# BAB 2

# LANDASAN TEORI

## 2.1 Teori - Teori Umum

### 2.1.1 Pengertian Sistem

Menurut Jogiyanto (1995, p1) terdapat dua kelompok pendekatan di dalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan yang lebih menekankan pada prosedur mendefinisikan sistem sebagai suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu. Pendekatan sistem yang merupakan jaringan kerja dari prosedur lebih menekankan urutan operasi dalam sistem. Menurut Richard F. Neuschel, prosedur adalah suatu urutan operasi klerikal (tulis-menulis), biasanya melibatkan beberapa orang di dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi.

Pendekatan sistem yang lebih menekankan pada elemen atau komponennya mendefinisikan sistem sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu.

### 2.1.2 Pengertian Data

Menurut Turban (2003, p2), data adalah fakta-fakta yang belum diolah atau gambaran-gambaran lebih lanjut dari benda-benda, kejadian-kejadian, kegiatan-kegiatan dan transaksi-transaksi yang ditangkap, direkam, disimpan, dan diklasifikasikan, tetapi tidak disusun untuk menyampaikan arti khusus lainnya

### 2.1.3 Pengertian Basis Data

Menurut Connoly (2005, p15), basis data adalah koleksi bersama dari data yang terkait secara logis, dan suatu deskripsi dari data, yang dirancang untuk memenuhi kebutuhan informasi dalam suatu organisasi.

Menurut McLeod (2004, p130), basis data adalah koleksi semua data yang berbasis komputer dalam suatu perusahaan.

Menurut Inmon (2002, p388) basis data adalah sebuah kumpulan dari data yang saling berhubungan yang disimpan (biasanya dengan redundansi yang terkontrol dan terbatas) berdasarkan suatu skema.

Menurut Date (2000, p10) basis data adalah koleksi dari *persistent data* yang digunakan oleh sistem aplikasi dari suatu perusahaan.

### 2.1.4 Database Management System *(DBMS)*

Menurut Connolly (2005, p16), DBMS diartikan sebagai suatu program yang mengontrol dan mengatur pengorganisasian, penyimpanan

dan pengambilan data dalam suatu basisdata. Selain bertanggungjawab atas keamanan (*security*) dan kesatuan (*integrity*) basisdata tersebut, DBMS juga menerima permintaan data dari program aplikasi yang kemudian memerintahkan sistem operasi untuk mengirimkan data yang dimaksudkan. DBMS merupakan program yang mempengaruhi program aplikasi pengguna dan basisdata. Beberapa komponen DBMS (Connolly, 2005, p18) :

- Perangkat Keras (*Hardware*)

  DBMS dan aplikasi-aplikasi memerlukan *hardware* untuk beroperasi. *Hardware* tersebut dapat berkisar dari komputer pribadi tunggal, ke suatu *mainframe* tunggal, ke suatu jaringan komputer.

- Perangkat Lunak *(Software)*

  Komponen *software* terdiri dari DBMS itu sendiri dan program aplikasi, bersama dengan sistem operasi, termasuk *software* jaringan jika DBMS sedang digunakan melalui suatu jaringan.

- Data

  Barangkali komponen yang paling penting dari lingkungan DBMS, tentu saja dari sudut pandang *end-users*, adalah data. Data berperan sebagai suatu jembatan antara komponen-komponen mesin dan komponen-komponen manusia.

- Prosedur

  Merupakan instruksi dan peraturan yang menguasai perancangan dan penggunaan basisdata. Pengguna sistem dan *staff* yang mengatur

basisdata yang memerlukan prosedur-prosedur yang telah didokumentasikan pada bagaimana menggunakan atau menjalankan sistem. Berikut ini terdiri dari instruksi-instruksi:

- Login ke dalam DBMS
- Menggunakan suatu fasilitas DBMS khusus atau program aplikasi
- Menjalankan dan menghentikan DBMS
- Membuat duplikat *backup* basisdata
- Menangani kesalahan-kesalahan *hardware* atau *software*
- Mengubah stuktur dari suatu tabel, mengatur ulang basisdata melewati *multiple disks*, meningkatkan kinerja, atau menyimpan data ke *secondary storage*.

- Manusia / orang.

  Terdiri dari :

  1. *Data dan Database Administrators.*

     - *Data Administrator* (DA) bertanggungjawab pada managemen sumber data termasuk perencanaan, pengembangan dan perawatan standar, *policies* dan prosedur, dan desain konseptual / logikal basisdata.
     - *Database Administrator* (DBA) bertanggungjawab atas realisasi fisikal basisdata, termasuk perancangan basisdata fisikal dan implementasi, keamanan dan kontrol integritas, perawatan sistem operasional, dan menyakinkan kinerja aplikasi yang memuaskan untuk pengguna.

2. *Database Designers.*

   Dibedakan menjadi 2 tipe desainer :

   - *Logical database designer* : mengidentifikasi data, relasi antara data, dan konstrain pada data yang akan di simpan di basisdata.
   - *Physical database designer* : menentukan bagaimana desain basisdata logikal bisa direalisasikan secara fisikal.

3. *Application Developers.*

   *Application developers* bertanggungjawab atas pengimplementasian fungsionalitas yang di perlukan *end-users* pada program aplikasi setelah basisdata diimplementasikan.

4. *End – Users*

   *End-users* adalah 'klien' untuk basisdata, yang telah dirancang dan diimplementasikan, dan sedang dirawat untuk menyajikan kebutuhan-kebutuhan informasi mereka. *End-users* dapat diklasifikasikan berdasarkan cara mereka menggunakan sistem: *Naive users* dan *Sophisticated users*.


Gambar 2.1 Komponen DBMS

(Sumber: Connoly, T. and Begg, C, *Database Systems : A Practical Approach to Design, Implementation, and Management*. 2005, Page 19)

### 2.1.5 Keuntungan dan Kerugian DBMS

DBMS memiliki keuntungan dan kerugian (Connolly, 2005, p26-p30) :

- Keuntungan DBMS :

  1. Kontrol terhadap pengulangan data (*data redundancy*).
  2. Data yang konsisten.
  3. Semakin banyak informasi yang didapat dari data yang sama.
  4. Data yang dibagikan (*sharing data*).
  5. Menambah integritas data.
  6. Menambah keamanan data.
  7. Penetapan standarisasi.
  8. Pengurangan biaya.
  9. Mempermudah pengoperasian data.
  10. Memperbaiki pengaksesan data dan hasilnya.
  11. Menambah produktivitas.
  12. Memperbaiki pemeliharaan data melalui independensi data.
  13. Memperbaiki pengaksesan data secara bersama-sama.

- Kerugian DBMS :

  1. Kompleksitas.
  2. Size / ukuran besar.
  3. Biaya dari suatu DBMS.
  4. Biaya penambahan perangkat keras.

### 2.1.6 Database Languages

#### 2.1.6.1 Data Definition Language (DDL)

Menurut Connolly (2005, p40), DDL (*Data Definition Language*) adalah suatu bahasa yang memungkinkan *Database Administrator* (DBA) atau pengguna untuk mendefinisikan dan memberi nama entitas, atribut dan relasi yang dibutuhkan aplikasi dengan kesatuan integrasi dan batasan *security*. DDL memungkinkan pembuatan dan penghancuran objek-objek yang ada dalam *database* seperti skema, domain, *table*, *view*, dan indeks.

Menurut Fathansyah (1999, p15), DDL adalah struktur atau skema basisdata yang menggambarkan desain basisdata secara keseluruhan dispesifikasikan dengan bahasa khusus. Hasil dari kompilasi perintah DDL adalah kumpulan tabel yang disimpan dalam file khusus yang disebut Kamus Data (*Data Dictionary*). **Kamus Data** merupakan suatu metadata yaitu data yang mendeskripsikan data yang sesungguhnya.

Beberapa statement DDL (Connolly, 2005, p168 ) :

1. *Create Table*

   Untuk membuat tabel dengan mengidentifikasikan tipe data untuk tiap kolom.

2. *Alter Table*

   Untuk menambah atau membuang kolom dan constrain.

3. *Drop Table*

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

4. *Create Index*

   Untuk membuat indeks pada suatu tabel.

5. *Drop Index*

   Untuk membuang atau menghapus indeks yang telah dibuat sebelumnya.

#### 2.1.6.2 Data Manipulation Language (DML)

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

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

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

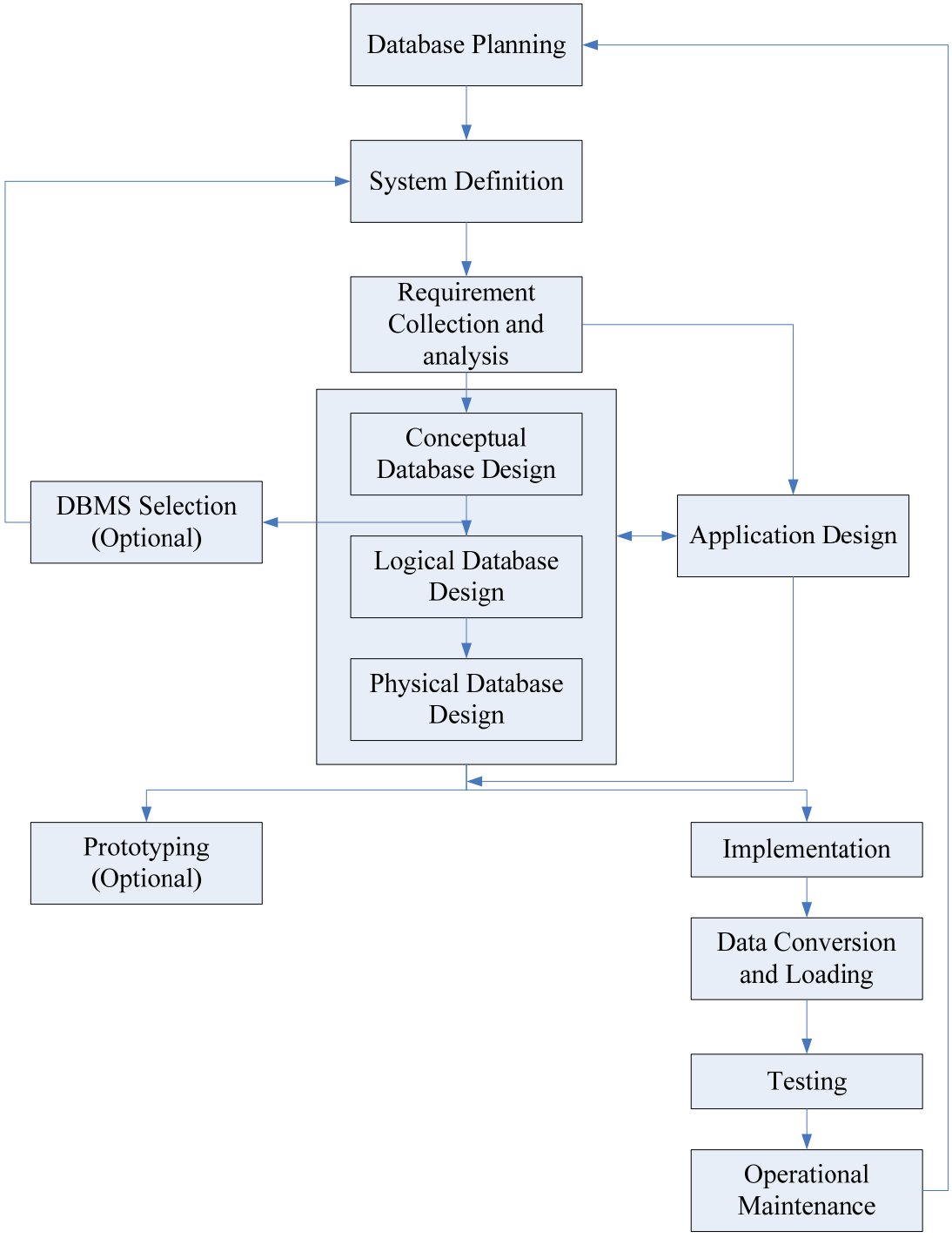
**a. Prosedural DML**

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

**b. Non-Procedural DML**

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

### 2.1.7 *Daur Hidup Database (Database Life Cycle)*




Gambar 2.2 Siklus Hidup Aplikasi Basisdata

(Sumber: Connoly, T. and Begg, C, *Database Systems : A Practical Approach to Design, Implementation, and Management*. 2005, Page 284)

Berikut ini ringkasan dari aktivitas utama yang ada di setiap langkah dari siklus hidup aplikasi basis data, antara lain :

#### 2.1.7.1 Perencanaan Basis Data (*Database Planning*)

Merencanakan bagaimana tahapan dari siklus hidup bisa direalisasikan secara efektif dan efisien. Tahapan perencanaan basis data :

- Mengidentifikasi rencana dan tujuan pembuatan aplikasi basis data untuk menetapkan kebutuhan sistem informasi.
- Mengevaluasi sistem yang sudah ada untuk menentukan kelebihan dan kekurangannya
- Menaksir kesempatan teknologi informasi untuk menghasilkan kelebihan.

#### 2.1.7.2 Pendefinisian Sistem *(System Definition)*

Menspesifikasikan ruang lingkup dan batasan dari aplikasi basis data, pengguna dan area aplikasi. Selain itu, pada tahap ini perlu diidentifikasikan pandangan pemakai (*user view)* yang mendefinisikan apa yang diperlukan dalam aplikasi basis data dari peran tertentu seperti manajer atau *supervisor* ataupun keseluruhan area aplikasi (*Marketing*, Personalia, dan bagian gudang).

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

Mengumpulkan dan menganalisa informasi tentang bagian

organisasi yang harus didukung oleh aplikasi basis data. Informasi ini kemudian dianalisa untuk mengidentifikasikan kebutuhan pemakai dari sistem basis data yang baru.

Informasi yang dikumpulkan mencakupi :

- Gambaran dari data yang digunakan dan diolah.
- Rincian mengenai bagaimana data akan digunakan dan diolah.
- Kebutuhan tambahan untuk aplikasi basis data yang baru.

#### 2.1.7.4 Perancangan Basis Data *(Database Design)*

Terdapat dua pendekatan utama yang digunakan dalam merancang basis data, yakni pendekatan secara *bottom-up* dan pendekatan secara *top-down*.

- Pendekatan secara *bottom-up* dilakukan melalui analisis terhadap atribut (properti dari *entity* dan *relasi*) beserta asosiasinya. Proses normalisasi merupakan representasi pendekatan secara *bottom-up* ini. Normalisasi mencakup pengidentifikasian atribut-atribut yang diperlukan beserta agregasi berikutnya menjadi relasi yang telah dinormalkan dengan berdasarkan pada ketergantungan fungsional diantara atribut-atribut.
- Pendekatan secara *top-down* dilakukan dengan terlebih membangun model data tingkat tinggi guna kemudian membangun model data yang lebih sederhana. Pendekatan ini diilustrasikan melalui konsep model *Entity-Relationship (ER)*.

#### 2.1.7.5 Pemilihan DBMS (DBMS Selection)

Proses memilih DBMS yang cocok untuk mendukung aplikasi basis data. Langkah-langkah dalam pemilihan DBMS, antara lain :

- Menentukan tujuan penelitian dan tugas yang harus bisa dilakukan DBMS.
- Menyeleksi beberapa DBMS yang memenuhi kriteria yang dibutuhkan seperti harga, perangkat keras yang didukung, atau *platform* yang didukung.
- Membandingkan kinerja beberapa DBMS yang sudah diseleksi berdasarkan fitur yang tersedia atau kinerjanya.
- Memilih produk DBMS yang terbaik dan membuat dokumentasi dari tahapan pemilihan tersebut.

#### 2.1.7.6 Perancangan Aplikasi *(Application Design)*

Merancang antarmuka pemakai dan program aplikasi yang menggunakan dan mengolah basis data. Dua aspek perancangan aplikasi :

1) Perancangan transaksi
   - Transaksi pengambilan data dari basis data untuk ditampilkan di layar maupun untuk pembuatan laporan *(retrieval transaction)*
   - Transaksi penambahan baris baru, menghapus baris lama dan memodifikasi baris dalam basis data *(update*

*transaction).*

- Transaksi campuran yang melibatkan pengambilan dan pembaharuan data *(mixed transaction)*

2) Pedoman perancangan antarmuka pemakai

- Judul yang berarti
- Instruksi yang dapat dimengerti
- Pengelompokan dan pengurutan kolom-kolom
- Layout *form* atau laporan yang konsisten
- Label kolom yang familiar
- Istilah dan singkatan yang konsisten
- Ruang dan batasan yang jelas untuk kolom pemasukan data
- Pergerakkan kursor yang nyaman
- Pengeditan kesalahan karakter dan nilai yang tidak sesuai
- Kolom-kolom opsional ditandai dengan jelas
- Pesan penjelasan untuk kolom-kolom yang ada
- Tanda penyelesaian suatu transaksi

### 2.1.7.7 Prototipe (Prototyping)

Membangun model aplikasi basis data dimana pemakai dan pembuat aplikasi dapat melihat dan mengevaluasi bagaimana sistem akhir akan terlihat dan bagaimana fungsi-fungsinya dioperasikan. Prototipe digunakan agar pemakai dapat mengetahui fitur dalam aplikasi basis data dan jika memungkinkan memberikan pendapat atau masukan terhadap aplikasi basis data.

#### 2.1.7.8 Implementasi (Implementation)

Penerapan dan perwujudan fisikal dari basis data dan perancangan aplikasi. Implementasi basis data didapat dari penerapan *Data Definition Language* (DDL) dan *Graphical User Interface* (GUI). Bahasa program yang biasa digunakan, antara lain : Visual Basic, Delphi, C++, Java, Pascal, Cobol, Ada. Keamanan dan *integrity* juga diimplementasikan pada tahap ini. Biasanya digunakan program yang mendukung *Structure Query Language* ( SQL ), seperti MySQL, SQL Server dan Oracle.

#### 2.1.7.9 Konversi Data dan Loading (Data Conversion and Loading)

Proses transfer data yang sudah ada ke dalam basis data yang baru dan mengkonversi aplikasi yang sudah ada untuk dijalankan di basis data baru.

#### 2.1.7.10 Pengujian *(Testing)*

Proses mengeksekusi program aplikasi untuk mencari kesalahan sehingga dapat diketahui apakah basis data dan program aplikasinya bekerja sesuai dengan spesifikasi dan harapan kita. Situasi yang ideal dalam pengujian juga harus melibatkan pemakai.

#### 2.1.7.11 Perawatan Operasional (Operational Maintenance)

Proses memonitor dan merawat sistem setelah pemasangan atau peng*install*an. Proses perawatan meliputi aktivitas :

- Mengawasi kinerja sistem. Apabila kinerja sistem menurun di

bawah tingkat yang diharuskan, diperlukan perbaikan atau pengorganisasian basis data.

- Memperbaharui basis data apabila diperlukan.

### 2.1.8 Entity–Relationship Modeling

Menurut Connolly (2005, p342), salah satu aspek yang sulit dalam perancangan basis data adalah kenyataan bahwa perancang, programer, dan pemakai akhir cenderung melihat data dengan cara yang berbeda. Untuk memastikan pemahaman secara alamiah dari data dan bagaimana data digunakan oleh perusahaan diperlukan sebuah model komunikasi yang non-teknis dan bebas dari kebingungan. Model *Entity Relationship* (ER) adalah salah satu contohnya. Pemodelan ER merupakan pendekatan *top-down* pada perancangan basis data yang dimulai dengan mengidentifikasi data yang penting yang disebut entitas dan relasi antar data yang harus direpresentasikan dalam model. Kemudian menambahkan detail lainnya seperti informasi tentang entitas dan relasi yang dinamakan atribut, serta batasan pada entitas, relasi dan atribut.

- Diagram *Entity-Relationship*

  Penggambaran Model ER secara sistematis dilakukan melalui diagram ER. Notasi-notasi simbolik di dalam Diagram ER yang dapat digunakan adalah:

  - Persegi panjang.

    Menyatakan entitas

- Lingkaran / Elips

  Menyatakan atribut (Atribut yang berfungsi sebagai key digaris bawahi).

- Belah ketupat

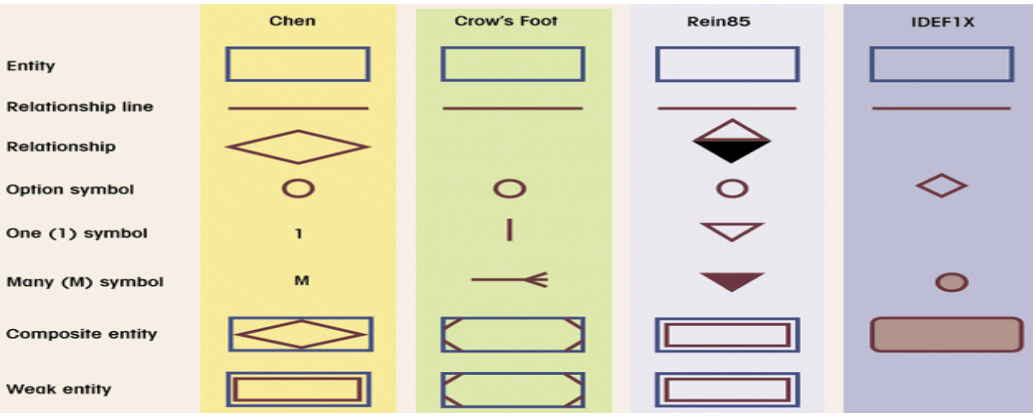
  Menyatakan relasi.

- Garis

  Sebagai penghubung antara relasi dengan entitas dan entitas dengan atributnya.

- Derajat Relasi

  Dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka (1 dan 1 untuk relasi *one to one*, 1 dan N untuk relasi *one to many* atau N dan N untuk relasi *many to many*).



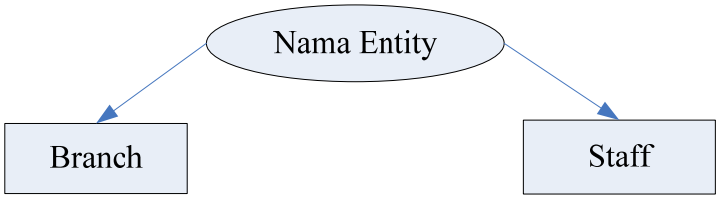

Gambar 2.3 Derajat Relasi

### 2.1.8.1 Tipe Entitas (*Entity Types*)

Menurut Connolly (2005, p343), Tipe entity adalah kumpulan dari objek dengan *property* yang sama, yang diidentifikasi oleh perusahaan mempunyai keberadaan yang

independen. Keberadaan tipe entitas dapat berupa fisik atau ‘nyata’ dan *conceptual* atau ‘abstrak’.

*Entity occurrence* adalah pengidentifikasian objek yang unik dari sebuah tipe entitas.

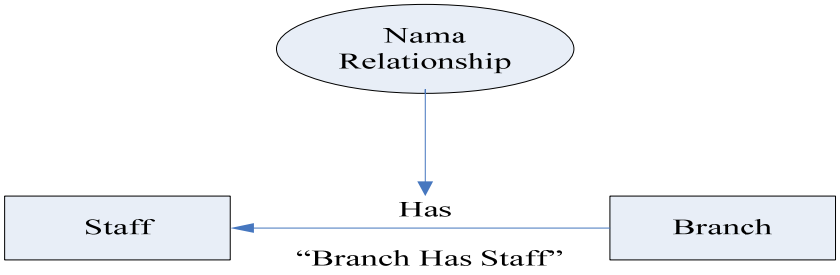



Gambar 2.4 Contoh tipe entitas

(Sumber Connolly dan Begg, 2005, p345)

### 2.1.8.2 Tipe Relasi (Relationship Types)

Merupakan kumpulan hubungan yang mempunyai arti antara tipe entitas yang ada (Connoly, 2005, p346). Setiap tipe relasi diberi nama yang mendeskripsikan fungsinya. Hal yang penting adalah kita dapat membedakan *relationship type* dengan *relationship occurrence*. *Relationship occurrence* merupakan keterhubungan yang diidentifikasi secara unik meliputi keberadaan setiap tipe entitas yang berpartisipasi. Derajat relationship, yaitu jumlah entitas yang berpartisipasi dalam suatu *relationship*.

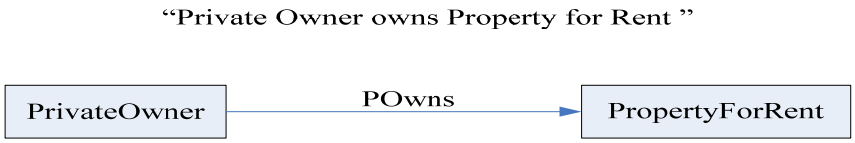



Gambar 2.5 Contoh Relationship Types

Derajat relationship terdiri dari:

- *Binary Relationship*

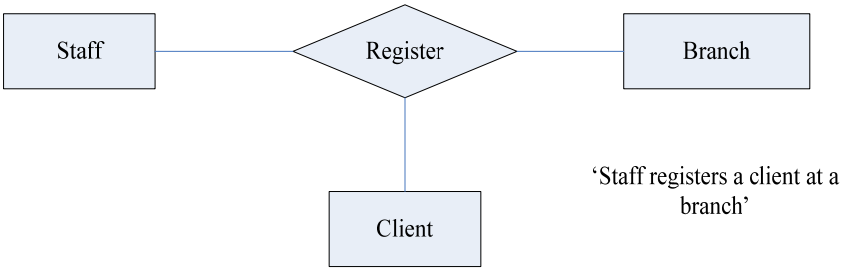
  Keterhubungan antara dua tipe entitas.




Gambar 2.6 Contoh Binary Relationship

- *Ternary Relationship*

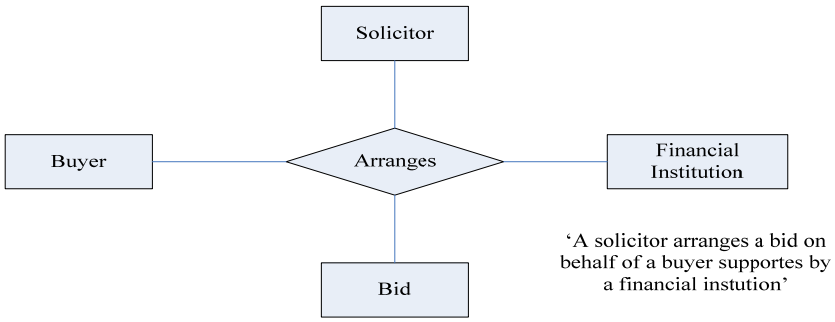
  Keterhubungan antara tiga tipe entitas.




Gambar 2.7 Contoh *Ternary Relationship*

- *Quaternary Relationship*

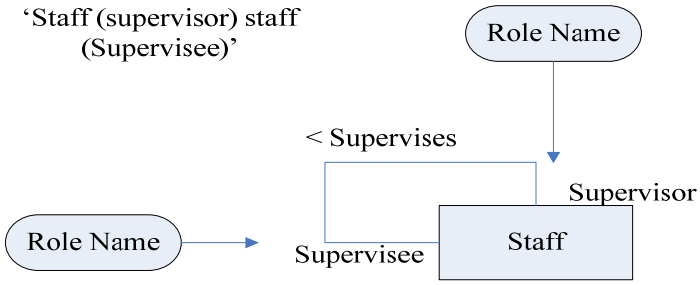
  Keterhubungan antara empat tipe entitas.




Gambar 2.8 Contoh Quaternary Relationship

- *Recursive relationship*

  Keterhubungan antara satu tipe entitas, dimana tipe entitas tersebut berpartisipasi lebih dari satu kali dengan peran yang berbeda. *Relationship* dapat diberikan *role name* untuk mengidentifikasikan keterkaitan tipe entitas dalam *relationship*.




Gambar 2.9 Contoh Recursive Relationship

### 2.1.8.3 Atribut–Atribut

Menurut Connolly (2005, pp350-354), Atribut adalah sifat atau *property* dari sebuah tipe entitas atau tipe *relationship*. Contohnya : sebuah entitas karyawan digambarkan oleh kdKary,

nmKary, almtKary dan jabatan. Atribut domain adalah himpunan nilai yang diperbolehkan untuk satu atau lebih atribut. Macam-macam atribut :

1) *Simple Attribute* disebut juga *Atomic Attribute*, yaitu atribut yang terdiri dari satu komponen tunggal dengan keberadaan yang independen dan tidak dapat dibagi menjadi bagian yang lebih kecil lagi.
2) *Composite Attribute* yaitu atribut yang terdiri dari beberapa komponen dimana masing-masing komponen memiliki keberadaan yang independen. Misalkan atribut alamat dapat terdiri dari jalan, kota dan kode pos.
3) *Single-valued Attribute* yaitu atribut yang mempunyai nilai tunggal untuk setiap kejadian. Misalnya entitas pemesanan memiliki satu nilai untuk atribut noPesan pada setiap kejadian.
4) *Multi-valued Attribute* yaitu atribut yang mempunyai beberapa nilai untuk setiap kejadian. Misalnya entitas Karyawan memiliki beberapa nilai untuk atribut noTelp pada setiap kejadian.
5) *Derived Attribute* yaitu atribut yang memiliki nilai yang dihasilkan dari satu atau beberapa atribut lainnya, dan tidak harus berasal dari satu entitas.
6) *Keys*

   Sebuah *key* memberikan kemudahan untuk mengidentifikasi

kumpulan atribut yang mencukupi untuk dapat membedakan entitas yang satu dengan yang lain. *Key* juga membantu mengidentifikasi *relationship* secara unik dan membedakan *relationship* antara yang satu dengan yang lainnya.

Menurut Connolly (2005, pp352-354), *Key* terbagi atas :

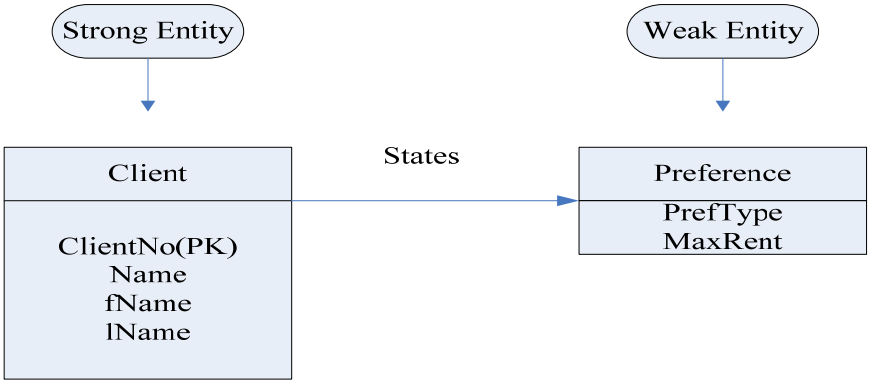
- *Candidate Key* adalah jumlah minimal atribut – atribut yang secara unik mengidentifikasikan setiap kejadian pada tipe entitas.
- *Primary Key* adalah *candidate key* yang terpilih untuk mengidentifikasikan setiap kejadian tipe entitas secara unik.
- *Composite Key* adalah *candidate key* yang terdiri dari dua atau lebih atribut.
- *Super Key* : atribut atau kumpulan atribut yang menentukan suatu *tuple* secara unik dalam suatu relasi.
- *Foreign Key*: atribut atau kumpulan atribut dalam suatu relasi yang sesuai dengan *candidate key* dari beberapa relasi.

#### 2.1.8.4 *Strong and Weak Entity Types*

*Strong Entity Types* adalah entitas yang keberadaannya tidak bergantung pada entitas lain (Connoly, 2005, p354).

*Weak Entity Types* adalah entitas yang keberadaanya bergantung pada entitas lain (Connoly, 2005, p355).

*Strong entity* terkadang disebut dengan *parent* dan *weak entity type* disebut *child*.

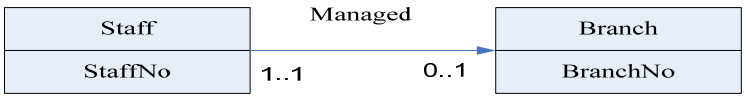



Gambar 2.10 Strong and Weak Entity

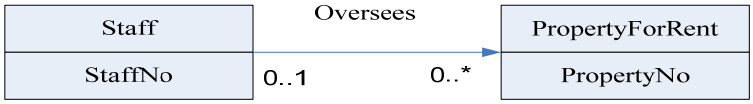
### 2.1.8.5 *Structural Constrains*

Batasan atau kendala utama pada *relationship* adalah *multiplicity*. Menurut Connolly (2005, p356), *Multiplicity* adalah jumlah (atau jangkauan) dari kejadian yang mungkin terjadi pada suatu entitas yang terhubung ke satu kejadian dari entitas lain yang berhubungan melalui suatu *relationship*. Menurut Connolly (2005, pp356-360), ada 3 tipe *relationship binary* yang menggunakan batasan *enterprise* yaitu :

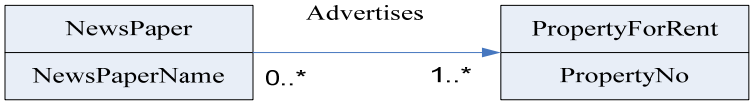
1) Relasi satu-satu (1:1) yaitu relasi antara dua tipe entitas dimana satu tipe entitas berelasi tepat satu atau nol dengan tipe entitas lainnya.
2) Relasi satu-banyak (1:*) yaitu relasi antara dua tipe entitas dimana satu tipe entitas berelasi nol, satu atau banyak dengan tipe entitas lainnya.
3) Relasi banyak-banyak (*:*)yaitu relasi antara dua tipe entitas dimana tipe entitasnya saling berelasi nol, satu atau banyak.




Relasi 1:1




Relasi 1:*

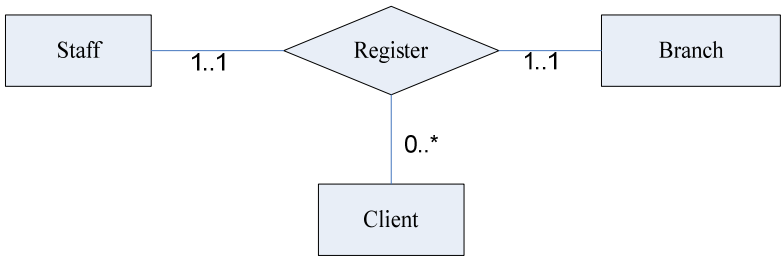



Relasi *:*

Gambar 2.11 Contoh Tipe Relationship pada Binary

4) *Mutliplicity* untuk relasi yang komplek

Menurut Connolly (2005, p361), *Multiplicity* untuk relasi yang kompleks adalah jumlah (atau jangkauan) dari kejadian yang mungkin dari suatu entitas dalam n-ary *relationship* ketika nilai entitas yang lain (n-1) diketahui.



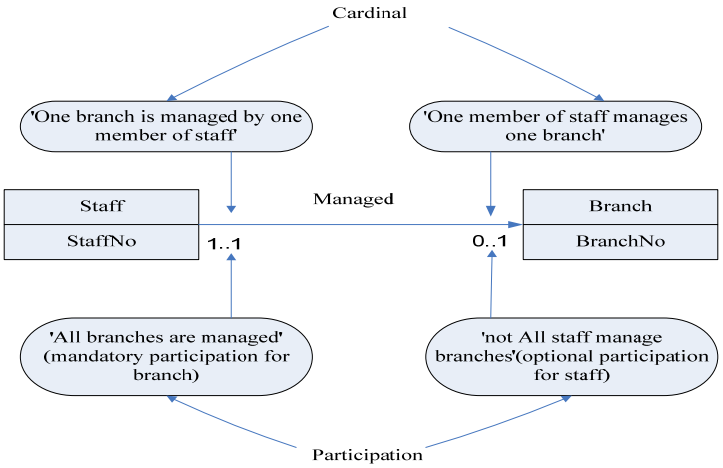

Gambar 2.12 Contoh Multiplicity pada Relasi Ternary

5) Cardinality dan Participation Constrains

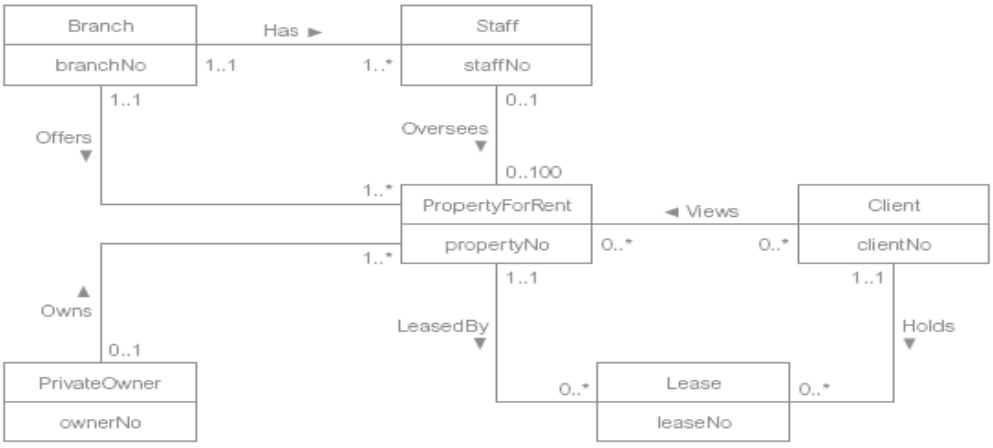
*Cardinality*, menjelaskan jumlah maksimal dari kejadian *relationship* yang mungkin untuk entitas yang berpartisipasi di

dalam *relationship* tersebut.

*Participation*, menetapkan apakah seluruh atau sebagian entitas yang berpartisipasi dalam suatu relationship.




Gambar 2.13 Cardinal and Participation




Gambar 2.14 Contoh ERD

### 2.1.9 Normalisasi

Menurut Hoffer (2002, p189), normalisasi adalah proses formal dalam menentukan atribut mana yang harus dikelompokkan bersama dalam satu relasi.

Menurut Post (2002, p66), relasi basisdata mengoperasikan tabel-tabel data dan tabel-tabel tersebut harus didefenisikan untuk mendapat keuntungan dari basisdata. Proses penentuan dari tabel-tabel yang sesuai tersebut untuk suatu basisdata disebut normalisasi.

Menurut Connolly (2005, p388), normalisasi merupakan suatu teknik untuk menghasilkan kumpulan relasi-relasi dengan properti yang diperlukan, untuk menyediakan kebutuhan data dari perusahaan. Normalisasi adalah suatu teknik formal untuk menganalisa relasi berdasarkan *primary key* dan ketergantungan fungsional diantara atribut tiap tabelnya.

Tujuan normalisasi adalah terjaminnya struktur yang konsisten, kerangkapan yang minimal, dan stabilitas struktur data yang maksimal.

Manfaat normalisasi:

1. Meminimalkan jumlah kapasitas penyimpanan yang diperlukan untuk menyimpan data.
2. Meminimalkan resiko data yang tidak konsisten dalam suatu basisdata.
3. Meminimalkan kemungkinan *update* dan *delete anomally.*
4. Memaksimalkan stabilitas dari struktur data.

Tabel yang belum dinormalisasikan disebut *Unnormalized Form*

(UNF). Menurut Connolly (2005, p403) *Unnormalized form* (UNF) adalah merupakan suatu tabel yang berisikan satu atau lebih *group* yang berulang. Bentuk-bentuk normalisasi antara lain :

1. *First Normal Form* (1NF)

   Menurut Connolly (2005, p403), *First Normal Form* adalah relasi dimana pertemuan antar setiap baris dan kolom terdiri 1 (satu) dan hanya 1 (satu) nilai. Dalam normalisasi pertama ini, data yang berulang-ulang dihilangkan.

2. *Second Normal Form* (2NF)

   Menurut Connolly (2005, p407), *Second normal form* (2NF) adalah merupakan sebuah relasi dalam 1NF yang setiap atribut non-*primary key* bersifat *Full Function Dependency* pada *primary key* dari relasi tersebut. Dalam normalisasi kedua ini, atribut yang tergantung pada sebagian dari suatu *composite key* sebuah tabel dipindahkan ke sebuah tabel yang terpisah.

   Menurut Connolly (2005, p412), *Full functional dependency* adalah mengindikasikan bahwa jika A dan B adalah atribut dari sebuah relasi, B adalah *fully functional dependent* dari A jika B adalah *functionally dependent* dari A tapi bukan merupakan bagian dari A.

3. *Third Normal Form* (3NF)

   Menurut Connolly (2005, p409), *Third normal form* (3NF) adalah sebuah relasi yang memenuhi normal pertama dan normal kedua dimana tidak terdapat atribut *non primary key* yang bersifat

*transitively dependent* dari *primary key*-nya. Menurut Connolly (2005, p412), *Transitive dependency* adalah jika kondisi A, B, dan C merupakan atribut-atribut dari sebuah relasi seperti jika A→B dan B→C, maka C adalah *transitively dependent* dari A melalui atau via B ( disediakan bahwa A bukan *functionally dependent* dari B dan C ) Dalam normalisasi ketiga ini, atribut yang tidak memberikan kontribusi terhadap penjelasan karakteristik *primary key*, akan dipindahkan ke sebuah tabel yang terpisah. Keuntungan dari tabel relasional dalam 3NF adalah menghilangkan data yang berulang-ulang dengan tujuan menghemat tempat dan mengurangi keanehan manipulasi. Boyce-Codd Normal Form (BCNF). Menurut Connolly (2005, p.419) BCNF adalah sebuah relasi di mana setiap penentu atau determinan adalah *candidate key*.

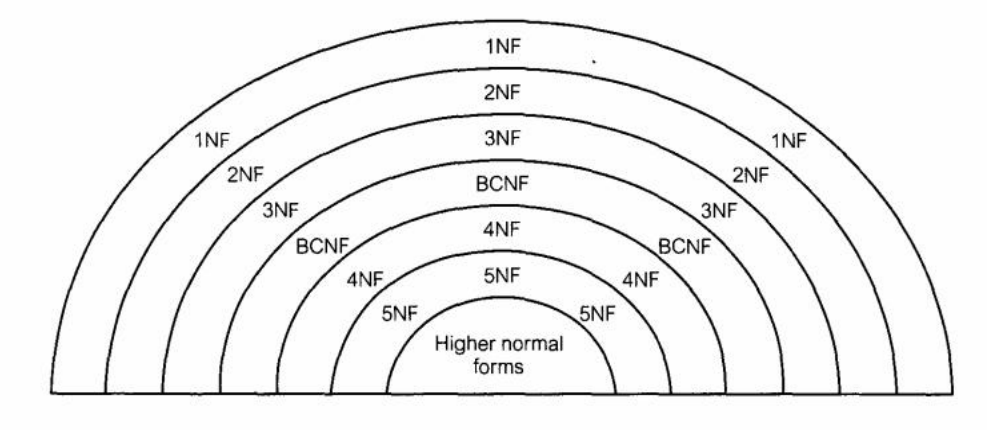
4. *Fourth Normal Form* (4NF).

   Normalisasi keempat dilakukan untuk menghilangkan *multi-valued dependency*.

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

   Normalisasi kelima menyebabkan relasi tidak mempuyai *join dependency*. Untuk relational data model, sangat penting untuk mengetahui bahwa hanya normal pertama saja (1NF) yang secara kritis dapat menciptakan relasi-relasi yang dibutuhkan. Semua normal form yang berurutan selanjutnya seperti normal kedua (2NF), normal ketiga (3NF), dan seterusnya sampai normal kelima (5NF) adalah optional

atau pilihan. Bagaimanapun juga semua bertujuan untuk menghindari *update anomalies*, maka dari itu secara normal disarankan untuk melakukan proses normalisasi paling tidak sedikitnya sampai dengan normal yang ketiga (3NF).




Gambar 2.15 Ilustrasi diagram dari hubungan diantara normal forms
(Sumber :Connolly, 2005, p401)

### 2.1.10 Metodologi Perancangan

Perancangan basis data terbagi kedalam tiga tahapan utama, yakni perancangan basis data secara konseptual, logikal, dan fisikal (Connolly, 2005, p439)

#### 2.1.10.1 Perancangan Basis Data Konseptual *(Conceptual Database Design).*

Menurut Connolly dan Begg (2005, p439), perancangan konseptual basis data adalah suatu proses membangun sebuah model dari informasi yang digunakan oleh perusahaan, independen pada semua pertimbangan fisik.

Langkah-langkah yang dilakukan pada tahap perancangan konseptual basis data adalah sebagai berikut :

1. Mengidentifikasi tipe *entitas*
2. Mengidentifikasi tipe relasi
3. Identifikasi dan asosiasi atribut dengan entitas-entitas atau tipe relasi.
4. Mengidentifikasi atribut domain
5. Mengidentifikasi atribut *candidate key*, *primary, alternate key*
6. Mempertimbangkan kegunaan dari konsep *enhanced modeling* (optional)
7. Mengecek relasi yang redudansi
8. Validasikan model konseptual lokal terhadap transaksi pengguna
9. Review konseptual data model dengan pengguna.

#### 2.1.10.2 Perancangan Basis Data Logikal (*Logical Database Design*)

Menurut Connolly dan Begg (2005, p439), perancangan logikal basis data merupakan proses membangun sebuah model informasi yang digunakan dalam sebuah perusahaan berdasarkan pada sebuah model data yang spesifik, tetapi tidak bergantung pada sebuah DBMS tertentu dan pertimbangan-pertimbangan fisikal lainnya.

Basis data logikal ada dua, yaitu membangun dan memvalidasi model data logikal bagi masing-masing *view* dan

model data logikal globalnya. Tahap-tahap yang dilakukan dalam membangun dan memvalidasi model data logikal bagi masing-masing *view* adalah:

1. Menghilangkan fitur-fitur yang tidak *compatible* dengan model relasional, diantaranya menghilangkan *many-to-many binary relationship types*, *many-to-many recursive relationship types*, relasi kompleks, dan menghilangkan atribut yang *multi-valued*.
2. Menurunkan relasi untuk model data logikal dengan cara:
    - Membuat relasi antara *strong entity* yang ada. Untuk atribut *composite* seperti nama, cantumkan yang penting saja seperti fName (nama depan) dan lName (nama belakang).
    - Untuk *weak entity types*, *primary key*-nya diturunkan dari setiap *owner entity*.
    - Unt*uk one-to-many binary relationship*, *entity* yang ada di satu sisi ditentukan sebagai *parent entity* dan sisi yang lain sebagai *child entity*.
    - Untuk *one-to-one binary relationship*, apabila terdapat *mandatory participation* pada kedua sisi, gabungkan entitas yang terlibat menjadi satu tabel dan pilih salah satu *primary key* dari entitas asalnya menjadi *primary key* pada tabel yang baru dan *primary key* yang lainnya digunakan sebagai *alternate key*. Apabila terdapat *mandatory participation* pada sisi pertama, maka yang harus dilakukan adalah

menentukan entitas *parent* dan *child* dari kedua tabel. *Primary key* pada tabel *parent* akan menjadi *primary key* juga di tabel *child*. Dan apabila terjadi *optional participation* pada kedua sisi, maka harus ditentukan *primary key* dari tabel mana yang akan di-*copy* ke tabel lain yang berhubungan.

- Untuk *superclass / subclass relationship types*, identifikasi entitas *superclass* sebagai *parent entity* dan entity *subclass* sebagai *child entity*.
- Untuk *many-to-many binary relationship types*, buat sebuah tabel untuk merepresentasikan *relationship* dan beberapa atribut yang menjadi bagian dari *relationship* tersebut. Letakkan *primary key* dari entitas-entitas yang berhubungan ke tabel yang baru sebagai *foreign key*. *Foreign key* tersebut juga akan menjadi *primary key* pada tabel yang baru.
- Untuk relasi kompleks, buat sebuah tabel yang merepresentasikan *relationship* dan beberapa atribut yang menjadi bagian dari *relationship* tersebut. Kemudian letakkan *primary key* dari entitas-entitas yang berelasi kompleks ke dalam tabel yang baru dibuat sebagai *foreign key*. *Foreign key* yang mewakili relasi many, misalnya 0...*

atau 1...* akan membentuk *primary key* juga pada table yang baru.

- Untuk atribut *multi-valued*, buat sebuah tabel yang merepresentasikan atribut *multi-valued* dan *primary key* dari tabel yang lama menjadi *foreign key* pada tabel yang dibuat.

3. Menvalidasi relasi menggunakan normalisasi
4. Memastikan bahwa model logikal yang dibuat dapat mendukung kebutuhan *view*.
5. Menentukan batasan-batasan seperti data yang dibutuhkan, batasan atribut domain, *entity integrity*, *referential integrity*, dan *enterprise constraint*.
6. *User* melakukan *review* terhadap model logikal yang telah dibuat, untuk memastikan bahwa model yang dibuat merupakan representasi dari *view* yang akan dibuat.

Kemudian langkah selanjutnya adalah merancang dan memvalidasi model data logikal global dengan mengkombinasikan model data logikal lokal yang telah dibuat. Tahap-tahap yang dilakukan dalam merancang dan memvalidasi model data logikal global adalah:

1. Menggabungkan model data logikal lokal menjadi model data logikal global dengan cara:

- *Review* nama dan isi setiap entitas beserta *candidate key* dari entitas tersebut.
- *Review* nama dan isi *foreign key* pada tiap entitas.
- Gabungkan entitas-entitas yang ada pada model data logical lokal.
- Masukkan (tanpa penggabungan) entitas yang unik pada tiap model data logikal lokal.
- Gabungkan *relationship / foreign key* model data logical lokal.
- Masukkan (tanpa penggabungan) *relationship / foreign key* yang unik pada setiap model data logikal lokal.
- Cek apakah masih ada *missing entity* atau *relationship / foreign key*.
- Cek *foreign key*.
- Cek *integrity constraint*.
- Gambarlah ERD globalnya.
- *Update* dokumentasi yang telah dibuat.

2. Validasi model data logikal global menggunakan teknik normalisasi untuk memastikan model data yang dibuat mendukung kebutuhan user.
3. Cek perkembangan di masa yang akan datang untuk menentukan apakah ada perubahan yang signifikan di masa

yang akan datang untuk memastikan bahwa model data logikal yang dibuat dapat mengakomodasi perubahan tersebut.

4. *Review* model data logikal global dengan *user* untuk memastikan bahwa model data logikal yang dibuat merupakan representasi perusahaan yang sebenarnya.

#### 2.1.10.3 Perancangan Basis Data Fisikal ( *Physical Database Design*)

1. Merancang relasi dasar

   Untuk memutuskan bagaimana caranya menggambarkan identitas relasi dasar dalam model data logikal global dalam target DBMS.

2. Merancang reperesentasi dari data turunan

   Untuk memutuskan bagaimana menggambarkan data yang diperoleh, disajikan dalam model data logikal global dalam target DBMS.

3. Merancang *enterprise constraints*

   Untuk merancang *enterprise constraints* untuk target DBMS. *Enterprise constraints* kadang-kadang disebut sebagai peraturan-peraturan bisnis.

4. Menganalisa transaksi

   Untuk mengerti fungsi dari transaksi yang akan dijalankan pada basis data dan untuk menganalisa transaksi-transaksi penting.

5. Memilih organisasi file

Untuk menentukan organisasi file yang efisien untuk setiap relasi dasar, salah satu tujuan penting dari desain basis data fisikal adalah untuk menyimpan data secara efisien.

6. Memilih index

   Untuk menentukan apakah terjadinya penambahan index akan meningkatkan kinerja sistem.

7. Memperkirakan keperluan ruang penyimpanan file

   Untuk memperkirakan dasarnya ruang penyimpanan file yang dibutuhkan oleh basis data.

8. Merancang tampilan user

   Merancang tampilan user yang diidentifikasikan selama pengumpulan kebutuhan-kebutuhan dan analisis tingkatan relasi daur hidup aplikasi basis data.

9. Merancang mekanisme keamanan

   Untuk mendesain keamanan basis data yang dispesifikasi oleh user. Pada umumnya ada dua jenis keamanan basis data:

   - Keamanan Sistem

     Menangani akses dan penggunaan basis data pada level sistem, seperti *user* dan *password*.

   - Keamanan Data

     Menangani akses dan penggunaan objek basis data (seperti relasi dan tampilan) dan tindakan yang *user* dapat peroleh pada objek.

10. Mempertimbangkan pengenalan *redundancy* yang terkontrol

    Untuk menentukan apakah pengenalan redundancy dalam tata cara yang terkontrol oleh peraturan-peraturan normalisasi akan meningkatkan kinerja sistem.

11. Mengawasi dan meningkatkan kinerja sistem operasi

    Memonitor sistem operasional dan meningkatkan kinerja dari sistem untuk memperbaiki ketidaklayakan keputusan-keputusan desain.

### 2.1.11 Definisi *Data Flow Diagram* (DFD)

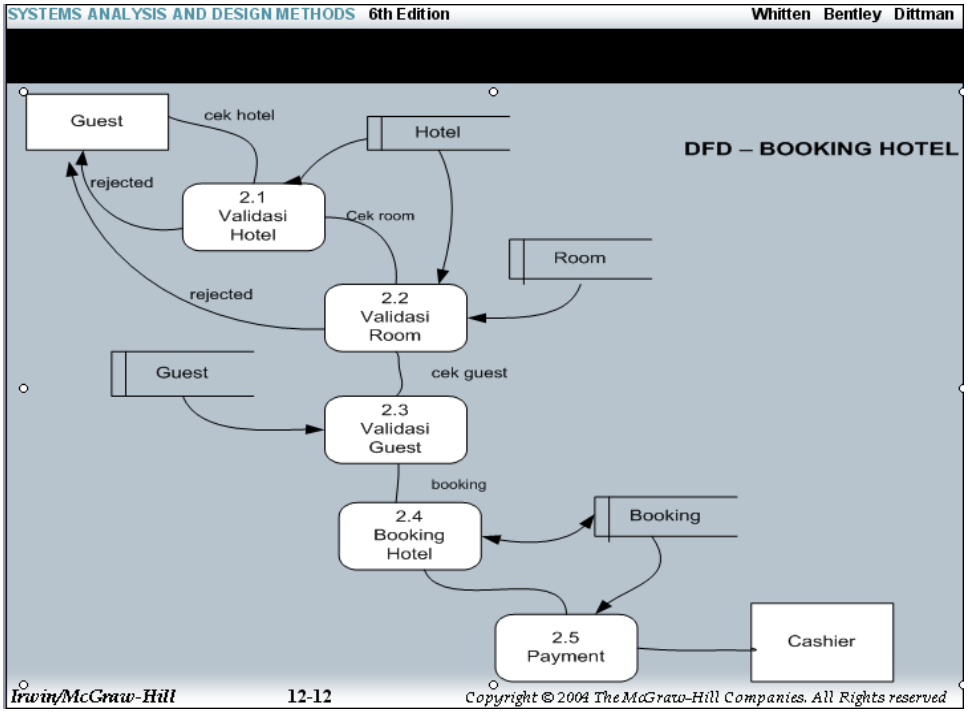
*Data Flow Diagram* (DFD) adalah suatu diagram yang menggunakan notasi-notasi untuk menggambarkan arus dari data sistem, yang penggunaannya sangat membantu untuk memahami sistem secara logika, terstruktur dan jelas.

DFD terdiri dari diagram konteks dan diagram rinci (*DFD Levelled*). Diagram kontek berfungsi memetakan model lingkungan (menggambarkan hubungan antara entitas luar, masukan dan keluaran sistem), yang direpresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem. DFD *levelled* menggambarkan sistem sebagai jaringan kerja antara fungsi yang *berhubungan satu sama lain dengan aliran dan penyimpanan data, model ini hanya memodelkan sistem dari sudut pandang fungsi.*

Simbol – symbol dalam DFD :

| Gane/Sarson | Yourdon/De Marco | Keterangan |
|---|---|---|
| Entitas Eksternal | Entitas Eksternal | Entitas eksternal, dapat berupa orang/unit terkait yang berinteraksi dengan sistem tetapi diluar sistem |
| Proses | Proses | Orang, unit yang mempergunakan atau melakukan transformasi data. Komponen fisik tidak diidentifikasikan. |
| Aliran data | Aliran data | Aliran data dengan arah khusus dari sumber ke tujuan |
| Data store | Data Store | Penyimpanan data atau tempat data direfer oleh proses. |

Gambar 2.16 Simbol dalam DFD




Gambar 2.17 Contoh DFD

## 2.2 Teori – Teori Khusus

### 2.2.1 Definisi *Supply*

Menurut Indrajit (2002, p4), *supply* adalah sejumlah material yang disimpan dan dirawat menurut aturan tertentu dalam tempat persediaan agar selalu dalam keadaan siap pakai dan ditatausahakan dalam buku perusahaan.

### 2.2.2 Definisi Rantai Pasok *(Supply Chain)*

Menurut Pujawan (2005, p5), rantai pasok adalah jaringan perusahaan-perusahaan yang secara bersama-sama bekerja untuk menciptakan dan menghantarkan suatu produk ke tangan pemakai akhir. Pada suatu rantai pasok biasanya ada 3 macam aliran yang harus dikelola yaitu aliran barang, uang dan informasi.

Sedangkan menurut Indrajit (2002, p5), rantai pasok adalah suatu tempat sistem organisasi menyalurkan barang produksi dan jasanya kepada para pelanggannya. Rantai ini juga merupakan jaringan dari berbagai organisasi yang saling berhubungan dan mempunyai tujuan yang sama, yaitu sebaik mungkin menyelenggarakan pengadaan atau penyalur barang tersebut.

Badan usaha yang melaksanakan fungsi suplai pada umumnya terdiri dari manufaktur, penyedia layanan jasa, distributor, dan saluran penjualan (seperti: pedagang eceran, *e-commerce*, dan pelanggan

(pengguna akhir). Aktivitas rantai pasok (rantai nilai dan proses siklus hidup) mengubah bahan baku dan bahan pendukung menjadi sebuah barang jadi yang dapat dikirimkan kepada pelanggan pengguna akhir. Rantai pasok menghubungkan rantai nilai.

### 2.2.3 Definisi Supply Chain Management

Pengertian *Supply Chain Management* menurut Schroeder adalah sebuah proses bisnis dan informasi yang berulang yang menyediakan produk atau layanan dari pemasok melalui proses pembuatan dan pendistribusian kepada konsumen. Sedangkan menurut Indrajit dan Djokopranoto rantai pasok adalah suatu tempat sistem organisasi menyalurkan barang produksi dan jasanya kepada para pelanggannya. Rantai ini juga merupakan jaringan dari berbagai organisasi yang saling berhubungan dan mempunyai tujuan yang sama, yaitu sebaik mungkin menyelenggarakan pengadaan atau penyalur barang tersebut.

### 2.2.4 Keuntungan Menerapkan Supply Chain Management

Keuntungan menerapkan *supply chain* menurut Indrajit dan Djokopranoto adalah:

- Mengurangi inventori barang. Inventori merupakan aset perusahaan yang berkisar antara 30%-40% sedangkan biaya penyimpanan barang berkisar 20%-40% dari nilai barang yang disimpan.

- Menjamin kelancaran arus barang. Rangkaian perjalanan dari bahan baku sampai menjadi barang jadi dan diterima oleh pemakai / pelanggan merupakan suatu mata rantai yang panjang (*chain*) yang perlu dikelola dengan baik.
- Menjamin mutu. Jaminan mutu juga merupakan serangkaian mata rantai panjang yang harus dikelola dengan baik karena mutu barang jadi ditentukan tidak hanya oleh proses produksi tetapi juga oleh mutu bahan mentahnya dan mutu keamanan dalam pengirimannya.

### 2.2.5 Tahapan Mencapai Supply Chain Management

Mencapai *Supply Chain* menurut Miranda dan Tunggal dapat melalui beberapa tahapan antara lain adalah :

Tahap 1 : *Baseline* (Dasar)

Posisi dari kebebasan fungsional yang lengkap di mana masing-masing fungsi bisnis seperti produksi dan pembelian melakukan aktivitas mereka secara sendiri-sendiri dan terpisah dari fungsi bisnis yang lain.

Tahap 2: Integrasi Fungsional

Perusahaan telah menyadari perlu sekurang-kurangnya ada penggabungan antara fungsi-fungsi yang melakukan aktivitas hampir sama, misalnya antara bagian distribusi dan manajemen persediaan atau pembelian dengan pengendalian material.

Tahap 3: Integrasi secara internal

Diperlukan pengadaan dan pelaksanaan perencanaan kerangka kerja *end-to-end.*

Tahap 4: Integrasi secara eksternal

Integrasi *supply chain* yang sebenarnya dengan konsep menghubungkan dan koordinasi yang dicapai pada Tahap 3, yang diperluas dengan bagian *supplier* dan pelanggan.

### 2.2.6 Permasalahan Supply Chain Management

Manajemen rantai pasok harus memasukan *problem* dibawah:

- Distribusi Konfigurasi Jaringan: Jumlah dan lokasi supplier, fasilitas produksi, pusat distribusi ( *distribution centre*/D.C.), gudang dan pelanggan.
- Strategi Distribusi: Sentralisasi atau desentralisasi, pengapalan langsung, Berlabuh silang, strategi menarik atau mendorong, logistik orang ke tiga.
- Informasi: Sistem terintregasi dan proses melalui rantai suplai untuk membagi informasi berharga, termasuk permintaan sinyal, perkiraan, inventaris dan transportasi dsb.
- Manajemen Inventaris: Kuantitas dan lokasi dari inventaris termasuk barang mentah, proses kerja, dan barang jadi.
- Aliran dana: Mengatur syarat pembayaran dan metodologi untuk menukar dana melewati entitas didalam rantai suplai.

- Eksekusi rantai suplai ialah mengatur dan koordinasi pergerakan material, informasi dan dana diantara rantai suplai tersebut. Alurnya sendiri dua arah.

### 2.2.7 Penjualan

Menurut Mulyadi (2001, p202), kegiatan penjualan terdiri dari transaksi penjualan barang dan jasa, baik secara kredit maupun secara tunai. Dalam transaksi penjualan kredit, jika order dari pelanggan telah dipenuhi dengan pengiriman barang atau penyerahan jasa, untuk jangka waktu tertentu, perusahaan memiliki piutang kepada pelanggannya. Dalam sistem penjualan secara tunai, barang atau jasa baru diserahkan oleh perusahaan kepada pembeli jika perusahaan telah menerima kas dari pembeli.

### 2.2.8 Pembelian

Menurut Mulyadi (2001,p299) "pembelian merupakan proses yang digunakan dalam perusahaan untuk pengadaan barang yang diperlukan oleh perusahaan".

Transaksi pembelian dapat digolongkan menjadi 2, yaitu pembelian lokal dan import. Pembelian lokal adalah pembelian dari pemasok dalam negeri, sedangkan pembelian *import* adalah pembelian dari pemasok luar negeri.

Fungsi pembelian bertanggung jawab untuk memperoleh informasi mengenai harga barang, menentukan pemasok yang dipilih dalam pengadaan barang, dan mengeluarkan order pembelian kepada pemasok yang dipilih.

### 2.2.9 Persediaan

Menurut Freddy Rangkuti (1998,p1) "persediaan adalah sebuah aktiva yang meliputi barang-barang milik perusahaan dengan maksud untuk dijual dalam satu periode waktu tertentu atau persediaan-persediaan barang-barang yang masih dalam pengerjaan / proses atau persediaan bahan baku yang menunggu penggunaannya dalam suatu proses produksi".

Dalam perusahaan dagang, persediaan hanya terdiri dari satu golongan, yaitu persediaan barang dagangan, yang merupakan barang yang dibeli untuk tujuan dijual kembali.

Karena pentingnya peranan persediaan barang bagi perusahaan, maka diadakan pengendalian dan pengawasan ketat.

Tujuan pengawasan persediaan :

- Menjaga agar jangan sampai kehabisan persediaan.
- Membentuk persediaan yang stabil.
- Menghindari pembelian yang kecil-kecilan atau tidak efisien
- Pemesanan yang ekonomis.

### 2.2.10 Definisi *Warehouse* (Pergudangan)

*Warehouse* atau pergudangan berfungsi menyimpan barang untuk produksi atau hasil produksi dalam jumlah dan rentang waktu tertentu yang kemudian didistribusikan ke lokasi yang dituju berdasarkan permintaan. Kendala yang dihadapi dalam pengelolaan *warehouse* adalah akurasi pergerakan barang dan menghitung rentang waktu barang disimpan. Dibutuhkan kontrol aktivitas pergerakan barang dan dokumen untuk meningkatkan efisiensi penggunaan *warehouse* agar jumlah dan rentang waktu barang disimpan dalam nilai minimum atau sesuai perencanaan.

### 2.2.11 Definisi *Material* (Bahan)

*Material* adalah sebuah masukan dalam produksi. Mereka seringkali adalah bahan mentah yang belum diproses, tetapi kadang kala telah diproses sebelum digunakan untuk proses produksi lebih lanjut. Umumnya, dalam masyarakat teknologi maju, *material* adalah bahan konsumen yang belum selesai. Beberapa contohnya adalah kertas dan sutra, bahan kadangkala digunakan untuk menunjuk ke pakaian atau kain.

### 2.2.12 Definisi Konsumen

Konsumen adalah seseorang atau sekelompok orang yang membeli suatu produk untuk dipakai sendiri dan tidak untuk dijual kembali. Jika

tujuan pembelian produk tersebut untuk dijual kembali, maka dia disebut pengecer atau *distributor*. Pada masa sekarang ini bukan suatu rahasia lagi bahwa sebenarnya konsumen adalah raja sebenarnya. Oleh karena itu sebagai produsen yang memiliki prinsip *holistic marketing* sudah seharusnya memperhatikan semua yang menjadi hak-hak konsumen.

### 2.2.13 Definisi *Bill of Material (B.O.M)*

BOM adalah suatu kondisi yang mendeskripsikan bahan baku, alat, bagian yang di butuhkan bagi pabrik untuk membuat suatu produk. BOM bisa mendefinisikan sebuah produk yang ingin mereka desain, yang sesuai dengan mereka pesan, yang mereka bangun, dan juga yang mereka rawat.

Untuk bisnis yang berbeda mereka juga menggunakan BOM yang berbeda misalnya untuk perusahaan Elektronik, BOM yang dibuat berdasarkan kebutuhan akan komponen yang digunakan pada papan sirkuit. Sedangkan BOM yang digunakan dalam perusahaan *industry* lebih ke *formula*, *recipe*, atau *ingredients list*.

### 2.2.14 Definisi Purchase Order (PO)

*Purchase Order* adalah sebuah dokumen yang dikeluarkan oleh pembeli dan ditujukan kepada penjual, yang berisikan tipe barang, banyaknya barang, dan juga harga barang tersebut, atau juga pelayanan yang akan diberikan penjual kepada pembeli tersebut.

*Purchase Order* juga menjelaskan syarat pembayaran yang berlaku, kapan tanggal kirim. Kontrak yang dilakukan oleh pembeli dan penjual belum berlaku jika PO belum disetujui.

### 2.2.15 Definisi *Lead Time*

*Lead Time* adalah periode waktu yang dibutuhkan antar proses produksi itu sendiri dengan proses untuk menyelesaikan proses produksi itu sendiri. Di dalam proses industri, *lead time* menjadi bagian yang terpenting dalam penjadwalan produksi itu sendiri.

### 2.2.16 Make to Stock

Menurut Donald (2002) *Make to Stock* adalah karakteristik dari industri - industri dengan skala ekonomi yang besar, hasilnya berasal dari produksi yang dilakukan secara terus menerus. Barang jadinya dipersiapkan untuk kebutuhan pelanggan dimasa datang. Dibutuhkan tempat penyimpanan berupa gudang yang dapat menampung kapasitas dari barang jadi yang sudah diproduksi tersebut dan dimudahkan untuk bermacam - macam barang yang ditujukan untuk pelanggan - pelanggan tertentu.

### 2.2.17 Make to Order

Menurut Donald (2002) *Make to Order* adalah melakukan produksi dengan jumlah yang sesuai pelanggan pesan. Jumlah yang diproduksi merupakan jumlah yang pasti dan relatip dalam jumlah yang kecil. Ruang untuk penyimpanan dibutuhkan untuk jangka waktu yang sementara dan barang jadinya digabungkan untuk mencapai jumlah yang cukup untuk dikirim, tetapi biasanya kebanyakan barang yang diproduksi dikirimkan langsung ke pelanggan.

## 2.3 Tinjauan Pustaka

| No. | Judul, Pengarang dan Tahun | Database | Fungsionalitas | Questioner | Review |
|---|---|---|---|---|---|
| 1. | Analisa dan Perancangan Supply Chain Management Berbasis Web Pada British Petroleum (BP) Indonesia.<br>Rita Citrawati 0500555974<br>Helen 0500557241<br>Ranny Pratiwi 0500558225<br>Tahun 2005 | • PO_HEADER_ALL<br>• PO_LINES_ALL<br>• PER_PEOPLE_F<br>• ROLE<br>• END_USER<br>• PO_LINES_LOCATION_ALL<br>• HR_LOCATION_ALL<br>• INSPECTION_STATUS_CODE<br>• RCV_TRANSACTIONS<br>• MTL_TXN_REQUEST_HEADER<br>• SOURCE_DOCUMENT_CODE<br>• MTL_TRANSACTION_ACTIONS<br>• MTL_MATERIAL_TRANSACTION<br>• UOM<br>• MTL_SYSTEM_ITEM_B<br>• MTL_ITEM_LOCATION<br>• MTL_ONHAND_QUANTITIES | • Informasi dapat menyebarkan dengan cepat, karena sistem yang diusulkan mengintegrasikan bagian2 yang terlibat dalam kegiatan supply chain secara online, sehingga pergerakan material dapat terkontrol dengan baik<br>• Manager dapat langsung mengawasi dan mengontrol pergerakan | Tidak ada kuisioner maupun wawancara | Sistem yang diusulkan hanya berjalan pada kondisi normal, artinya permasalahan untuk barang retur dan barang yang tidak sesuai dengan spesifikasi tidak termasuk dalam ruang lingkup. |

| | | | | | |
|---|---|---|---|---|---|
| | | • MTL_TRANSACTION_TYPES<br>• ARC_MATERIAL_ISSUES<br>• MTL_TRANSACTION_ACTIONS<br>• MTL_TXN_REQUEST_LINES<br>• MTL_TXN_REQUEST_STATUS | material yang ada. manager juga dapat memberikan approval dengan cepat dan melihat informasi dari setiap bagian secara online<br>• Bagian warehouse dapat melakukan pencarian locater barang dengan mudah. sehingga bagian warehouse dapat mengeluarkan barang dengan cepat | | |
| 2. | Analisa dan Perancangan E-SCM pada PT.Lucky Print | • Account<br>• Bahan_baku | • Dapat mengetahui arus pergerakan material, | Tidak ada kuisioner. | • Penamaan penamaan table – |

| | | | | | |
|---|---|---|---|---|---|
| | Abadi<br><br>Husin 0700681680<br><br>Jessie Fascal 0800758883<br><br>JamesKurniawan0800765586<br><br>Tahun 2008 | • bbc<br>• bbk_d<br>• bbk_h<br>• bbs<br>• bpb_h<br>• pelanggan<br>• daerah_kirim<br>• daerah_kirim_d<br>• design<br>• designengrave<br>• designengrave_d<br>• do_d<br>• do_h<br>• fj<br>• gudang<br>• kendaraan | uang dan informasi dengan cepat dan real time. sehingga dapat mendukung dalam pembuatan keputusan.<br>• Dengan adanya perencanaan produksi dan pengendalian bahan baku dalam sistem E-SCM, dapat memenuhi permintaan pelanggan tepat pada waktunya dan sesuai dengan keinginan<br>• Dengan adanya | Melakukan wawancara | tabel yang susah dimengerti fungsi tabelnya<br>• Perlu mengintegrasikan secara penuh antara sistem E-SCM PT.Lucky Print Abadi dengan sistem yang dimiliki pemasok dan pelanggan<br>• Perlu dikembangkannya E-SCM yang |

| | | | | | |
|---|---|---|---|---|---|
| | | • kontrak<br>• kwc<br>• kws<br>• mesin<br>• mesin_bahan_baku<br>• op_d<br>• op_h<br>• perencanaan<br>• perencanaan_finishing<br>• perencanaan_weaving<br>• pesanan_d<br>• pesanan_h<br>• po_d<br>• po_h<br>• pp_d<br>• pp_h | pemantauan bahan baku pada PT.Lucky Print Abadi secara langsung, pemasok dapat merencanakan produksinya menjadi lebih efektif dan efisien<br>• Dengan adanya sistem E-SCM, PT.Lucky Print Abadi menambah satu channel baru dalam berhubungan dengan pemasok dan pelanggannya.<br>• Online katalog yg | | mencakup hubungan dengan pemasok dan pelanggan luar negeri |

| | | | | | |
|---|---|---|---|---|---|
| | | • produk<br>• qcc_d<br>• qcc_h<br>• qcs_h<br>• qcs_d<br>• rencana_kirim_d<br>• rencana_kirim_h<br>• sj<br>• pemasok<br>• pemasok_d<br>• berita | berfungsi untuk menyediakan informasi mengenai rangcangan kain yang ditawarkan.<br>• Online order processing -> pelanggan dapat melakukan pemantauan status pemesanannya | | |

Table 2.1 Tinjauan Pustaka