BAB 2

LANDASAN TEORI

2.1 Teori-Teori Basis Data

2.1.1 Pengertian Sistem

Menurut Hall (2001, p5) pengertian sebuah sistem adalah sekelompok dua atau lebih komponen-komponen yang saling berkaitan atau subsistem-subsistem yang bersatu untuk mencapai tujuan yang sama.

Menurut Mulyadi (1997, p2), suatu sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu sama lainnya, yang berfungsi bersama-sama untuk mencapai suatu tujuan tertentu.

Menurut McLeod (2001, p9) sistem adalah sekumpulan elemen yang diintegrasikan dengan maksud yang sama untuk mencapai suatu tujuan.

2.1.2 Data

Menurut Elmasri (2000 , p4), data didefinisikan sebagai fakta yang dapat dicatat dan memiliki arti implisit. Menurut Mcleod (2001, p12), data terdiri dari fakta-fakta dan figur-figur yang relatif tidak bermakna bagi pengguna. Agar dapat digunakan, data diproses sedemikian rupa ke dalam bentuk informasi.

2.1.3 Sistem Berbasis *File*

2.1.3.1 Definisi *Data Field*

Data field adalah unit data yang terkecil (McLeod, R, 2001, p173). Contohnya NIK (Nomor Induk Karyawan).

2.1.3.2 Definisi *Record*

Record adalah kumpulan *data field* yang saling berhubungan (McLeod, R, 2001, p173).

2.1.3.3 Definisi *File*

Menurut McLeod, R (2001, p173), *file* adalah kumpulan *record* yang saling berhubungan satu sama lain.

2.1.3.4 Definisi Sistem Berbasis File

Connolly & Begg (2002, p7) mendefinisikan sistem berbasis file sebagai sekumpulan program aplikasi yang memberikan pelayanan pada pengguna. Misalnya membuat laporan, menampilkan data, manipulasi data. Setiap program mendefinisikan dan mengatur datanya sendiri-sendiri.

2.1.4 Basis Data

2.1.4.1 Konsep Basis Data

McLeod, R (2001, p182) menjelaskan konsep basis data sebagai sebuah integrasi logis atas sejumlah *file*.

2.1.4.2 Definisi Basis Data dan Sistem Basis Data

Menurut Elmasri (2000 , p4), basis data adalah kumpulan data yang saling berhubungan. Pengertian basis data menurut Mcleod, R(2000 , p16-17) adalah suatu koleksi data komputer yang terintegrasi, diatur dan disimpan dengan suatu cara yang memudahkan untuk diambil kembali.

Menurut Date (2000, p10) database merupakan kumpulan data persistent yang digunakan oleh sistem aplikasi pada beberapa perusahaan.

Database menurut Connolly & Begg (2002, p14), adalah *"a shared collection of logically related data, and a description of this data,*

designed to meet the information needs of an organization". Dalam bahasa Indonesia, definisi tersebut dapat dikatakan sebagai berikut : "kumpulan data yang terhubung secara logis, beserta deskripsi data tersebut, yang dirancang untuk memenuhi kebutuhan informasi sebuah organisasi"

Dalam praktek, penggunaan istilah basis data menurut Elmasri lebih dibatasi pada arti implisit yang khusus yaitu:

1. Basis data merupakan penyajian suatu aspek dari dunia nyata. Terkadang disebut "*miniworld "* atau "*Universe of Disclosure*". Perubahan-perubahan yang terjadi pada *miniworld* direfleksikan pada database.
2. Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implisit. Sehingga data yang terkumpul secara acak dan tanpa mempunyai arti tidak dapat disebut basis data.
3. Basis data perlu dirancang, dibangun, dan data dikumpulkan untuk suatu tujuan. Basis data dapat digunakan oleh beberapa pemakai dan beberapa aplikasi yang sesuai dengan kepentingan pemakai.

Sedangkan pengertian sistem basis data adalah komputerisasi sistem penyimpanan data yang bertujuan menyimpan dan memelihara informasi serta mengizinkan user untuk melakukan pengupdatean atau pengambilan data yang dibutuhkan (Date, 2000, p5).

2.1.4.3 Tujuan Basis Data

Menurut Mcleod (2001, p182), dua tujuan konsep basis data adalah meminimisasi pengulangan data dan mencapai independensi data. Pengulangan data (*data redundancy*) adalah duplikasi data, dengan kata lain

terdapat data yang sama di beberapa file dan lokasi (). Adanya redundansi data ini dapat menyebabkan *data inconsistency*. *Data inconsistency* adalah keadaan dimana terjadi ketidakcocokan nilai / *value* pada *item* yang sama.

Independensi data adalah kemampuan untuk mengubah struktur pada data tanpa mengubah program aplikasi yang akan memproses data tersebut. Independensi data dapat dicapai dengan menempatkan spesifikasi data dalam tabel dan kamus yang terpisah secara fisik dari program. Perubahan pada suatu data hanya dilakukan sekali, yaitu dalam tabel.

2.1.4.4 Database Administrator (DBA)

Menurut Connolly & Begg (2002, p21), Database Administrator adalah orang yang bertanggung jawab atas realisasi fisik dari database. Termasuk di dalamnya adalah perancangan database fisikal dan implementasinya, keamanan, pengaturan integrity, pemeliharaan sistem operasional, dan menjamin kinerja aplikasi yang memuaskan bagi pengguna.

2.1.5 *Database Management System* (DBMS)

McLeod, R (2001, p178) mendefinisikan DBMS sebagai aplikasi piranti lunak yang menyimpan struktur dari basis data tersebut, data yang terkandung di dalamnya, hubungan antara data dalam basis data, dan juga berkas-berkas dan laporan-laporan yang berhubungan dengan basis data yang bersangkutan.

Definisi DBMS menurut Elmasri (2000, p5) adalah sekumpulan program yang memungkinkan user membuat dan mengurus basis data. Dengan kata lain, DBMS adalah sistem piranti lunak yang memfasilitasi

proses mendefinisikan, membangun, dan memanipulasi basis data untuk berbagai aplikasi.

Proses mendefinisikan basis data melibatkan penspesifikasian tipe data, struktur dan aturan-aturan untuk penyimpanan data di sistem basis data. Proses pembangunan basis data meliputi proses menyimpan data ke dalam media penyimpanan yang berada di bawah pengaturan DBMS. Manipulasi basis data mencakup fungsi-fungsi seperti melakukan query atas basis data untuk mendapatkan data tertentu, mengubah isi basis data, dan menghasilkan laporan dari data-data tersebut.

Mengutip Connolly & Begg (2002, p48-p52), disebutkan bahwa Codd (1982) menyebutkan delapan fungsi DBMS yang harus disediakan oleh *full-scale DBMS* manapun. Connolly & Begg juga menambahkan dua fungsi tambahan yang diharapkan ada. Kesepuluh fungsi tersebut disebutkan di bawah ini :

1. Penyimpanan, pengambilan kembali, dan pengubahan data (s*torage*, *retrieval* dan *update*)
2. Katalog yang dapat diakses oleh pengguna : Katalog berisi deskripsi *item* data yang disimpan.
3. Mendukung transaksi : DBMS harus menyediakan mekanisme yang akan menjamin bahwa jika terjadi pengubahan pada data, seluruh data yang berkaitan akan ikut diubah, dan jika tidak terjadi pengubahan, tidak ada data yang berubah.
4. Layanan pengendalian concurrency : DBMS harus menyediakan mekanisme untuk memastikan bahwa database diupdate secara benar

ketika sejumlah pengguna mengubah database tersebut secara bersama-sama.

5. Layanan recovery : DBMS harus menyediakan mekanisme untuk me-recover database ketika terjadi kerusakan dalam segi apapun.
6. Layanan otorisasi : DBMS harus menyediakan mekanisme untuk memastikan bahwa hanya pengguna yang berhaklah yang dapat mengakses database.
7. Mendukung komunikasi data : DBMS harus mampu berintegrasi dengan piranti lunak komunikasi. Misalnya data antar jaringan, dll.
8. Layanan integrity : DBMS harus menyediakan mekanisme untuk memastikan bahwa data di database dan perubahan pada data tersebut mengikuti aturan-aturan tertentu.
9. Layanan untuk mendukung independensi data : DBMS harus menyediakan fasilitas untuk mendukung independensi program dari struktur asli database.
10. Layanan alat tambahan (*utility*) : DBMS harus menyediakan layanan alat-alat tambahan untuk memudahkan tugas DBA.

Menurut Petroutsos (2002, p5), DBMS menyediakan fungsi – fungsi sebagai berikut :

1. DBMS mengijinkan aplikasi mendefinisikan struktur dari basis data dengan pernyataan SQL. Pernyataan SQL ini disebut *Data Definition Language* (DDL).
2. DBMS mengijinkan aplikasi memanipulasi informasi yang disimpan didalam basis data dengan pernyataan SQL. Pernyataan SQL yang

memanipulasi informasi ini disebut dengan *Data Manipulation Language* (DML).

3. DBMS melindungi integritas basis data dengan menerapkan beberapa aturan, yang dimaksudkan kedalam perancangan basis data tersebut.

2.1.5.1 Arsitektur DBMS

Elmasri (2000, p27) membagi DBMS menjadi arsitektur yang terdiri dari tiga skema :

1. Skema internal

   Skema ini mendefinisikan struktur penyimpanan fisik dari database tersebut. Skema ini menggunakan model data fisik dan mendeskripsikan rincian lengkap mengenai penyimpanan data dan *path* akses ke database.

2. Skema konseptual

   Skema ini mendefinisikan keseluruhan struktur database untuk semua pengguna. Skema ini menyembunyikan rincian struktur fisik database dan berfokus pada entitas-entitas, tipe data, hubungan, operasi *user*, dan *constraints* (batasan-batasan).

3. Skema eksternal / tingkatan "*view*"

   Skema ini menunjukkan bagian-bagian dari database yang digunakan kelompok pengguna tertentu dan menyembunyikan bagian database lain yang tidak berkaitan dengan user tersebut.

2.1.5.2 Komponen – komponen DBMS

Menurut Connolly & Begg (2002, p18-20), *Database Management System* (DBMS) memiliki 5 komponen penting yaitu :

1. *Hardware* (perangkat keras)

   Dalam menjalankan aplikasi dan DBMS diperlukan perangkat keras. Perangkat keras dapat berupa komputer pribadi, mainframe, atau *berupa* jaringan komputer.

2. *Software* (perangkat lunak)

   Komponen perangkat lunak meliputi *DBMS software* dan program aplikasi beserta *operating system*-nya, termasuk perangkat lunak jaringan bila DBMS digunakan dalam jaringan seperti LAN.

3. Data

   Data mungkin merupakan komponen terpenting dari DBMS khususnya dipandang dari sisi *end- user*.

4. Prosedur

   Prosedur berupa panduan dan instruksi yang mengatur perancangan dan penggunaan basis data. Pengguna sistem dan staff pengelola basis data membutuhkan instruksi dan prosedur dalam menjalankan sistem dan mengolah basis data itu sendiri.

   Contoh prosedur tersebut antara lain dapat berupa panduan bagaimana cara untuk :

   - login di dalam basis data
   - menggunakan sebagian fasilitas aplikasi DBMS
   - menjalankan dan menghentikan DBMS

- membuat salinan *backup* database
- menangani kegagalan piranti lunak maupun keras
- mengubah struktur basis data, meningkatkan kinerja atau membuat arsip data pada media penyimpanan sekunder.

5. Manusia

   Komponen yang terakhir yaitu manusia sendiri yang terlibat dalam sistem tersebut.

2.1.5.3 Keuntungan dan Kerugian DBMS

Menurut McLeod, R (2001, p192-193), DBMS memungkinkan pembuatan basis data dalam penyimpanan akses langsung komputer, pemeliharaan isinya, dan penyediaan isi tersebut bagi pemakai tanpa pemrograman khusus yang mahal.

Ketika perusahaan atau pemakai individu memutuskan apakah akan menggunakan suatu DBMS, keuntungan dan kerugiannya harus dipertimbangkan. Menurut McLeod (2001, p193), keuntungan DBMS adalah sebagai berikut :

1. Mengurangi pengulangan data

   Jumlah total file dikurangi dengan menghapus file-file duplikat. Juga hanya terdapat sedikit data yang sama di beberapa file.

2. Mendapatkan independensi data

   Spesifikasi data disimpan dalam skema tiap program aplikasi. Perubahan dapat dibuat pada struktur data tanpa mempengaruhi program yang mengakses data.

3. Mengintegrasikan data dari sejumlah file

   Ketika file dibuat untuk membentuk kaitan logis, pengaturan secara fisik tidak lagi menjadi kendala. Pengaturan logikal, view bagi tiap user dan program aplikasi itu sendiri tidak harus direpresentasikan secara langsung di media penyimpanan fisik.

4. Mengambil data dan informasi secara cepat

   Hubungan-hubungan logis dan DML serta *Query language* memungkinkan pemakai mengambil data dalam hitungan detik atau menit, yang sebelumnya memungkinkan memerlukan beberapa jam atau bahkan berhari-hari.

5. Meningkatkan keamanan

   Baik DBMS *mainframe* maupun komputer mikro dapat menyertakan beberapa lapis keamanan seperti kata sandi (*password*), direktori pemakai, dan penyandian (*encryption*). Data yang dikelola oleh DBMS juga lebih aman daripada kebanyakan data lain dalam perusahaan.

Sedangkan kerugian dari DBMS adalah sebagai berikut :

1. Piranti lunak yang mahal

   DBMS *mainframe* masih sangat mahal. DMBS berbasis komputer mikro, walaupun biayanya hanya beberapa ratus dolar, merupakan pengeluaran yang besar bagi organisasi kecil.

2. Konfigurasi perangkat keras berskala besar

   DBMS sering memerlukan kapasitas penyimpanan primer dan sekunder yang lebih besar daripada yang diperlukan oleh program

aplikasi lain. Juga kemudahan yang dibuat oleh DBMS dalam mengambil informasi mendorong lebih banyak terminal pemakai yang disertakan dalam konfigurasi daripada jika sebaliknya.

3. Mempekerjakan staff DBA

   Untuk dapat memanfaatkan kemampuan DBMS secara penuh, diperlukan pengetahuan khusus. Pengetahuan khusus ini umumnya didapatkan dari para pengelolah database (DBA).

2.1.6 Mekanisme *View*

Menurut Connolly & Begg (2002, p17), mekanisme view memungkinkan masing-masing user untuk memiliki perspektif sendiri atas suatu database. View menampilkan data yang relevan bagi user tertentu dan menyembunyikan data lainnya yang tidak relevan.

Ada beberapa keuntungan penggunaan view menurut Connolly & Begg (2002, p17- p18) :

1. View memberikan tingkat pengamanan yang lebih

   View dapat diatur untuk menyembunyikan data-data yang tidak relevan bagi user. Misalnya bagi user supervisor, dapat dibuatkan view untuk menampilkan data karyawan dengan mengecualikan data gaji karyawan. Namun bagi user lainnya misalnya kepala HRD, view tersebut dapat dikustomisasi menjadi juga menampilkan data gaji karyawan.

2. View menyediakan mekanisme bagi user untuk mengubah penampilan database secara khusus bagi mereka. Misalnya staff pembelian dapat menampilkan nama field “jumlah” yang sedikit ambigu menjadi “Jumlah Pembelian”.

2.1.7 *Entity Relationship Modelling*

Conceptual modelling merupakan tahap yang penting dalam perancangan aplikasi database yang sukses. (Elmasri, 2000, p41). Dalam pendekatan tradisional yang memusatkan pada struktur dan constraints database selama perancangan database, *Entity Relationship Modelling* berperan penting. Model ER merupakan model data konseptual tingkat tinggi yang populer. Model ini beserta variasi-variasinya sering digunakan untuk perancangan konseptual aplikasi database.

2.1.7.1 Entity

Entity, menurut Elmasri (2000, p45) adalah obyek utama yang direpresentasikan oleh model ER. Di dunia nyata, sebuah entity mungkin merupakan obyek yang memiliki wujud fisik.

2.1.7.2 *Entity Type and Entity Set*

Menurut Elmasri (2000, p49), *entity type* memiliki arti satu set entity yang memiliki atribut yang sama. Sedangkan entity set adalah kumpulan seluruh entity dari entity type tertentu.

2.1.7.3 *Attributes*

Setiap entity memiliki atribut-atribut, yaitu sesuatu, hal, yang terkait yang dapat digunakan untuk menjelaskan suatu entity (Elmasri, 2000, p45). Menurut Connolly dan Begg (2002,p338-p342), atribut

adalah sifat dari sebuah entitas atau tipe *relationship*. Sifat tertentu dari entitas disebut sebagai atribut. Atribut menyimpan nilai dari setiap *entity occurrence* dan disimpan di dalam basis data.

Atributte domain adalah sejumlah nilai yang diperkenankan untuk satu atau lebih atribut. Setiap atribut yang dihubungkan dengan sejumlah nilai disebut domain. Domain menetapkan nilai potensial yang sebuah atribut bisa simpan adalah sama dengan konsep domain pada model relasional.

Simple atributte adalah sebuah susunan atribut dari komponen tunggal (*single component*) dengan keberadaan yang bebas (*independent existence). Simple atributte* tidak bisa dibagi lagi kedalam komponen yang lebih kecil. Contohnya posisi dan gaji dari entitas pegawai. Sedangkan *composed atributte* adalah sebuah susunan atribut dari banyak komponen dengan sebuah keberadaan yang bebas dari masing-masingnya. Contohnya atribut alamat dari entitas kantor cabang yang mengandung nilai (jalan, kota, kodepos) bisa dipecahkan menjadi *simple atributte* jalan, kota dan kodepos .

Single value atributte adalah atribut yang hanya menyimpan nilai tunggal untuk suatu sifat dari entitas. *Multi-valued atributte* adalah atribut yang bisa menyimpan nilai lebih dari satu untuk suatu sifat dari entitas. Contohnya atribut telepon pada entitas kantor cabang yang bisa memiliki lebih dari satu nomor telepon.

Derived atributte (atribut turunan) adalah atribut yang menunjukan nilai yang diperoleh dari atribut yang berhubungan atau

kumpulan atribut yang berhubungan, tidak terlalu dibutuhkan dalam tipe entitas yang sama. Atribut turunan mungkin juga menyangkut hubungan dari atribut pada tipe entitas yang berbeda.

2.1.7.4 *Relationship Type*

Pengertian *relationship type* menurut Connolly dan Begg (2002,p334), "*relationship type is a set of association between one or more participating entity types"*, yang dapat diartikan, "tipe relasi adalah sekumpulan hubungan antara satu atau lebih tipe-tipe entitas".

Derajat dari r*elationship* adalah jumlah dari partisipasi tipe entitas dalam sebuah tipe *relationship* tertentu. Entitas yang berkaitan dalam sebuah tipe *relationship* dikenal sebagai *participant* dalam *relationship* dan jumlah *participant* dalam *relationship* disebut sebagai derajat (*degree)* dari *relationship.* Oleh karena itu, derajat dari sebuah *relationship* menunjukkan jumlah dari entitas yang terkait dalam *relationship.* Sebuah *relationship* berderajat 2 disebut *binary*, *relationship* berderajat 3 disebut *ternary*, dan *relationship* berderajat 4 disebut *quartenary*.

2.1.7.5 Key Attributes

Menurut McLeod, R (2001, p178) *key field* adalah atribut yang memiliki nilai yang secara unik mengidentifikasi setiap record dalam tabel. Definisi *key attributes* menurut Elmasri (2000, p50) adalah atribut-atribut yang memiliki nilai yang unik di seluruh entity tunggal pada keseluruhan sistem. Nilai key attributes ini dapat digunakan untuk mengidentifikasi masing-masing entity secara unik.

Menurut Connolly & Begg (2002, p340), *candidate key* adalah kumpulan atribut yang secara unik mengidentifikasi setiap *entity type*. *Primary key* adalah *candidate key* yang telah dipilih untuk mengidentifikasi tiap baris secara unik. *Primary key* harus merupakan field yang benar-benar unik dan tidak boleh ada nilai null. *Alternate key* adalah *candidate key* yang tidak terpilih sebagai *primary key*. *Composite key* : pada kondisi tertentu, suatu atribut tidak dapat digunakan untuk mengidentifikasi suatu baris secara unik dan membutuhkan kolom yang lain untuk digunakan sebagai *primary key*. *Foreign key* berfungsi sebagai penghubung antar tabel apabila sebuah *primary key* terhubung ke tabel lain.

2.1.8 *Data Definition Language* (DDL)

Menurut Connolly & Begg (2002, p40), DDL adalah bahasa yang memungkinkan *Database Administrator* untuk mendeskripsikan dan mendefinisikan entity-entity, atribut-atribut, dan hubungan yang diperlukan oleh aplikasi, beserta dengan integrity dan constraint yang terkait. DDL digunakan untuk mendefinisikan basis data, tabel, dan *view*.

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]

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

2. *Alter table*

   Pernyataan *alter table* dipakai untuk menambah atau membuang kolom dan konstrain.

   Bentuk umum:

   ALTER TABLE table_name

   [ADD column_name DataType [NULL | NOT NULL]]

   [DROP colum_name DataType [RESTRICT |CASCADE]]

   [ADD Constraint_name]

   [DROP Constraint_name [RESTRICT |CASCADE]]

3. *Drop table*

   Pernyataan *drop table* digunakan untuk membuang atau menghapus tabel beserta semua data yang terkait didalamnya.

   Bentuk umum:

   DROP TABLE table_name;

4. *Create index*

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

   Bentuk umum:

   CREATE [UNIQUE] INDEX index_name

   ON table_name

   [colum_name [,column_name]...)

5. *Drop index*

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

DROP INDEX index_name

2.1.9 *Data Manipulation Language* (DML)

Menurut Connolly & Begg (2002, p40), DML adalah. suatu bahasa yang menyediakan satu set operasi untuk mendukung operasi manipulasi data yang dasar terhadap data yang disimpan di database. DML dipakai untuk menampilkan, menambah, mengubah dan menghapus data di dalam objek-objek yang didefinisikan oleh DDL.

1. *Select*

   pernyataan *select* digunakan untuk menampilkan sebagian atau seluruh isi dari tabel dan menampilkan kombinasi isi dari beberapa tabel.

   Bentuk umum:

   SELECT fields

   FROM table_name

   WHERE condition

2. *Update*

   Pernyataan *update* digunakan untuk mengubah isi satu atau beberapa atribut dari suatu tabel

   Bentuk umum:

   UPDATE table_name

   SET column1 = value1, column2 = value2,...

   WHERE condition

3. *Insert*

   Pernyataan *insert* digunakan untuk menambahkan satu atau beberapa baris nilai baru ke dalam suatu tabel.

   Bentuk umum:

   INSERT table_name (column_list) VALUES (value list)

4. *Delete*

   Pernyataan *delete* digunakan untuk menghapus sebagian atau seluruh isi dari tabel

   Bentuk umum:

   DELETE FROM table_name

   WHERE condition

2.1.10 Kamus Data dan Normalisasi

2.1.10.1 Kamus Data

Menurut Mcleod (2001, p396), kamus data adalah suatu penjelasan tertulis mengenai data yang berada di dalam basis data. Kamus data ini dimaksudkan untuk melengkapi pembuatan model proses yang menggunakan diagram alir data. Pada kamus data berbasis komputer, deskripsi data dimasukkan ke dalam komputer dengan DDL, sistem kamus data atau *CASE tool*.

2.1.10.2 Normalisasi

Pengertian normalisasi Connolly dan Begg (2002, p376) "*Normalisation is a technique for producing a set of relation with desirable properties, given the data requirements of an enterprise"* yang dapat diartikan "Normalisasi adalah sebuah teknik untuk menghasilkan

relasi dengan properti – properti yang diinginkan, memberikan kebutuhan data dari sebuah perusahaan”

Menurut Petroutsos (2002,p71) , ada beberapa aturan dalam perancangan basis data, yang disebut dengan aturan normalisasi. Aturan-aturan ini akan merancang basis data yang normal, atau setidaknya memverifikasi rancangan.

Basis data dianggap normal jika basis data tersebut tidak mengulangi data (*data redudancy*) atau tidak menimbulkan keanehan pada proses *update,delete,* atau *insert*.

Proses pembentukan tabel normal (normalisasi) bertujuan untuk :

1. Membuat sekecil mungkin terjadinya data rangkap.
2. Menghindarkan adanya data yang tidak konsisten terutama bila dilakukan penghapusan atau penambahan data sebagai akibat adanya data rangkap.
3. Menjamin bahwa identitas tabel secara tunggal sebagai determinan semua atribut.

Proses normalisasi tabel secara detil dibagi menjadi lima tahap sehingga dikenal bentuk – bentuk tabel normal sesuai dengan tahapan normalisasi yang telah dilakukan yaitu bentuk normal pertama, kedua, ketiga, *Boyce-Codd*, keempat dan kelima.

1. Bentuk Normal Pertama (*First Normal Form/1NF)*

   Aturan bentuk normal pertama (1NF) menurut Connolly dan Begg (2002,p388), "*A relation in which the intersection of each row and column contains one and only one value"*, yang dapat diartikan, "sebuah relasi dimana tiap baris dan kolom hanya berisi satu nilai".

   Bentuk normal pertama dicapai bila tiap nilai atribut adalah tunggal. Kondisi ini dapat diperoleh dengan melakukan eliminasi terjadinya data ganda (*repeating groups*). Pada kondisi normal pertama ini kemungkinan masih terjadi adanya data rangkap

2. Bentuk Normal Kedua (*Second Normal Form/2NF)*

   Aturan bentuk normal kedua (2NF) menurut Connolly dan Begg (2002,p392), "*A relation that is in first normal form and every non-primary key atributte is fully function dependent on the primary key"*, yang dapat diartikan, "sebuah relasi dalam bentuk normal pertama dan setiap atribut bukan *primary key* bergantung secara fungsional terhadap *primary key*".

   Bentuk normal kedua adalah berdasarkan konsep ketergantungan fungsional penuh (*full functional dependency). Full functional dependency* dinyatakan dengan jika A dan B adalah atribut dari suatu relasi (*relation)*, B adalah fungsional ketergantungan penuh (*fully functional depedency)* pada A jika B adalah secara fungsional bergantung pada A, tetapi bukan merupakan himpunan bagian dari A. Bentuk normal kedua menciptakan sebuah relasi pada bentuk normal

pertama dan semua atribut yang bukan *primary key* adalah fungsional tergantung penuh terhadap *primary key*.

3. Bentuk Normal Ketiga *(Third Normal Form/3NF)*

Aturan bentuk normal ketiga (3NF) menurut Connolly dan Begg (2002,p394), "*A relation that is in first and second normal form, and in which none primary key atributte is transitively dependent on the primary key"*, yang dapat diartikan, "sebuah relasi dalam bentuk normal pertama dan kedua dan semua atribut bukan *primary key* yang bergantung secara transitif kepada *primary key*".

Bentuk normal ketiga berdasarkan pada konsep peralihan ketergantungan (*transitive dependency*). *Transitive dependency* adalah sebuah kondisi dimana A,B dan C adalah atribut dari sebuah relasi bahwa jika A -> B dan B-> C, maka C adalah *transitive dependency* terhadap A melewati B (menyatakan bahwa A bukan *functional dependent* pada B atau C). Pada bentuk normal ketiga, sebuah relasi pada bentuk normal pertama dan kedua, dimana tidak ada atribut *non-primary key* bergantung secara transitif (*transitively dependency)* pada *primary key.*

4. Bentuk Normal *Boyce-Codd(Boyce-Codd Normal Form/BCNF)*

Aturan bentuk normal *Boyce-Codd* (BCNF) menurut Connolly dan Begg (2002,p398), "*A relation is in BCNF, if and only if, every determinant is a candidate key"*, yang dapat diartikan, "sebuah relasi disebut BCNF, jika dan hanya jika setiap determinannya adalah sebuah *candidate key*".

Untuk menguji apakah suatu relasi sudah BCNF, dilakukan identifikasi semua determinan dan memastikan bahwa determinan tersebut adalah *candidate key*. Determinan adalah sebuah atribut, atau kumpulan atribut, dimana beberapa atribut yang lain masih bergantung secara fungsional penuh (*fully functionaly dependent).*

Perbedaan antara 3NF dan BCNF dalam hal *functional dependency*. A->B, 3NF mengijinkan ketergantungan ini dalam sebuah relasi jika B adalah atribut *primary key* dan A bukan *candidate key*. Sedangkan dalam BCNF ketergantungan ini tetap ada dalam sebuah relasi, dimana A harus sebuah *candidate key.*

5. Bentuk Normal Keempat (*Fourth Normal Form/4NF)*

Aturan bentuk normal keempat (4NF) menurut Connolly dan Begg (2002,p407-408), *"A relation that is in Boyce-Codd Normal Form and contains no nontrivial multi-valued dependencies"*, yang dapat diartikan, "sebuah relasi dalam Boyce-Codd normal form (BCNF) dan tidak mengandung ketergantungan *multi value* nontrivial (*nontrivial multi-valued dependencies)*

Bentuk normal keempat (4NF) merupakan bentuk yang lebih kuat dari BCNF dimana 4NF mencegah relasi dari *nontrivial multi-valued dependency* dan *data redudancy*. Normalisasi dari BCNF ke 4NF meliputi pemindahan *multi-valued dependency* dari relasi dengan menempatkan atribut dalam sebuah relasi baru bersama dengan determinan.

Multi-valued dependency (MVD) menggambarkan ketergantungan antara atribut-atribut dalam suatu relasi.

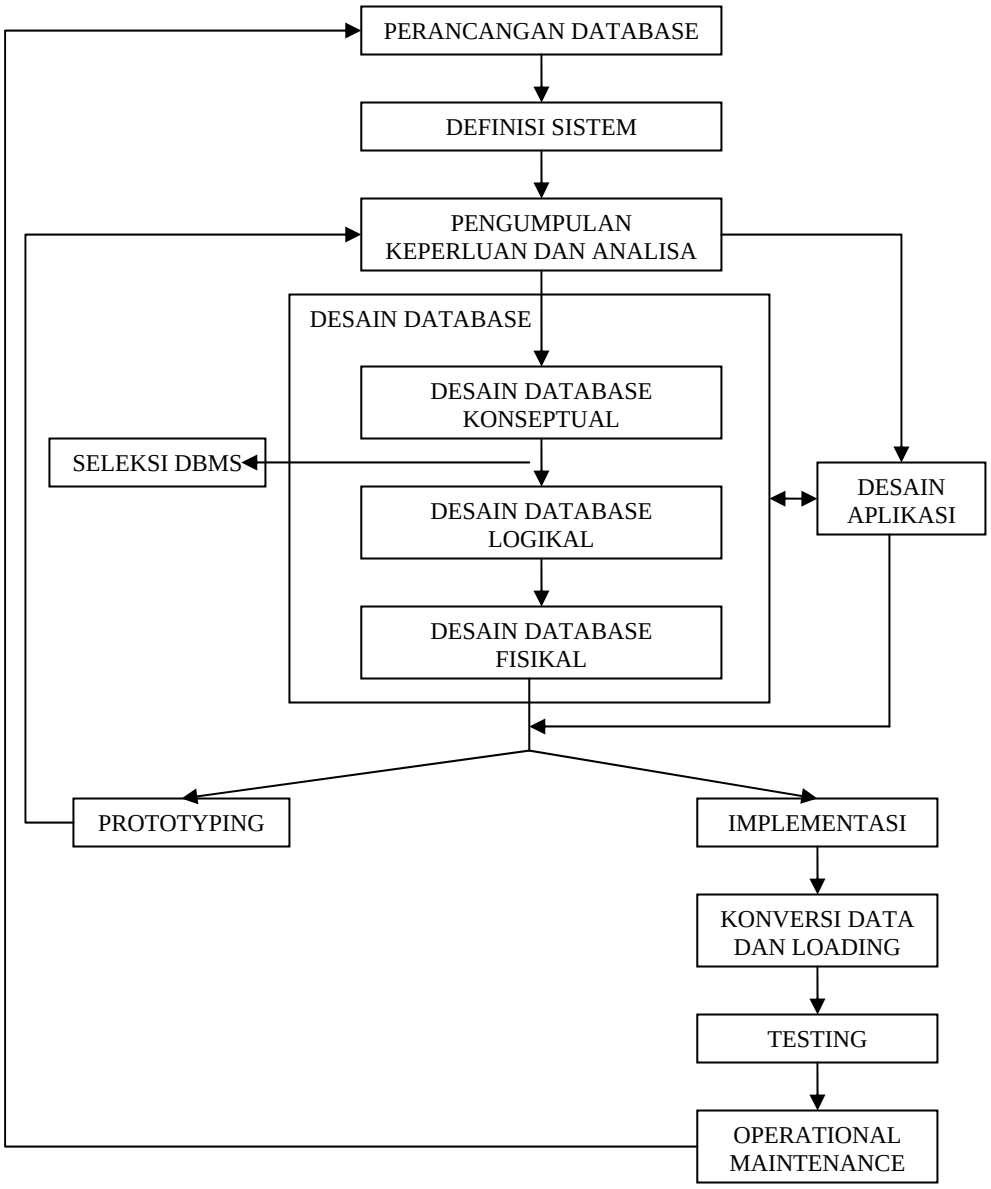
6. Bentuk Normal Kelima (*Fifth Normal Form/5NF)*

Aturan bentuk normal kelima (5NF) menurut Connolly dan Begg (2002,p410), *"A relation that has no join dependency"*, yang dapat diartikan, "sebuah relasi yang tidak mempunyai kertergantungan gabungan(*join dependency).*

Join dependency menggambarkan sebuah tipe ketergantungan. Sebagai contoh, untuk sebuah relasi R dengan subset – subset atribut dari R yang dimisalkan dengan A,B,....,Z, sebuah relasi R menunjukan *join dependency*, jika dan hanya jika, setiap nilai dari R sama dengan gabungan dari proyeksi-proyeksinya pada A,B,...,Z.

2.1.11 *Database Application Lifecycle*

Tahapan penerapan *lifecycle* dalam metodologi perancangan basis data menurut Connolly dan Begg (2002,p270), *Database Systems A pratical* sebagai berikut :



Gambar 2.1. Tingkatan dari Aplikasi database Lifecycle

(Connolly and Begg p272)

2.1.11.1 Perencanaan Basis Data (*Database Planning*)

Menurut Connolly dan Begg(2002,p273-274), perencanaan basis data (*database planning)* merupakan aktifitas manajemen yang mengizinkan tingkatan dari aplikasi basis data untuk direalisasikan seefisien dan seefektif mungkin. *Database planning* harus diintergrasikan dengan keseluruhan strategi sistem informasi dari perusahaan. Ada tiga hal penting dalam menyusun sebuah strategi sistem informasi, yaitu :

1. Identifikasi dari tujuan dan rencana perusahaan dengan penentuan kebutuhan sistem informasi berikutnya.
2. Evaluasi dari sistem informasi saat ini untuk menentukan kelebihan dan kelemahan yang ada saat ini.
3. Penilaian dari kesempataan – kesempataan TI yang mungkin menghasilkan keuntungan kompetitif.

Langkah penting dari tahap ini adalah mendefinisikan secara jelas tentang pernyataan misi untuk proyek basis data. Pernyataan tersebut mendefinisikan tujuan utama dari aplikasi *data base*. Bila pernyataan tersebut selesai maka langkah selanjutnya adalah mengidentifikasikan sasarannya. Pernyataan dan sasaran ini perlu didukung oleh informasi-informasi tambahan yang menentukan pekerjaan apa saja yang harus diselesaikan, sumber – sumber yang mendukung dan biaya yang harus dikeluarkan.

2.1.11.2 Definisi Sistem (*System Definition*)

Menurut Connolly dan Begg(2002,p274), definisi sistem (*system definition)* adalah proses yang memaparkan jangkauan dan batasan dari aplikasi basis data dan pandangan – pandangan utama para pemakai. Sebelum mendesain suatu aplikasi basis data penting untuk terlebih dahulu mengidentifikasikan batasan – batasan dari sistem yang sedang diteliti dan bagaimana kaitannya dengan bagian lain sistem informasi perusahaan. Perlu dipirkan untuk kebutuhan yang akan datang selain dari keadaan saat ini. Dan tak lupa untuk mengidentifikasi pandangan pemakai yang merupakan aspek penting dari pengembangan aplikasi basis data karena membantu untuk memastikan bahwa tidak ada pemakai utama basis data yang terlupa ketika pengembangan aplikasi baru tersebut

2.1.11.3 Analisa dan Pengumpulan Kebutuhan (*Requirement Collection and Analysis*)

Menurut Connolly dan Begg(2002,p276), analisis dan pengumpulan kebutuhan(*requirement collection and analysis)* adalah sebuah proses pengumpulan dan analisis informasi tentang bagian dari perusahaan yang akan didukung oleh aplikasi basis data, dan menggunakan informasi ini untuk mengidentifikasi kebutuhan pemakai terhadap sistem baru.

Informasi yang dikumpulkan termasuk :

1. Penjabaran dari data yang digunakan,
2. Detail mengenai bagaimana data digunakan,
3. Kebutuhan tambahan apapun untuk aplikasi basis data yang baru.

Informasi ini kemudian dianalisis untuk mengidentifikasikan kebutuhan yang dimasukan untuk aplikasi basis data yang baru tersebut. Ada tiga macam pendekatan untuk mengatur kebutuhan dari sebuah aplikasi basis data dengan berbagai pandangan pemakai, yaitu:

1. Pendekatan *Centralized*

   Kebutuhan untuk tiap pandangan pemakai disatukan menjadi satu set kebutuhan untuk aplikasi basis data. Umunya pendekatan ini dipakai jika basis datanya tidak terlalu kompleks.

2. Pendekatan *View Integration*

   Kebutuhan untuk tiap pandangan pemakai digunakan untuk membangun sebuah model yang terpisah yang mempresentasikan tiap pandangan pemakai tersebut. Hasil dari data-data model tersebut kemudian di satukan dibagian desain basis data

3. Kombinasi keduanya

2.1.11.4 Desain Basis Data (*Database Design*)

Menurut Connolly dan Begg (2002,p279), perancangan basis data (*database design)* merupakan proses pembuatan suatu desain untuk sebuah basis data yang akan mendukung oprasional dan sasaran suatu perusahaan.

Ada dua pendekatan untuk merancang sebuah basis data, yaitu :

1. Pendekatan *bottom-up*

   Yang dimulai pada tingkat awal dari atribut (yaitu, properti dari entity dan *relationship*), yang mana melalui analisis dari asosiasi antar atribut, dikelompokan menjadi hubungan yang merepresentasikan jenis-

jenis entitas dan hubungan antar entitas. Pendekatan ini cocok untuk mendesain basis data yang simpel dengan jumlah atribut yang tidak banyak.

2. Pendekatan *top-down*

Metode perancangan top-down adalah metode perancangan basis data dimana perancangan dimulai dari pengembangan model data yang awalnya mengandung beberapa entity tingkat atas beserta relasi-relasi antara entity tersebut. Kemudian dilakukan penyesuaian bertahap untuk mengidentifikasi entity dengan tingkat lebih rendah, hubungan antara mereka, dan atribut-atribut yang dimilikinya. Pendekatan ini biasanya digambarkan melalui ER (*entity Relationship)*

Pada tahap ini ada bagian yang disebut *data modelling* yang digunakan untuk membantu pemahaman dari data dan untuk memudahkan komunikasi tentang kebutuhan informasi. Dengan dibuatnya model dta dapat membantu memahami :

1. Pandangan tiap pemakai mengenai data
2. Kealamian data itu sendiri, kebebasan representasi fisiknya
3. Kegunaan dari data berdasarkan pandangan pemakai.

Kriteria untuk model data, yaitu :

1. *Structural validity*

   Konsistensi dengan cara yang didefinisikan perusahaan dan menyusun informasi

2. *Simplicity*

   Kemudahan untuk pemahaman baik bagi yang profesional di bidang sistem informasi maupun pemakai yang nonteknis.

3. *Expressibility*

   Kemampuan untuk membedakan antara data yang berbeda dan hubungan antara data.

4. *Nonredudancy*

   Pembuangan informasi yang tak ada hubungannya; khususnya representasi dari tiap potongan informasi tepatnya hanya sekali.

5. *Shareability*

   Tidak spesifik untuk aplikasi dan teknologi khusus apapun dan dengan demikian dapat digunakan oleh banyak orang.

6. *Extensibility*

   Kemampuan mengembangkan untuk mendukung kebutuhan baru dengan efek yang minimal bagi pemakai yang ada.

7. *Integrity*

   Konsistensi tehadap cara yang digunakan perusahaan dan mengatur informasi.

8. *Diagramic representation*

   Kemampuan untuk merepresentasikan sebuah model menggunakan notasi diagram yang dapat dipahami dengan mudah.

Menurut Connolly dan Begg (2002,p419-437), *Database Design* dibagi dalam tiga tahapan yaitu *conseptual database design, logical database design,* dan *physical database design.*

2.1.11.5 Seleksi DBMS

Menurut Connolly dan Begg (2002,p284), pemilihan DBMS yang sesuai untuk mendukung aplikasi basis data. Yang mencakup :

1. Mendefinisikan syarat-syarat referensi studi

   Menentukan sasaran,batasan masalah, dan tugas yang harus dilakukan

2. Mendaftar 2 atau 3 jenis barang

   Membuat daftar barang – barang, misalkan dari mana barang ini didapat, berapa biayanya serta bagaimana bila ingin mendapatkannya.

3. Mengevaluasi barang

   Barang-barang yang ada dalam daftar diteliti lebih lanjut untuk mengetahui kelebihan dan kekurangan barang tersebut

4. Merekomendasikan pilihan dan membuat laporan

   Merupakan langkah terakhir dari seleksi DBMS yaitu mendokumentasikan proses dan untuk menyediakan pernyataan mengenai kesimpulan dan rekomendasi terhadap salah satu produk DBMS.

2.1.11.6 Desain Aplikasi (*Application Design*)

Menurut Connolly dan Begg (2002,p287-288), perancangan aplikasi (*application design)* adalah merancang antarmuka pemakai (*user interface)* dan program aplikasi yang akan memproses basis data. Ditinjau dari gambar 2.1 bahwa, perancangan basis data dan perancangan aplikasi adalah aktivitas bersamaan pada *database*

application lifecycle. Dalam kasus sebenarnya, adalah tidak mungkin untuk menyelesaikan perancangan aplikasi sebelum perancangan basis data selesai.

Dalam perancangan aplikasi harus memastikan semua pernyataan fungsional dari spesifikasi kebutuhan pemakai (*user requirement specification*) yang menyangkut perancangan aplikasi program yang mengakses basis data dan merancang transaksi yaitu cara akses ke basis data dan perubahan terhadap isi basis data (*retireve,update* dan kegiatan keduanya). Artinya bagaimana fungsi yang dibutuhkan bisa terpenuhi dan merancang antarmuka pemakai (*user interface*) yang tepat. Antarmuka yang dirancang harus memberikan informasi yang dibutuhkan dengan cara menciptakan '*user-friendly'*. Kebanyakan antarmuka pemakai yang diabaikan akan niscaya membuat masalah. Bagimanapun, antarmuka pemakai harus diakui sebagai komponen dari sistem yang penting, dimana agar mudah dipelajari dan mudah digunakan, sehingga pemakai akan cenderung memberdayagunakan informasi yang disajikan.

2.1.11.7 *Prototyping*

Menurut Connolly dan Begg (2002,pp291-292), *prototyping* adalah membuat model kerja dari aplikasi basis data, yang membolehkan perancang atau *user* untuk mengevaluasi hasil akhir sistem, baik dari segi tampilan maupun fungsi yang dimiliki sistem. Tujuan dari pengembangan *prototype* aplikasi basis data adalah untuk memungkinkan pemakai menggunakan *prototype* untuk

mengidentifikasi keistimewaan sistem atau kekurangannya, dan memungkinkan perancang untuk memperbaiki atau melengkapi keistimewaan (*feature*) dari aplikasi basis data baru.

Ada dua strategi *prototyping* yang umun digunakan sekarang, yaitu, *requirement prototyping* dan *evolutionary prototyping. Requirement prototyping* adalah menggunakan *prototype* untuk menetapkan kebutuhan dari tujuan aplikasi basis data dan ketika kebutuhan sudah terpenuhi, *prototype* tidak digunakan lagi atau dibuang (*discard)*. Sedangkan *evolutionary prototype* menggunakan tujuan yang sama, tetapi perbedaan pentingnya adalah *prototype* tetap digunakan untuk selanjutnya dikembangkan menjadi aplikasi basis data yang bekerja.

2.1.11.8 Implementasi (*Implementation*)

Menurut Connolly dan Begg (2002,p292), implementasi (*implementation)* adalah membuat definisi basis data secara eksternal, konseptual, dan internal dan program aplikasi. Implementasi merupakan realisasi dari basis data dan perancangan aplikasi. Implementasi basis data dicapai menggunakan *Data Definition Language* (DDL) dari DBMS yang dipilih atau *graphical user interface GUI).* Pernyataan DDL digunakan untuk membuat struktur basis data dan file basis data kosong. Pandangan pemakai (*user view*) lainnya juga diimplementasikan dalam tahapan ini. Bagian dari aplikasi program adalah transaksi basis data yang diimplementasikan dengan *Data Manipulation Language (DML)* dari sasaran DBMS, mungkin termasuk

host programming language seperti, Visual Basic, Delphi, C,C++,Java,COBOL,Ada atau Pascal.

2.1.11.9 *Data Conversion and Loading*

Menurut Connolly dan Begg (2002,pp292-293), *data convertion and* loading adalah mencakup pengambil data dari sistem yang lama untuk dipindahkan ke dalam sistem yang baru. Tahapan ini dibutuhkan ketika sistem basis data yang baru menggantikan sistem yang lama. Pada masa sekaran umumnya DBMS memiliki kegunaan (*utility)* untuk memasukan file ke dalam basis data baru. Biasanya menggunakan spesifikasi dari sumber file dan sasaran basis datanya. Kegunaan ini memungkinkan pengembang (*developer)* untuk mengkonversi dan menggunakan aplikasi program lama untuk digunakan oleh sistem baru. Ketika *conversion* dan *loading* dibutuhkan, prosesnya harus direncanakan untuk memastikan kelancaran transaksi untuk keseluruhan operasi.

2.1.11.10*Testing*

Menurut Connolly dan Begg (2002,p293), *testing* adalah proses menjalankan program aplikasi untuk menemukan keasalahan – kesalahan. Sebelum digunakan, aplikasi basis data yang baru dikembangkan harus diuji secara menyeluruh. Untuk mencapainya harus hati – hati dalam menggunakan perencanaan strategi uji dan menggunakan data asli untuk semua proses pengujian. Di dalam definisi *testing* ini tidak menggunakan pandangan yang biasa, *testing* adalah proses demonstrasi tanpa kesalahan. Dalam kenyataan *testing* tidak

luput dari kesalahan. Jika *testing* menunjukkan keberhasilan, maka pengujian akan menemukan kesalahan pada program aplikasi dan kemungkinan struktur basis datanya.

Di dalam merancang basis data, *users* dari sistem baru seharusnya terlibat di dalam proses *testing*. Situasi yang ideal untuk melakukan uji sistem adalah menguji basis data pada perangkat keras yang berbeda, tetapi hal ini sering tidak dilakukan. Jika data yang asli digunakan perlu *backups* untuk mengantisipasi kesalahan, atau *error*. Setelah *testing* selesai, sistem aplikasi siap digunakan, dan diserahkan kepada *users*.

2.1.11.11*Operational Maintenance*

Menurut Connolly dan Begg (2002,pp293-294), *operational maintenance* adalah proses memantau dan memelihara sistem setelah diinstal. Pada tahapan sebelumnya, basis data benar – benar diuji dan diimplementasikan. Sekarang sistem beralih ketahapan pemeliharaan. Yang termasuk aktivitas dari tahapan pemeliharaan adalah sebagai berikut

1. Memantau kinerja dari sistem. Jika kinerjanya menurun dibawah level yang dapat diterima, mungkin basis data perlu direorganisasi.
2. Pemeliharaan dan *upgrade* aplikasi basis data-nya (jika dibutuhkan)

Ketika basis data sepenuhnya bekerja, pemantauan harus memastikan kinerjanya dapat berada dalam tingkat yang dapat diterima. Sebuah DBMS biasanya menyediakan berbagai kegunaan (*utillities*) untuk membantu administrasi basis data termasu kegunaan (*utilities*) untuk mengisi data ke dalam basis data dan untuk memantau sistem. Kegunaan ini

memperbolehkan sistem pemantauan untuk memberikan informasi seperti tentang pemakaian basis data, dan strategi eksekusi *query*. *Database administrator* dapat menggunakan informasi ini untuk memperbaiki sistem agar dapat memberikan kinerja yang lebih baik.

2.1.12 Desain Konseptual, Logikal, dan Fisik Basis Data

2.1.12.1 Desain Konseptual Basis Data

Merupakan proses pembuatan model basis data dengan menggunakan informasi dan data yang diperoleh dari perusahaan, bebas dari segala pertimbangan fisikal. Perancangan basis data konseptual seluruhnya independen dan implementasi seperti target DBMS *software*, program aplikasi, bahasa pemograman, atau pertimbangan-pertimbangan fisikal lainnya.

Jelasnya perancangan basis data konseptual merupakan tahapan pertama dari tahapan perancangan basis data dan menciptakan model data konseptual (*conseptual data model*) dari bagian perusahaan yang akan dibuat basis datanya. Model data dibuat dengan menggunakan suatu dokumentasi informasi yaitu spesifikasi kebutuhan yang dimiliki oleh *user*.

Membangun model data lokal konseptual, tujuannya untuk membangun suatu model data konseptual lokal dari suatu perusahaan.

Langkah-langkah untuk membuat model data konseptual dapat dijabarkan sebagai berikut :

1. Mengidentifikasi *entity*

   Untuk menentukan entitas utama yang dibutuhkan oleh masing-masing *view* dalam sistem. Misalkan :

Staff, yang menggambarkan seluruh tingkatan staff yang ada.

PropertyForRent, menggambarkan semua properti yang disewakan

2. *Identify relationship types*

Untuk menentukan hubungan – hubungan penting yang ada antara jenis – jenis entitas yang telah diidentifikasi. Misalkan :

- *Staff Manages PropertyForRent*, yaitu staf mengatur entitas properti
- *PrivateOwner Owns PropertyForRent,* yaitu entitas *PropertyOwner* memiliki yang ada pada entitas *PropertyForRent.*
- *PropertyForRent AssiciatedWith Lease,* yaitu entitas *PropertyForRent* saling bekerja sama dengan entitas *Lease*.

Biasanya dilanjutkan dengan membuat diagram hubungan tersebut yang disebut ER diagramserta menentukan hubungan kemajemukannya.

3. *Identify and associate attributes with entity or relationship types*

Untuk menentukan atribut yang berkaitan dengan entitas yang telah ditentukan. Misalkan untuk entitas *staff* ditentukan atribut seperti *staffNo* (yang mengandung nomor – nomor kode setiap staf), *name, position, sex*. Begitu pun untuk setiap entitas lainnya.

4. *Determine attribute domains*

Untuk menentukan domain untuk tiap – tiap atribut yang ada. Suatu domain adalah suatu kelompok nilai yang dari mana satu atau lebih atribut mengambil nilainya. Misalkan :

- Domain atribut untuk nilai *staffNo* yang valid misalnya panjang maksimalnya 5 karakter dengan 2 karakter pertama harus huruf dan yang 3 berikutnya berupa angka yang berkisar antara 1 – 999.
- Nilai yang mungkin untuk atribut *sex* dari entiti *Staff* misalnya huruf M atau F saja.

5. *Determine candidate and primary key attributes*

   Untuk mengidentifikasikan *candidate key* dan *primary key* dari kumpulan atribut pada tiap – tiap entitas, misalkan pada entitas *Staff primary key*-nya adalah *staffNo* yang mewakili atribut lainnya, sehingga pada saat kita mengakses suatu basis data hanya dengan memasukkan nilai *staffNo* kita dapat mengetahui nilai – nilai atribut lainnya yang ada dalam entitas *Staff*.

6. *Consider use of enchanced modeling concepts (optional)*

   Memikirkan kegunaan dari model konsep yang telah dikembangkan. Tahap ini tidak perlu pembahasan lebih lanjut dikarenakan hanya merupakan langkah tambahan saja, boleh dilakukan boleh juga tidak.

7. *Check model for redudamcy*

   Untuk memriksa kelebihan entitas maupun atribut yang ada pada model tersebut. Pertama yang dilakukan adalah memeriksa kembali hubungan yang ada, apabila terdapat suatu hubungan yang mirip misalnya entitas klien dengan entitas penyewa yang memiliki hubungan dengan ke satu entitas tapi dengan jenis hubungan yang sama yaitu menyewa. Langkah kedua yaitu menggabungkan kedua entitas yang dianggap memiliki kesamaan dan bisa digabung.

8. *Validate local conceptual model against user transaction*

   Untuk memastikan bahwa model konsep tersebut mendukung proses transaksi yang dibutuhkan. Misalkan menggambarkan transaksi dengan memeriksa semua informasi yang ada. Contohnya dengan adanya hubungan *Staff manages PropertyForRent* kita dapat mengetahui detil dari properti dan staf yang menangani properti tersebut.

9. *Revalidate local conceptual data model with user*

   Untuk mengkaji ulang model konsep yang telah kita buat dengan pemakai sehingga para pemakainya dapat memahami maksud basis data yang telah dibuat.

2.1.12.2 Desain Logikal Basis Data

Merupakan suatu proses pembuatan model dengan menggunakan informasi yang diperoleh dari perusahaan serta berdasarkan pada model adta spesifik, tetapi bebas dari *particular DBMS* dan *physical consideration* lainnya. Model data konseptual yang dibangun pada fase sebelumnya diperhalus dan dipetakan pada model data logikal. Model data logikal didasarkan pada target model data untuk basis data (sebagai contoh : model data relasional). Model data logikal merupakan sumber informasi untuk fase selanjutnya, yang dinamakan *physical database design*. Aktivitas pada *logical database design* adalah terdiri dari dua langkah besar, dimana langkah pertama adalah membangun sebuah model data logikal lokal dari model data konseptual lokal yang menggambarkan pandangan (*view*) tertentu dari perusahaan dan kemudian mengesahkan model ini untuk memastikan strukturnya telah benar atau menggunakan teknik normalisasi.

Sedangkan langkah kedua atau langkah selanjutnya adalah untuk mengkombinasikan model data logikal lokal individual ke dalam sebuah model data logikal global tunggal yang menggambarkan perusahaan.

Hasil akhir tahapan ini berupa sebuah kamus data yang berisi semua atribut beserta *key*-nya (*primary key*, *alternate key*, dan *foreign key*) dan ERD keseluruhan (relasi global) dengan semua atribut *key*-nya.

2.1.12.3 Desain Fisik Basis Data

Merupakan suatu proses pembuatan deskripsi dari suatu implementasi basis data pada *secondary storage*; hal ini mendeskripsikan *base relation*, organisasi file, dan indeks yang digunakan untuk mencapai efisiensi akses kedalam data, dan *associated integrity constraints* yang lainnya dan *security measures*. *Physical database design* merupakan fase ketiga dan terakhir dari proses desain basis data. Dimana desainer memutuskan bagaimana basis data tersebut diimplementasikan. Secara garis besar, tujuan utama dari *physical database design* adalah untuk mendeskripsikan bagaimana desainer bermaksud untuk mengimplementasikan secara fisik dari *logical database design*.

Untuk model relasional, ini meliputi :

1. Membuat sejumlah atau kumpulan tabel relasional dan *constraints* pada tabel tersebut dari informasi yang didapat dalam model data logikal (*logical data model*).
2. Mengidentifikasi struktur penyimpanan tertentu dan metode akses terhadap data untuk mencapai performa optimal dari sistem basis data.
3. Merancang proteksi keamanan untuk sistem.

2.2 Teori-Teori Penjualan dan Pembelian

2.2.1 Pengertian Penjualan

Menurut kamus akuntansi (Widjajanto, 1999, p122), penjualan adalah transfer hak atas barang untuk mendapatkan sumber daya lainnya, seperti kas atau janji untuk membayar kas (piutang).

Menurut Swatsha (1999, p8) penjualan merupakan ilmu dan seni mempengaruhi pribadi yang dilakukan oleh penjual untuk mengajak orang lain agar bersedia membeli barang atau jasa yang ditawarkannya.

2.2.2 Pengertian Pembelian

Definisi pembelian berdasarkan kamus akuntansi (Widjajanto, 1999, p109) adalah suatu perkiraan yang digunakan untuk mencatat perolehan barang dagangan untuk dijual kembali atau bahan untuk digunakan dalam proses produksi.