah suatu atribut atau grup atribut dengan nilai jamak untuk 1 kejadian, dimana atribut tersebut merupakan calon atributt kunci untuk tabel tersebut.

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

   Suatu relasi dimana di 1NF atribut bukan *primary key* bergantung secara fungsional penuh terhadap *primary key*. Hal ini dapat dilakukan dengan menghilangkan ketergantungan parsial. Kita dapat menghilangkan atribut yang bergantung secara fungsional penuh dalam relasi dengan menempatkan atribut tersebut dalam relasi baru.

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

   Suatu relasi dimana tidak ada atribut bukan *primary key* yang bergantung secara transitif terhadap *primary key*. Hal ini dapat dilakukan dengan menghilangkan ketergantungan transitif. Kita dapat menghilangkan atribut yang bergantung secara transitif dalam relasi dengan menempatkan atribut tersebut dalam relasi baru.

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

   Suatu relasi dimana jika dan hanya jika, setiap determinan adalah *candidate key*.

2.3. Teori-teori tentang pembelian dan *cashflow*

2.3.1 Pembelian

2.3.1.1 Definisi Pembelian

Menurut Mulyadi (1997,p301), pembelian adalah suatu usaha yang digunakan dalam perusahaan untuk pengadaan barang yang diperlukan oleh perusahaan.

Menurut Assauri (1990,p205), pembelian merupakan salah satu fungsi yang paling penting dalam berhasilnya operasi suatu perusahaan yang dibebani tanggung jawab untuk mendapat kuantitas dan kualitas bahan-bahan yang tersedia pada waktu dibutuhkan dengan harga yang sesuai dengan harga yang berlaku.

2.3.1.2 Jenis-jenis pembelian

Berdasarkan jenis transaksinya, pembelian dibedakan menjadi 2, yaitu:

1. Pembelian tunai, yaitu jenis transaksi dalam pembayaran langsung dilakukan pada saat penerimaan barang.
2. Pembelian kredit, yaitu jenis transaksi dimana pembayaran tidak dilakukan pada saat penyerahan barang, tapi dilakukan selang beberapa waktu sesuai perjanjian dengan pihak pemasok

Sedangkan berdasarkan jenis pemasok, pembelian dapat dibedakan menjadi 2, yaitu:

1. Pembelian lokal, yaitu pembelian yang dilakukan dari pemasok dalam negeri
2. Pembelian impor, yaitu pembelian yang dilakukan dari pemasok luar negeri

2.3.1.3 Fungsi yang terkait dalam pembelian

Menurut Mulyadi (1997,p302) fungsi yang terkait dalam sistem pembelian adalah:

1. Fungsi gudang, bertanggung jawab dalam mengajukan permintaan pembelian sesuai dengan posisi persediaan yang ada di gudang dan untuk menyimpan barang yang telah diterima oleh fungsi penerimaan
2. Fungsi pembelian, bertanggung jawab untuk memperoleh informasi mengenai harga barang, menentukan pemasok yang dipilih dalam pengadaan barang, dan mengeluarkan *purchase order* kepada pemasok yang dipilih
3. Fungsi penerimaan, bertanggung jawab untuk melakukan pemeriksaan terhadap jenis, mutu, kuantitas bahan yang diterima dari pemasok guna menentukan dapat tidaknya barang tersebut diterima oleh perusahaan
4. Fungsi akuntansi yang berkaitan dengan transaksi pembelian adalah fungsi pencatatan hutang yang bertanggung jawab untuk mencatat transaksi pembelian ke register bukti kas keluar untuk menyelenggarakan arsip dokumen sumber bukti kas keluar yang berfungsi sebagai catatan hutang atau menyelenggarakan kartu hutang sebagai buku pembantu hutang

2.3.2 *Cashflow*

Menurut Edmonds (1998,p574), *cashflow* menyatakan bagaimana suatu perusahaan megelola, menerima dan menggunakan kas selama periode tertentu. Sumber kas biasanya dikenal sebagai *cash inflows* dan bagian kas yang digunakan

disebut sebagai *cash outflows*. Arus kas mengklasifikasikan *receipts* (*inflows*) dan *payments* (*outflows*) ke dalam tiga kategori, antara lain :

- *Operating activities*

  Aktivitas bisnis yang menghasilkan *cash inflows* dan *cash outflows* (arus kas) melalui proses pengoperasian bisnis.

- *Investing activities*

  Aktivitas bisnis yang menghasilkan arus kas melalui transaksi pembelian perusahaan atau pembelian aset operasional jangka panjang, dan investasi terhadap perusahaan lain.

- *Financing activities*

  Aktivitas bisnis yang menghasilkan arus kas melalui proses transaksi dengan pemilik perusahaan.