BAB 2

LANDASAN TEORI

2.1 Teori Umum

2.1.1 Definisi *Database*

Database merupakan data yang saling terhubung dan deskripsi dari data yang dirancang untuk kebutuhan organisasi (Connolly dan Begg, 2004, p15). Menurut McLeod dan Schell (2004, p196) *database system* adalah sistem penyimpanan informasi yang terorganisasi dengan suatu cara sehingga memudahkan untuk proses pengolahan data. Sedangkan menurut Date (2000, p5). *database* adalah *record* yang terkomputerisasi yang bertujuan menyediakan informasi ketika dibutuhkan.

Dari teori-teori tersebut dapat disimpulkan bahwa *database* adalah sejumlah data yang terorganisasi yang saling terhubung untuk menyediakan informasi yang dibutuhkan oleh perusahaan.

2.1.2 DBMS (*Database Management System)*

DBMS adalah sebuah sistem perangkat lunak yang mengizinkan pengguna untuk mendefinisikan, membuat, memelihara, dan mengatur akses ke *database* (Connolly dan Begg, 2004, p16).

Menurut McLeod dan Schell (2004,p196), DBMS adalah aplikasi perangkat lunak yang menyimpan struktur *database*, hubungan antar-data dalam *database*, serta berbagai formulir laporan yang berkaitan dengan *database* tersebut.

Keuntungan DBMS (Connolly dan Begg, 2004, p26) :

1. Kontrol terhadap pengulangan data
2. Konsistensi data
3. Lebih banyak informasi yang didapat dari jumlah data yang sama
4. Data dapat dipakai bersamaan
5. Peningkatan integritas data
6. Peningkatan keamanan
7. Penetapan standardisasi
8. Peningkatan produktivitas
9. Peningkatan layanan *recovery* dan *backup*

Kekurangan DBMS (Connolly dan Begg, 2004, p29):

1. Kompleksitas
2. Ukuran
3. Biaya DBMS
4. Penambahan biaya hardware
5. Biaya konversi
6. Performa
7. Tingkat kegagalan lebih tinggi

2.1.3 Komponen DBMS

Lingkungan DBMS terdiri dari 5 komponen penting (Connolly dan Begg, 2004, p18) :

1. Perangkat keras

   Aplikasi dan DBMS memerlukan perangkat keras untuk berfungsi. Contoh perangkat keras yang digunakan DBMS dan aplikasi antara lain *personal computer, mainframe.*

2. Perangkat lunak

   Komponen dari perangkat lunak terdiri dari DBMS dan program aplikasi bersama dengan sistem operasi. Umumnya, program aplikasi ditulis dalam 3GL (bahasa pemrograman generasi ketiga) seperti java, c++, c, atau menggunakan 4GL (bahasa pemrograman generasi keempat), seperti SQL, yang dikombinasikan dengan 3GL.

3. Data

   Data adalah komponen terpenting dalam lingkungan DBMS. Menurut Connolly dan Begg(2004, p20), data bertindak sebagai penghubung antara komponen mesin dan pengguna. Database berisi data operasional dan metadata (data tentang data).

4. Prosedur

   Prosedur berkaitan dengan instruksi dan aturan yang menentukan rancangan dan penggunaan database.

   Instruksi-instruksi tersebut umumnya berisi :

   a. Instruksi untuk menjalankan dan menghentikan DBMS.
   b. Instruksi untuk menggunakan fasilitas tertentu dari DBMS.
   c. Instruksi untuk membuat salinan *backup* dari *database.*
   d. Instruksi untuk menangani kegagalan perangkat keras atau perangkat lunak.

5. Manusia

Komponen terakhir yaitu manusia yang terlibat dengan sistem.

2.1.4 *Data Warehouse*

Menurut Inmon (2005, p29), "*A data warehouse is a subject oriented, integrated, non volatile and time variancy collection of data in support of management's decision making process*", *data warehouse* adalah koleksi data yang mempunyai sifat berorientasi subjek, terintegrasi, rentang waktu dan tidak mengalami perubahan dari koleksi data dan mendukung proses pengambilan keputusan dalam manajemen.

Menurut Connolly dan Begg (2004, p1151), *data warehouse* adalah koleksi data yang mempunyai sifat berorientasi pada subjek, terintegrasi, memiliki rentang waktu, dan koleksi datanya tidak mengalami perubahan dalam mendukung proses pengambilan keputusan ditingkatan manajerial.

Tujuan utama *datawarehouse* adalah untuk mengintegrasi data yang dimiliki perusahaan ke dalam sebuah *repository* yang akan memudahkan pengguna untuk menjalankan *query,* menghasilkan laporan, dan menampilkan analisa sehingga memudahkan perusahaan dalam mengambil keputusan.

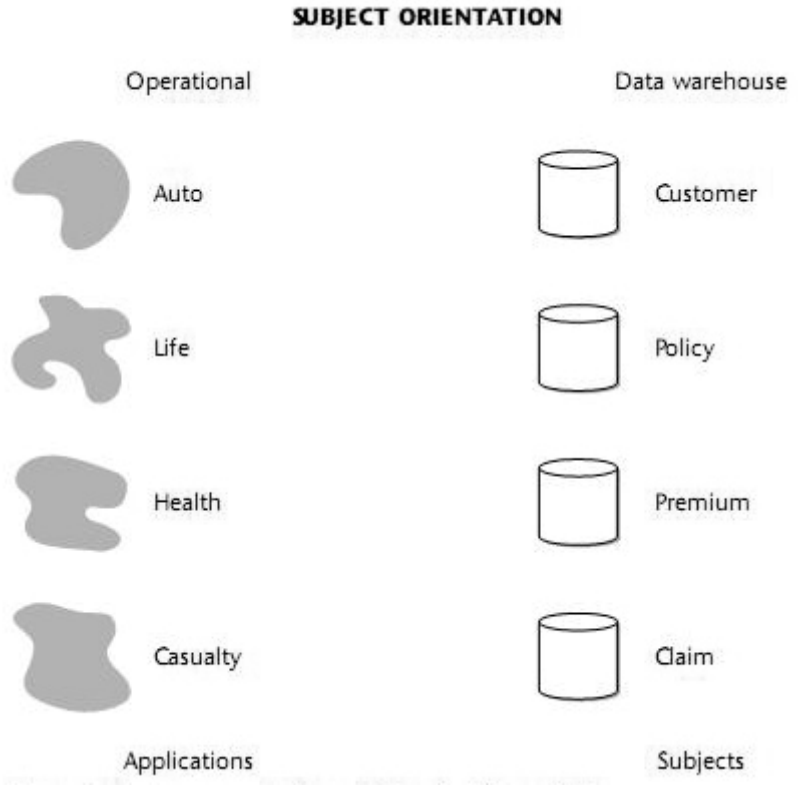
2.1.5 Konsep *Data Warehouse*

Konsep awal *data warehouse* adalah sebagai "*information warehouse"* dan sebagai solusi untuk mengakses data yang terdapat dalam sistem. *Information warehouse* dirancang agar perusahaan mampu menggunakan data yang mereka miliki untuk meningkatkan keuntungan bisnis.

Karakteristik dari *data warehouse* antara lain (Connolly dan Begg, 2004, p1151):

a. *Subject oriented*

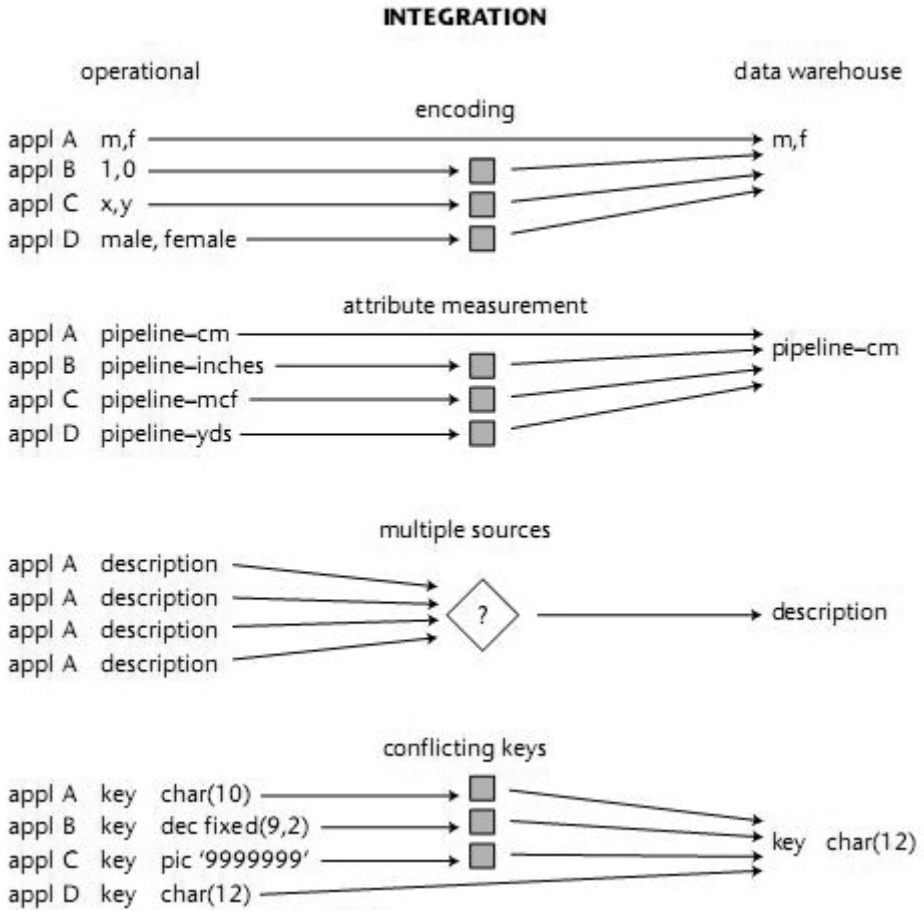
Data warehouse diorganisasikan terhadap subjek utama perusahaan, seperti pelanggan, produk, dan penjualan, bukan terhadap area utama aplikasi, seperti *stock control,* penjualan produk. Hal ini tercermin pada kebutuhan untuk menyimpan *decision-support data* dibandingkan *application-oriented data.*



Gambar 2.1 *Subject Oriented*

b. *Integrated*

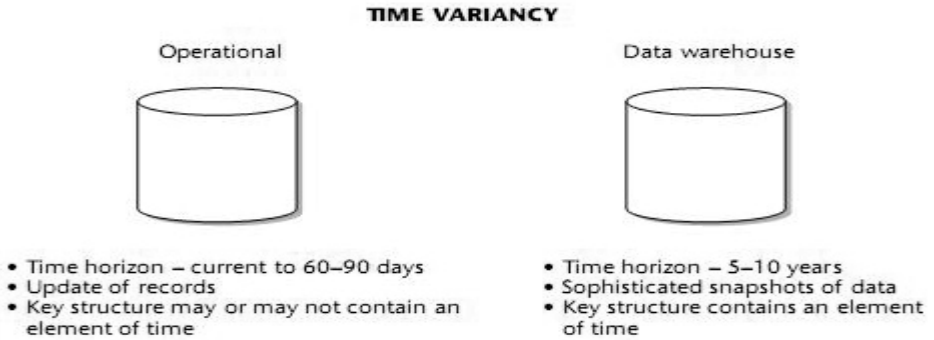
Sumber data berasal dari berbagai sistem aplikasi perusahaan. Sumber data tersebut umumnya tidak konsisten. Sumber data yang terintegrasi haruslah dibuat konsisten agar pengguna dapat menyamakan pandangan terhadap data.



Gambar 2.2 ***Integrated***

c. *Time-variant*

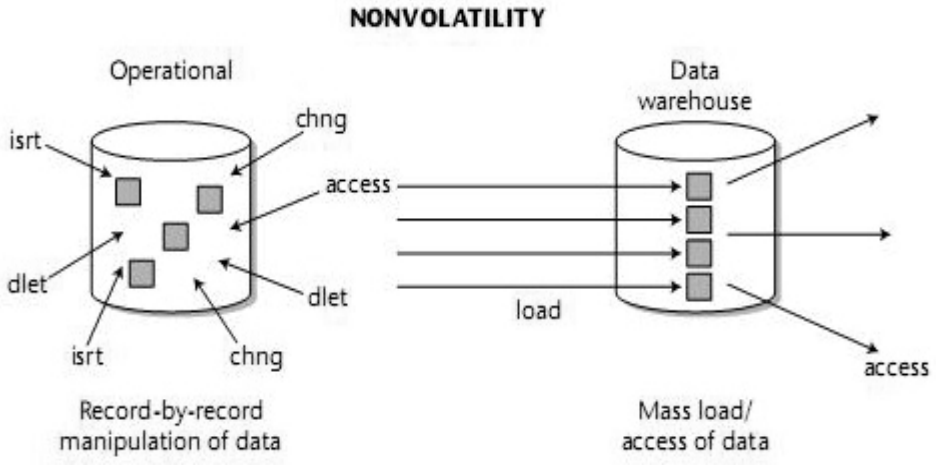
Data dalam *data warehouse* hanya akurat dan valid pada suatu rentang waktu tertentu.



Gambar 2.3 ***Time-variant***

d. *Non-volatile*

Data pada *data warehouse* tidak di-*update* secara *real time*, tetapi di-*refresh* dalam rentang waktu tertentu. Data baru yang ditambahkan ke dalam *data warehouse* adalah sebagai suplemen dan bukan sebagai pengganti.



Gambar 2.4 *Non-volatile*

2.1.6 Arsitektur *Data Warehouse*

Arsitektur dan komponen data warehouse terdiri dari (Connolly dan Begg, 2004, p1156) :

1. *Operational data*

   Sumber data *data warehouse* didapatkan dari :

   a. *Mainframe operational data* terletak di hirarki pertama
   b. *Departemental data* terletak di *propietary file system* seperti VSAM, RMS dan *relational* DBMS seperti Oracle.
   c. *Private data* terletak di *workstation* dan *server* pribadi
   d. Sumber external seperti dari internet dan rekanan bisnis.

2. *Operational data store*

   Operational data store (ODS) merupakan tempat penyimpanan data terintegrasi yang digunakan untuk analisis. Dengan membangun ODS, akan membantu dalam membangun *data warehouse* karena data yang terdapat pada ODS merupakan data yang bersih dan sudah diekstrasi dari sistem sumber.

3. *Load manager*

   Load mananger melakukan semua operasi yang berhubungan dengan ekstrasi dan *me-load* data. Data tersebut mungkin saja diekstrasi secara langsung dari sumber data atau lebih sering dari *operational data store*. Operasi yang dilakukan oleh *load manager* dapat termasuk transformasi sederhana dari data untuk mempersiapkan data tersebut masuk ke dalam *data warehouse*.

4. *Warehouse manager*

   Warehouse manager melakukan semua operasi yang berhubungan dengan manajemen data di dalam *warehouse*. Operasi yang dilakukan *Warehouse manager* antara lain :

   a. Analisis data untuk memastikan konsistensi data.
   b. Transformasi dan penggabungan sumber data dari *temporary storage* ke dalam *warehouse tables*.
   c. Membuat *index* dan *view* pada *base tables*.
   d. Melakukan denormalisasi, jika diperlukan.
   e. Melakukan agregasi, jika diperlukan.
   f. Mem-*back up* dan mengarsipkan data

5. *Query manager*

   Disebut juga komponen *backend* melakukan semua operasi yang berhubungan dengan manajemen *query* pengguna. Operasi yang dilakukan oleh komponen ini termasuk mengarahkan *query* ke tabel yang tepat dan menjadwalkan eksekusi *query* tersebut.

6. *Current Detail Data*

   Data yang aktif saat ini merupakan level terendah dari *data warehouse*, dan biasanya memerlukan tempat penyimpanan (*storage*) yang besar.

7. *Old detail data*

   Data lama berupa hasil *backup* yang disimpan data *storage* yang terpisah yang dapat diakses kembali pada saat dibutuhkan. Penyusunan *direktori* untuk data ini harus mencerminkan umur dari data sehingga memudahkan untuk diakses kembali.

8. *Lightly dan highly summarized data*

   Lightly summarized data merupakan data hasil ringkasan *detail data* namun belum bersifat total *summary.* Akses terhadap data jenis ini banyak digunakan untuk *view* dari kondisi yang tengah berjalan. Sedangkan , *highly summarized data* merupakan hasil *summary* yang bersifat total dan mudah diakses untuk melakukan analisis perbandingan data berdasarkan urutan waktu dan analisis yang menggunakan *database* multidimensi

9. *Archive/backup data*

   Area ini menyimpan data yang telah di*summary* untuk tujuan pengarsipan dan *backup.*

10. *Metadata*

Area *warehouse* ini menyimpan semua definisi *metadata* (data tentang data) yang digunakan oleh semua proses di dalam *warehouse*. *Metadata* digunakan untuk berbagai tujuan termasuk :

a. Proses extraksi dan *loading*. *Metadata* digunakan untuk memetakan sumber data ke sebuah *view* dari data dalam *warehouse*.
b. Proses manajemen *warehouse*. *Metadata* digunakan untuk melakukan otomatisasi terhadap produksi tabel *summary*.
c. Bagian dari proses manajemen *query*. *Metadata* digunakan untuk mengarahkan sebuah *query* ke sumber data yang cocok.

Struktur dari *metadata* dibedakan oleh tiap prosesnya, dikarenakan tujuan dari tiap *metadata* berbeda. Sebagai contoh, *End-user access tools* menggunakan *metadata* untk mengerti bagaiman membangun sebuah *query*.

11. *End-user access tools*

Secara prinsip, tujuan melakukan *data warehouse* adalah untuk menyediakan informasi kepada para pebisnis untuk membuat keputusan yang bersifat strategis. *data warehouse* harus dapat mendukung analisis rutin. Performa tinggi dapat tercapai dengan perencanaan awal untuk melakukan *join*, *summations* dan laporan periodik oleh *end-users*.

Menurut Berson dan Smith (2001, p121), terdapat lima kategori utama dari *End-user access tools* :

a. *Reporting and query tools*

Reporting tools meliputi *production reporting tools* dan *report writers*. *production reporting tools* digunakan untuk menghasilkan laporan

operasional *regular*. Sedangkan, *report writers* merupakan *desktop tools* yang murah dan dirancang untuk *end-users*.

Query tools dirancang untuk menerima SQL atau menghasilkan SQL *statement* untuk melakukan *query* terhadap data yang disimpan dalam *data warehouse.*

b. *Application development tools*

Aplikasi yang digunakan oleh user yang dirancang untuk lingkungan *client-server*.

c. *Executive Information System (EIS) tools*

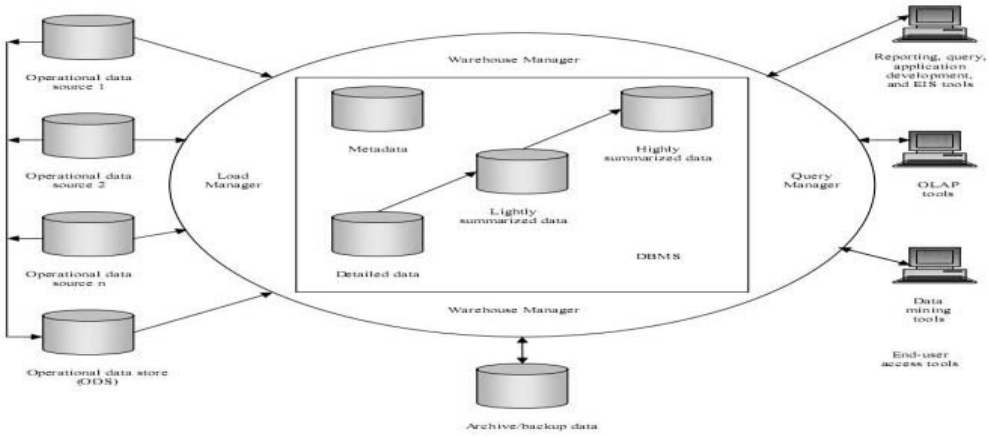
Executive information system pada awalnya dikembangkan untuk mendukung keputusan strategis tingkat tinggi. Namun kemudian fokus dari EIS berkembang menjadi semua tingkat manajemen. EIS *tools* yang terhubung dengan *mainframe* memungkinkan user untuk membuat aplikasi pendukung keputusan berbasis grafik yang bertujuan untuk menyediakan gambaran luas terhadap data perusahaan dan akes ke sumber data external.

d. *Online Analytical Processing (OLAP) tools*

OLAP *tools* didasarkan pada konsep *database* multi-dimensi dan mengizinkan pengguna untuk menganalisis data menggunakan *view* yang kompleks dan multi-dimensi.

e. *Data mining tools*

Data mining adalah proses untuk menemukan hubungan, pola dan tren baru yang bermakna dengan menggali sejumlah data yang besar menggunakan teknik statistik, matematika dan kecerdasan buatan.



Gambar 2.5 Arsitektur *Data Warehouse*

2.1.7 Tahapan Perancangan Data Warehouse

Terdapat sembilan langkah dalam membangun *data warehouse* (Connolly dan Begg, 2004, p1187). Sembilan langkah (*nine-step methodology*) tersebut adalah :

1. *Choosing the process*

   Proses yang mengacu pada pokok *data mart* tertentu. *Data mart* pertama yang akan dibangun harus tepat waktu, sesuai anggaran, dan untuk menjawab sebagian besar pertanyaan bisnis yang penting secara komersial. Sumber data ini mudah diakses dan berkualitas tinggi.

2. *Choosing the grain*

   Memutuskan dengan pasti apa yang menjadi fakta. Dengan memilih *grain*, kita mengidentifikasi fakta tabel dimensi. Pertimbangan *grain* untuk tabel fakta juga menentukan *grain* dari masing – masing dimensi tabel.

3. *Identifying and conforming the dimensions*

   Set dimensi yang dibangun dengan baik akan memudahkan data mart untuk dimengerti dan digunakan. Jika terdapat dimensi yang muncul pada dua data

mart, kedua data mart harus berasal dari dimensi yang sama. Ketika sebuah dimensi digunakan di lebih dari satu data mart disebut *conformed dimension.*

4. *Choosing the facts*

   Grain dari tabel fakta menentukan fakta – fakta yang digunakan dalam *data mart*. Semua fakta harus diekspresikan pada tingkat yang ditunjukkan oleh *grain*. Fakta tambahan dapat ditambahkan ke dalam tabel fakta kapan saja selama grain konsisten dengan tabel.

5. *Storing pre-calculations in the fact table*

   Setelah fakta-fakta terpilih, masing-masing fakta harus dikaji ulang untuk menentukan apakah ada peluang untuk melakukan *pre-calculations*. Sebagai contoh, *pre-calculations* terjadi ketika fakta-fakta terdiri dari keuntungan dan kekurangan. Situasi tersebut sering muncul ketika tabel fakta berbasis penjualan.

6. *Rounding out the dimension tables*

   Dalam langkah ini, kita kembali ke tabel dimensi dan menambahkan sebanyak mungkin keterangan berbentuk teks ke dimensi. Deskripsi teks harus bersifat intuitif dan dapat dimengerti oleh para pengguna sebaik mungkin. Kegunaan *data mart* ditentukan oleh ruang lingkup dan sifat dari atribut yang dimiliki oleh tabel dimensi.

7. *Choosing the duration of the database*

   Durasi mengukur seberapa jauh jangka waktu tabel fakta. Di banyak perusahaan, ada kebutuhan untuk menyimpan data dalam jangka waktu satu atau dua tahun.

8. *Tracking slowly changing dimensions*

   Perubahan dimensi dapat terjadi dengan seiring berjalannya waktu pada tabel dimensi. Perubahan yang dimaksud adalah penambahan data (*insert*) ataupun perubahan data (*update*). Terdapat tiga tipe *Slowly Changing Dimensions* (SCD) yaitu :

   a. Atribut dimensi yang berubah tertulis ulang (*overwritten*).
   b. Atribut dimensi yang berubah menyebabkan sebuah dimensi baru.
   c. Atribut dimensi yang berubah menyebabkan terciptanya atribut alternatif, sehingga *value* baru dan lama dari atribut tersebut dapat diakses secara bersamaan pada dimensi yang sama.

9. *Deciding the query priorities and the query modes*

   Dalam langkah ini perancangan difokuskan pada perancangan secara fisik, yang paling penting diperhatikan dalam desain fisik adalah urutan fisik dari tabel fakta pada disk dan keberadaan *pre-store summaries* atau *pre-store aggregations*. Di samping hal tersebut terdapat, hal lain yang penting diperhatikan adalah administrasi, cadangan, pengindeksian kinerja, dan keamanan.

2.1.8 *Dimensionality Modeling*

Dimensionality modeling adalah suatu teknik perancangan logikal yang bertujuan untuk menampilkan data dalam bentuk yang sesuai standar sehingga dapat diakses dengan kinerja tinggi (Connolly dan Begg, 2004, p1183). *Dimensionality modeling* menggunakan konsep model *entity-relationship* (ER) dengan beberapa perubahan penting. Setiap *dimensional model* (DM) terdiri dari tabel dengan *primary key* yang

disebut dengan *fact table*, dan sekumpulan tabel-tabel yang lebih kecil, disebut *dimension table.* Setiap *dimension table* mempunyai sebuah *primary key* yang terhubung ke salah satu komponen dari *composite key* di dalam *fact table.* Dengan kata lain, *primary key* yang ada pada *fact table* dibuat oleh dua atau lebih *foreign key.* Karakteristik struktur *star-like* ini disebut dengan *star schema* atau *star join.*

Fitur penting DM yang lain yaitu semua *natural key* digantikan oleh *surrogate key.* Setiap *join* antara *fact table* dan *dimension table* didasarkan pada *surrogate key* dan bukan *natural key.* Kegunaan *surrogate key* adalah membolehkan data pada *data warehouse* untuk memiliki kebebasan dalam penggunaan data , tidak seperti yang ada pada sistem OLTP.

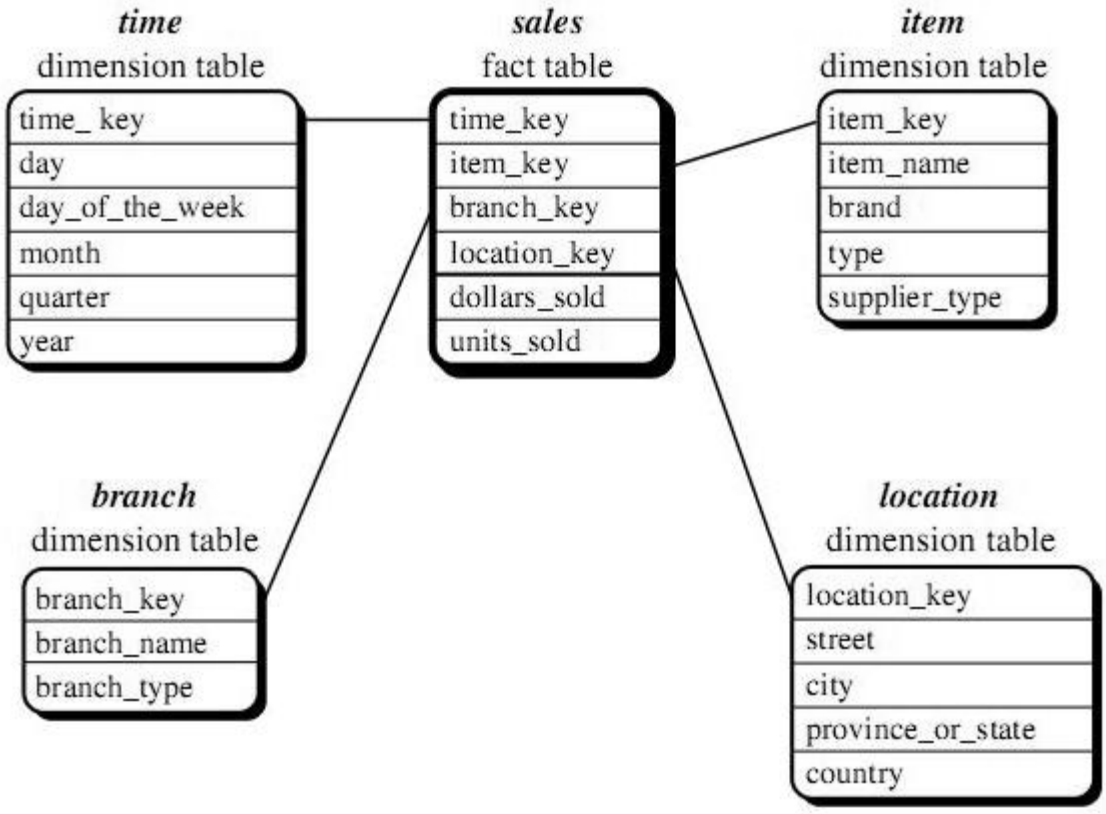
2.1.9 *Star Schema*

Star schema adalah *a logical structure that has a fact table containing factual data in the center, surrounded by dimension tables containing reference data or which can be denormalized.* Struktur logikal fakta yang memiliki tabel fakta di tengah, dikelilingi oleh tabel dimensi yang berisi referensi data (yang dapat di *denormalized*) (Connolly dan Begg, 2004, p1183).

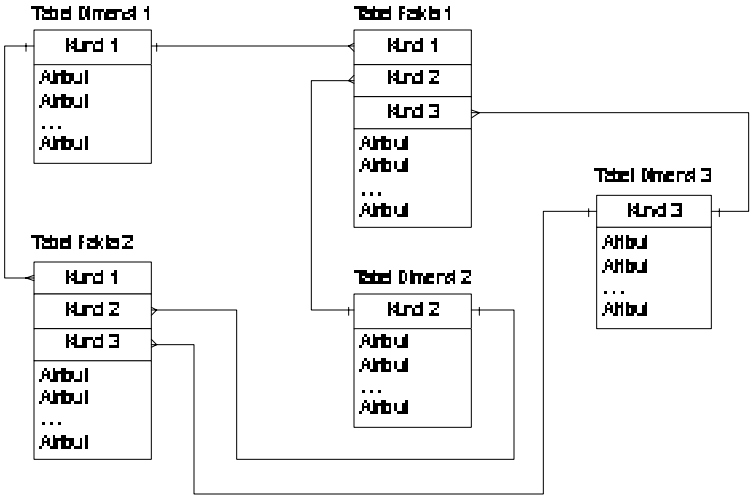
Fakta dalam tabel fakta harus berupa data numerik dan aditif karena aplikasi *data warehouse* hampir tidak pernah mengakses *single record*, melainkan mengakses *record* dalam jumlah ratusan bahkan jutaan dalam sekali waktu dan sangat bermanfaat jika dilakukan agragasi terhadap *record* yang berjumlah banyak tersebut.

Dengan menggunakan skema bintang ini akan menghasilkan waktu respon yang lebih cepat dalam *query* data dibandingkan dengan proses transaksional.

Dalam sebuah skema bintang, bisa terdapat lebih dari satu tabel fakta, karena terdapat fakta yang tidak saling berhubungan. skema semacam ini umumnya digunakan untuk jumlah data yang besar dan untuk berbagai macam tabel data yang teragregasi. Contoh dari skema bintang dapat dilihat pada gambar 2.6 dan gambar 2.7



Gambar 2.6 Skema Bintang (Han dan Kamber)



Gambar 2.7 Skema Bintang Dengan Beberapa Tabel Fakta

(Johan Setiawan)

2.1.10 *Data Mart*

A subject of a data warehouse that supports the requirements of a particular department or business function. *Data mart* adalah sebuah subyek data warehouse yang mendukung kebutuhan departemen atau fungsi bisnis tertentu (Connolly dan Begg, 2004, p1171).

Menurut Inmon (2005, p370) *a data mart is a data structure that is dedicated to serving the analytical needs of one group of people, such as the accounting department or the finance department.* Struktur data yang didedikasikan untuk memenuhi kebutuhan analitis satu kelompok, seperti departemen akuntansi atau departemen keuangan.

Data mart terdiri dari dua tipe yaitu *independent data mart* dan *dependent data mart* (Inmon, 2005, p370-371). *Independent data mart* adalah data mart yang dibangun langsung dari aplikasi turunannya. Sedangkan *dependent data mart* adalah *data mart* yang dibangun dari data yang berasal dari *data warehouse.*

Karakteristik yang membedakan *data mart* dengan *data warehouse* antara lain :

1. *Data mart* hanya berfokus pada kebutuhan user yang berhubungan dengan fungsi bisnis.
2. *Data mart* umumnya tidak berisi data operasional yang detil seperti *data warehouse.*
3. *Data mart* lebih mudah dimengerti dan diarahkan karena berisi lebih sedikit data dibandingkan dengan *data warehouse.*

2.1.11 *Online Analytical Processing*

Online Analytical Processing (OLAP) adalah sintesis dinamis, analisis dan konsolidasi dari sekumpulan besar data multi-dimensi (Connolly dan Begg, 2004, p1205). OLAP memungkinkan pengguna untuk untuk mendapatkan pengertian dan pengetahuan yang mendalam mengenai berbagai aspek dari data perusahaan dengan akses yang cepat, konsisten, interaktif melalui kemungkinan variasi *view* dari data.

Tabel 2.1 Perbedaan OLTP dan OLAP

OLTP	OLAP
Berorientasi pada transaksi	Berorientasi pada analisa
Digunakan oleh DBA	Digunakan oleh para pengambil keputusan
Performa tinggi	Fleksibilitas tinggi
Unit kerjanya transaksi sederhana	Unit kerjanya query kompleks
Data *up-to-date*	Data histori
Rancangan database berorientasi pada	Rancangan database berupa

aplikasi	*star*/*snowflake*
Berfungsi dalam menangani operasi sehari-hari	Berfungsi dalam pengambilan keputusan

Keberhasilan dalam mengimplementasikan OLAP akan membawa keuntungan seperti :

1. Meningkatkan produktifitas para pebisnis, pengembang IT, dan tentunya keseluruhan organisasi.
2. Mengurangi *backlog* dari pengembangan aplikasi untuk staff IT dengan mencukupi kebutuhan *end-user* dalam membuat perubahan skema dan membangun model yang mereka inginkan.
3. Memiliki kendali terhadap integritas data perusahaan seperti aplikasi OLAP bergantung pada *data warehouse* dan sistem OLTP untuk me-*refresh* tingkat sumber data.
4. Mengurangi *query* dan lalu lintas jaringan pada sistem OLTP atau pada *data warehouse*.
5. Meningkatkan nilai potensial dalam hal pendapatan dan keuntungan dengan memungkinkan perusahaan untuk memberikan tanggapan yang lebih cepat kepada permintaan pasar.

2.1.12 *Data Mining*

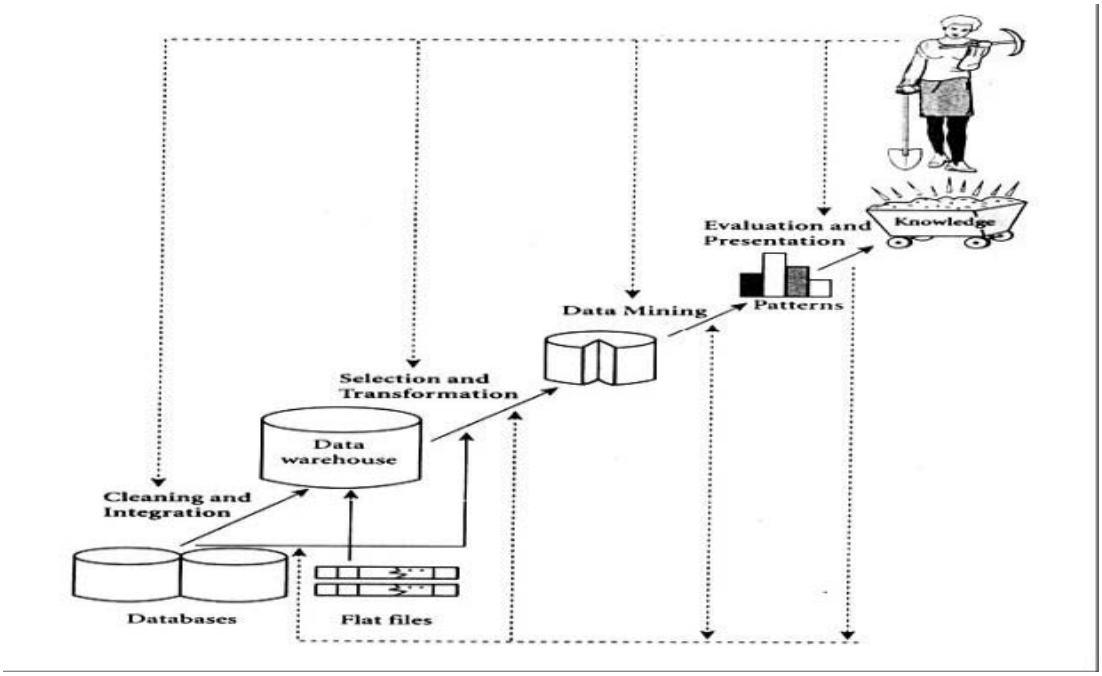
"Data mining adalah proses pengolahan informasi dari sebuah database yang besar, meliputi proses ekstraksi, pengenalan, komprehensif, dan penyajian informasi

sehingga dapat digunakan dalam pengambilan keputusan bisnis yang krusial" (Connolly dan Begg, 2004, p1233).

Menurut Berry dan Linoff (2004, p7)., "Data mining adalah mengeksplorasi dan menganalisis data dalam jumlah besar untuk menemukan pola dan *rule* yang berarti".

Sedangkan menurut Han dan Kamber (2006, p7), "*Data mining* adalah proses menambang (*mining*) pengetahuan dari sekumpulan data yang sangat besar. ". *Data mining* merupakan suatu langkah dalam *knowledge discovery in database (KDD).*

Langkah-langkah dalam menemukan pengetahuan (discovery knowledge) meliputi :



Gambar 2.8 Langkah-Langkah *Discovery Knowledge* (Han dan Kamber)

1. *Data cleaning*

   Menghilangkan *noise* dan data yang tidak konsisten.

2. *Data integration*

   Menggabungkan berbagai sumber data

3. *Data selection*

   Menerima data yang berhubungan dengan analisa dari *database*.

4. *Data transformation*

   Mengubah data ke bentuk yang sesuai untuk *mining* dengan melakukan agregasi atau *summary*.

5. *Data mining*

   Melakukan proses *mining* untuk mengekstrak data.

6. *Pattern evaluation*

   Mengidentifikasi pola yang menggambarkan pengetahuan (*knowledge*).

7. *Knowledge presentation*

   Menampilkan *mined knowledge* kepada pengguna.

Tabel 2.2 Perbedaan *Data Warehouse* dan *Data Mining*.

Data warehouse	***Data mining***
Digunakan oleh pengambil keputusan	Digunakan oleh *business analyst*
Berupa data mentah	Menggunakan data yang disimpan dalam *data warehouse*
Tidak dapat digunakan untuk melakukan prediksi	Dapat digunakan untuk melakukan prediksi
Menyediakan data untuk diekstrak oleh *data mining*	Mengekstrak pengetahuan atau informasi tersembunyi dalam *data warehouse*

2.1.13 Kategori *Data Mining*

Teknik data mining berhubungan dengan penemuan dan pembelajaran, pembelajaran tersebut dapat dibagi menjadi dua metode utama, yaitu *supervised*, dan *unsupervised* (Berson dan Smith, 2001, p416).

1. *Supervised*

   Teknik ini melibatkan tahap pelatihan dimana data lama yang telah dilatih tersebut memiliki *characteristic map* yang telah diketahui dahulu untuk diberikan kepada algoritma *data mining*. Proses ini melatih algoritma untuk mengenali variabel dan nilai-nilai kunci, yang kemudian menjadi dasar untuk membuat prediksi ketika membaca data baru.

2. *Unsupervised*

   Teknik ini tidak dapat melibatkan tahap pelatihan, tetapi bergantung pada penggunaan algoritma yang mendeteksi semua bentuk seperti asosiasi dan rangkaian yang terjadi berdasarkan kriteria yang spesifik dalam memasukkan data. Pendekatan ini membawa ke generasi yang mempunyai banyak peraturan yang menggolongkan penemuan asosiasi, *cluster*, dan *segment*. Peraturan ini kemudian akan menganalisa untuk menentukan mana yang memiliki ketertarikan secara universal.

2.1.14 Teknik *Data Mining*

1. Teknik *decision tree*

Decision tree adalah suatu model prediksi yang dilihat sebagai *tree*. Setiap cabang merupakan hasil klasifikasi dari pertanyaan dan daun merupakan hasil partisi dari kumpulan data sesuai dengan klasifikasinya. Implementasinya dalam dunia

bisnis, *decision tree* digunakan untuk membuat pengelompokan (segmentasi) data sumber. Algoritma *decision tree* berhenti mengembangkan segmen atau cabangnya ketika semua segmen terdiri dari hanya satu *record* atau ketika sejumlah *record* dalam segmen memiliki karakteristik yang sama, pengelompokan data telah tersusun dengan baik dengan satu nilai prediksi, pengembangan data perusahaan tidak memenuhi syarat untuk melakukan pembagian lebih lanjut (Berson dan Smith, 2001, p351-352).

2. Teknik *clustering*

Clustering adalah metode mengelompokkan *record* ke dalam grup-grup berdasarkan kesamaan karakteristik yang dimiliki oleh *record-record* tersebut. Teknik ini digunakan untuk *high level end user* untuk melihat apa yang terjadi di *database* dan untuk menyederhanakan tampilan data yang ada dalam *database*. Teknik ini digolongkan dalam *unsupervised learning* (Berson dan Smith, 2001, p416). *Clustering* juga berarti segmentasi yang sangat berguna dalam prediksi dan analisa masalah bisnis (Berson dan Smith, 2001, p407).

3. Teknik *nearest neighbor*

Merupakan salah satu teknik tertua dalam *data mining*. *Nearest neighbor* memiliki kemiripan dengan *clustering*, yang digunakan dalam memprediksi nilai dari sebuah *record*, yaitu pengguna teknik tersebut harus mencari *record* lain dengan jenis dan nilai yang mirip dan menggunakan nilai prediksi dari *record* yang terdekat tersebut sebagai acuan terhadap *record* yang belum terklasifikasi (Berson dan Smith, 2001, p407). Teknik ini tergolong kategori *supervised learning* (Berson dan Smith, 2001, p416).

4. Teknik *neural network*

Neural network adalah jaringan sistem saraf dalam otak manusia yang mengenali pola, membuat perkiraan dan pembelajaran (Berson dan Smith, 2001, p375). Sedangkan yang dimaksudkan disini adalah "*artificial neural network*" yaitu program komputer yang menerapkan pencarian pola dan algoritma pembelajaran bagi mesin komputer untuk membuat model perkiraan dari data historikal yang besar. *Artificial neural network,* merupakan "otak" tiruan dari otak manusia yang dibuat sedemikian rupa agar dapat menirukan cara kerja otak manusia dalam menyelesaikan masalah.

5. Teknik *rule induction*

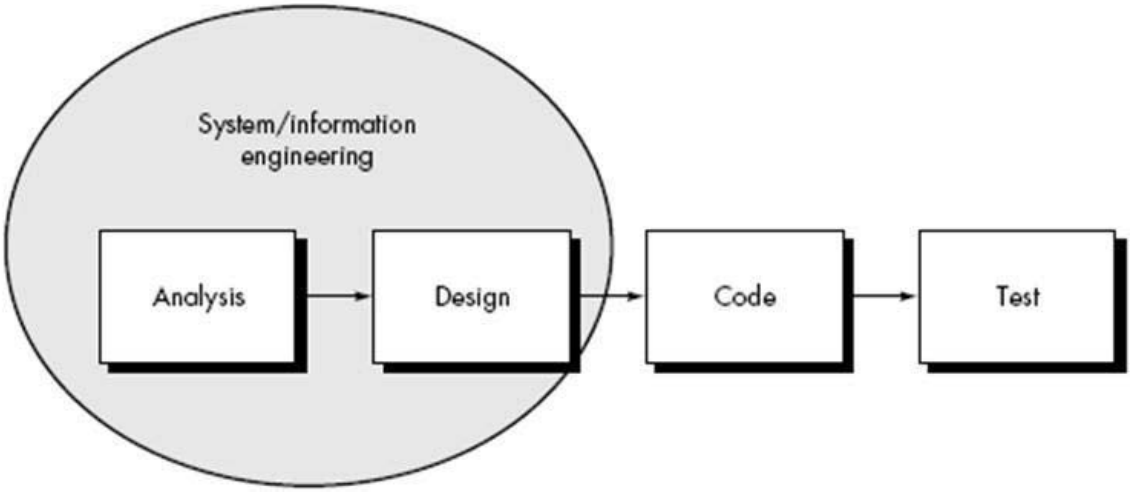
Rule induction merupakan salah satu teknik utama dalam *data mining* yang paling banyak digunakan dalam menemukan pengetahuan dalam sistem *supervised learning*. Agar aturan tersebut bermanfaat maka harus ditambahkan informasi tambahan sesuai dengan keadaan sebenarnya, yaitu : keakuratan yang menunjukkan seberapa besar persentase aturan tersebut benar.

2.1.15 Definisi *Software Engineering*

Software engineering adalah penetapan dan penggunaan prinsip-prinsip teknik untuk mendapatkan perangkat lunak yang handal dan bekerja secara efisien pada mesin (Pressman, 2001, p20). *Software engineering* berhubungan dengan teori, metode dan alat yang dibutuhkan untuk mengembangkan perangkat lunak bagi komputer (Sommerville,2011, p4). *Software engineering* berbeda dengan pemrograman tradisional karena *Software engineering* berhubungan dengan masalah-masalah yang terdapat dalam manajerial.

2.1.16 Model *Waterfall*

Model ini melakukan pendekatan sistematis dan berurutan pada pengembangan perangkat lunak yang dimulai pada level sistem dan perkembangan melalui analisis, desain, koding, testing dan *support* (Pressman, 2001, p28).



Gambar 2.9 Model *Waterfall* (Pressman)

2.1.17 Teori Interaksi Manusia dan Komputer

Interaksi manusia dan komputer adalah ilmu yang mempelajari mengenai bagaimana mendesain, mengevaluasi, dan mengimplementasikan sistem komputer yang interaktif sehingga dapat dipahmi dan digunakan oleh manusia dengan mudah. Interaksi yang dipelajari disini adalah komunikasi dua arah antara manusia (*user*) dan sistem komputer.

Berikut adalah pedoman dalam mendesain suatu aplikasi oleh shneiderman (2005, p74) yang dikenal dengan *eight golden rules*, yaitu :

1. *Strive for consistency* (konsistensi)

Konsisten pada urutan tindakan, perintah, dan istilah yang digunakan pada prompt, menu.

2. *Enable frequent users to use shortcuts* (memungkinkan pengguna untuk menggunakan shortcut)

   Memenuhi kebutuhan pengguna yang sudah ahli untuk meningkatkan efisiensi pada interaksi sehingga diperlukan tombol fungsi, perintah tersembunyi.

3. *Offer informative feedback* (memberikan umpan balik yang informatif)

   Setiap langkah pengguna diberikan umpan balik. Untuk langkah yang sering dilakukan dan tidak penting, dapat diberikan umpan balik yang sederhana. Namun ketika pengguna melakukan langkah yang jarang dilakukan maka umpan balik yang diberikan harus lebih penting.

4. *Design dialog to yield closure* (merancang dialog untuk menghasilkan suatu penutupan)

   Urutan langkah diorganisasi ke dalam kelompok : permulaan, pertengahan, akhir. Umpan balik yang informatif akan memberikan indikasi bahwa langkah yang dilakukan sudah benar.

5. *Offer simple error handling* (memberikan penanganan kesalahan yang mudah)

   Sistem dirancang agar pengguna tidak dapat melakukan kesalahan fatal. Jika terjadi kesalahan, sistem dapat mendeteksi kesalahan dan memberikan mekanisme dan pesan sederhana yang dapat dipahami pengguna.

6. *Permit easy reversal of actions* (membolehkan pengguna kembali ke langkah sebelumnya)

Mengurangi kekhawatiran pengguna ketika melakukan kesalahan, sehingga pengguna dapat lebih mengerti langkah yang tidak biasa digunakan.

7. *Support internal locus of control* (mendukung tempat pengendali internal)
   Sistem dirancang sedemikian rupa sehingga pengguna menjadi inisiator daripada responden.
8. *Reduce short-term memory load* (mengurangi beban ingatan jangka pendek)
   Karena keterbatasan ingatan manusia maka dibutuhkan tampilan yang sederhana atau banyak tampilan halaman yang perlu disatukan, serta diberikan cukup waktu pelatihan untuk kode, mnemonic dan urutan langkah.

2.1.18 Aplikasi Berbasis *Web*

Aplikasi web adalah aplikasi *client-server* yang umumnya menggunakan *web browser* sebagai *client* (Shklar dan Rosen, 2003, p202). Sedangkan menurut Kappel et al (2006, p2), aplikasi *web* adalah sistem perangkat lunak berbasis teknologi dan standar dari *World Wide Web Consortium* (W3C) yang menyediakan spesifik *resource* seperti konten dan layanan melalui antarmuka *web browser*. Cara kerja umum aplikasi *web* dimulai dengan *browser* mengirimkan *request* ke *server* dan kemudian *server* mengirimkan respon melalui *browser*.

Dalam aplikasi web dikenal tiga teknologi, yaitu *client-side* teknologi, *document-spesific* teknologi, *server-side* teknologi (Kappel et al, 2006, p116-26). Java Applets adalah contoh dari *client-side* teknologi. Sedangkan Contoh *document-spesific* teknologi adalah HTML. HTML adalah bahasa format dokumen yang digunakan untuk merancang halaman *web* (Connolly dan Begg, 2004, p1001). Dalam menuliskan kode

perintah HTML dapat menggunakan notepad, Edit Plus ataupun editor lainnya yang berbasis GUI (*Graphical User Interface*) seperti Microsoft Front Page, Dreamweaver CS3, Adobe Golive dan sebagainya. Contoh *server-side* teknologi adalah *server side scripting*. *server side scripting* dikerjakan oleh *web* server seperti Personal *Web* Server untuk sistem operasi Windows 98, IIS untuk sistem operasi Windows 2000/XP, Apache, Tomcat, Xitami dan Zope. Sedangkan untuk contoh bahasa *server side* yaitu ASP (.Net), PHP, Cold Fusion (Kappel et al, 2006, p127).

2.1.19 XML

XML adalah sebuah *meta-language* (bahasa yang mendeskripsikan bahasa lain) yang membolehkan perancang untuk menciptakan tag buatan mereka sendiri untuk menyediakan fungsionalitas yang tidak terdapat pada HTML (Connolly dan Begg, 2004, p1073). XML mendukung *link-link* yang menunjuk pada banyak dokumen, sedangkan *link* HTML hanya merujuk pada satu dokumen tujuan (Connolly dan Begg, 2004, p1073).

Kelebihan XML diantaranya (Connolly dan Begg, 2004, p1074) :

1. Sederhana.
2. *Open standard*.
3. Dapat digunakan kembali.
4. Meningkatkan *load balancing*
5. Mendukung integrasi data dari banyak sumber.

2.1.20 *Web Server*

Web server adalah sebuah perangkat lunak yang menerima permintaan dari komputer pengguna melalui *web browser* (Ambegaonkar, 1997, p31). Web server bertugas menjadi *host* sebuah website Internet atau Intranet (Whitten et al, 2004, p486). *Web server* berkomunikasi dengan *fat client* dan *thin client* dengan memberikan kepada *client* tersebut dokumen-dokumen (dalam format-format seperti HTML) dan data (pada format-format seperti XML) (Whitten et al, 2004, p486). *Fat client* adalah sebuah komputer personal, komputer notebook, atau workstation yang umumnya lebh powerful (dan mahal) dalam hal kecepatan prosesor, memori, dan kapasitas penyimpanan, sedangkan *thin client* adalah komputer personal yang tidak harus powerful (atau mahal) dalam hal kecepatan prosesor dan memori karena ia hanya menyajikan antarmuka bagi pengguna (Whitten et al, 2004, p486). Perangkat lunak *web server* yang banyak digunakan antara lain apache, IIS, nginx, GWS, lighttpd.

2.1.21 *Web Browser*

Web browser adalah sebuah aplikasi perangkat lunak yang digunakan oleh pengguna untuk mengakses *web* (Turban et al, 2006, p681).

Keuntungan menggunakan *web browser* diantaranya (Blackmore, 2001, p7-8) :

1. *Cross platform*
2. Akses global
3. Kemudahan penggunaan
4. Fleksibel
5. Standard terbuka
6. Biaya rendah

2.1.22 *Intranet*

Intranet adalah jaringan privat yang menggunakan peranti lunak Internet dan protokol TCP/IP (Turban et al, 2006, p683). Sedangkan menurut Connolly dan Begg (2004, p996), Intranet adalah sebuah situs *web* atau kumpulan situs yang dimiliki oleh sebuah organisasi yang hanya dapat diakses oleh anggota organisasi. Intranet dibangun dengan konsep dan teknologi yang sama dengan internet, seperti *client-server* dan *internet protocol suite* (TCP/IP). HTTP, FTP, SMTP adalah beberapa protokol yang sering digunakan dalam intranet. Aplikasi yang paling umum pada intranet perusahaan diantaranya untuk gudang data dan akeses pendukung keputusan, basis data pelanggan, katalog produk, manual kerja, bagan organisasi, program pelatihan, mesin pencari, kebijakan, prosedur dan sumber daya manusia, direktori telepon organisasi, berita terbaru mengenai organisasi (Turban et al, 2006, p683).

2.1.23 TCP/IP

TCP/IP adalah protokol transfer *file* yang dapat mengirimkan berbagai *file* informasi berukuran besar melalui jaringan yang kadang kurang andal, dengan jaminan data tersebut akan masuk dalam bentuk yang tidak rusak (*uncorrupted*) (Turban et al, 2006, p662). TCP/IP merupakan protokol standar komunikasi untuk Internet. TCP bertanggung jawab dalam memverifikasi kebenaran data yang dikirim dari *client* ke *server*, sedangkan IP menyediakan mekanisme *routing* (Connolly dan Begg, 2004, p701).

TCP/IP mengimplementasikan arsitektur berlapis yang terdiri dari empat *layer*, diantaranya :

1. *Application layer*

   Layer ini menetapkan protokol-protokol untuk komunikasi antar aplikasi (Lammle, 2007, p69).

2. *Transport layer*

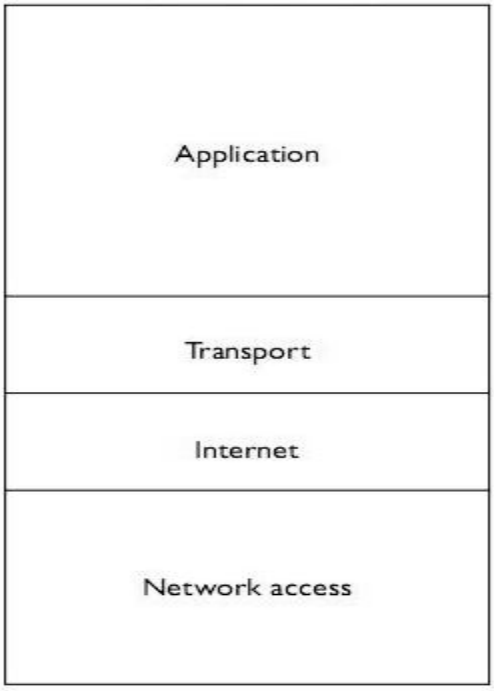
   Layer ini menangani *packet sequincing* dan menjaga integritas data dengan cara memastikan pengiriman data bebas *error* (Lammle, 2007, p69).

3. *Internet layer*

   Layer ini merancang protokol yang berhubungan dengan pengiriman paket secara *logical* ke seluruh jaringan (Lammle, 2007, p69).

4. *Network access layer*

   Layer ini meliputi *addressing hardware* dan menetapkan protokol untuk transmisi data secara *physical* (Lammle, 2007, p69).



Gambar 2.10 Arsitektur TCP/IP

Berikut ini adalah layanan yang berjalan di protokol TCP/IP :

1. Telnet

   Protokol yang digunakan untuk mengakses *resource* lain dengan menggunakan komputer dalam jarak jauh (Lammle, 2007, p72).

2. FTP

   Protokol yang digunakan untuk melakukan transfer *file* (Lammle, 2007, p72).

3. NFS

   Protokol yang digunakan untuk melakukan *sharing file* (Lammle, 2007, p72).

4. SMTP

   Protokol yang digunakan dalam pengiriman *e-mail* di Internet (Lammle, 2007, p72).

5. DNS

DNS adalah sistem yang menyimpan informasi mengenai *hostnames* (Lammle, 2007, p73).

2.1.24 HTTP

HTTP adalah protokol yang digunakan untuk melakukan transfer halaman *web* melalui internet (Connolly dan Begg, 2004, p999). Sebuah Transaksi HTTP terdiri dari beberapa langkah berikut ini (Connolly dan Begg, 2004, p999) :

- *Connection* : *client* membuat koneksi dengan *web server*.
- *Request : client* mengirimkan pesan *request* ke *web server*.
- *Response : web server* mengirimkan respon (umumnya berupa dokumen HTML) ke *client.*
- *Close :* koneksi ditutup oleh *web server*.

Protokol HTTP bersifat *stateless*, yang artinya *server* tidak menyimpan informasi antar tiap *request* (Connolly dan Begg, 2004, p1000).

2.1.25 MySQL

MySQL adalah perangkat lunak DBMS. MySQL

merupakan perangkat lunak dibawah lisensi GNU General Public License (GPL), tetapi juga tersedia versi komersial MySQL. Banyak proyek perangkat lunak gratis yang menggunakan MySQL sebagai DBMS, seperti Joomla, Wordpress, Drupal. Beberapa web skala besar seperti Facebook, Flickr, Wikipedia menggunakan MySQL.

Gambar 2.11 Logo MySQL

MySQL menyediakan modul berbasis GUI untuk melakukan administrasi diantaranya MySQL administrator, MySQL Query Browser atau phpMyadmin yang berbasis web. MySQL mendukung banyak bahasa pemrograman seperti Java, PHP, C, C++, C#, Phyton, Ruby.

Kelebihan MySQL antara lain (Pachev, 2003, p4) :

1. Kecepatan
2. Multi *platform*
3. *Requirement hardware* rendah
4. *Reliability*
5. *Scalability*
6. Mendukung ODBC
7. Tersedianya *source code* yang dapat diunduh secara gratis

Kekurangannya antara lain (Pachev, 2003, p4):

1. Kurangnya fitur-fitur SQL
2. Kurangnya percobaan pada *platform* tertentu

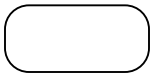
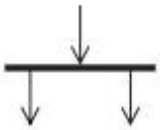
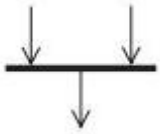
3. Sulit dalam bekerja dengan *source code* MySQL.

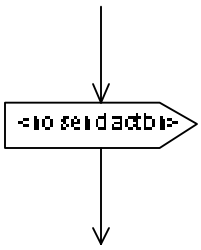
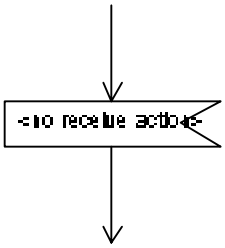
2.1.26 *Activity* Diagram

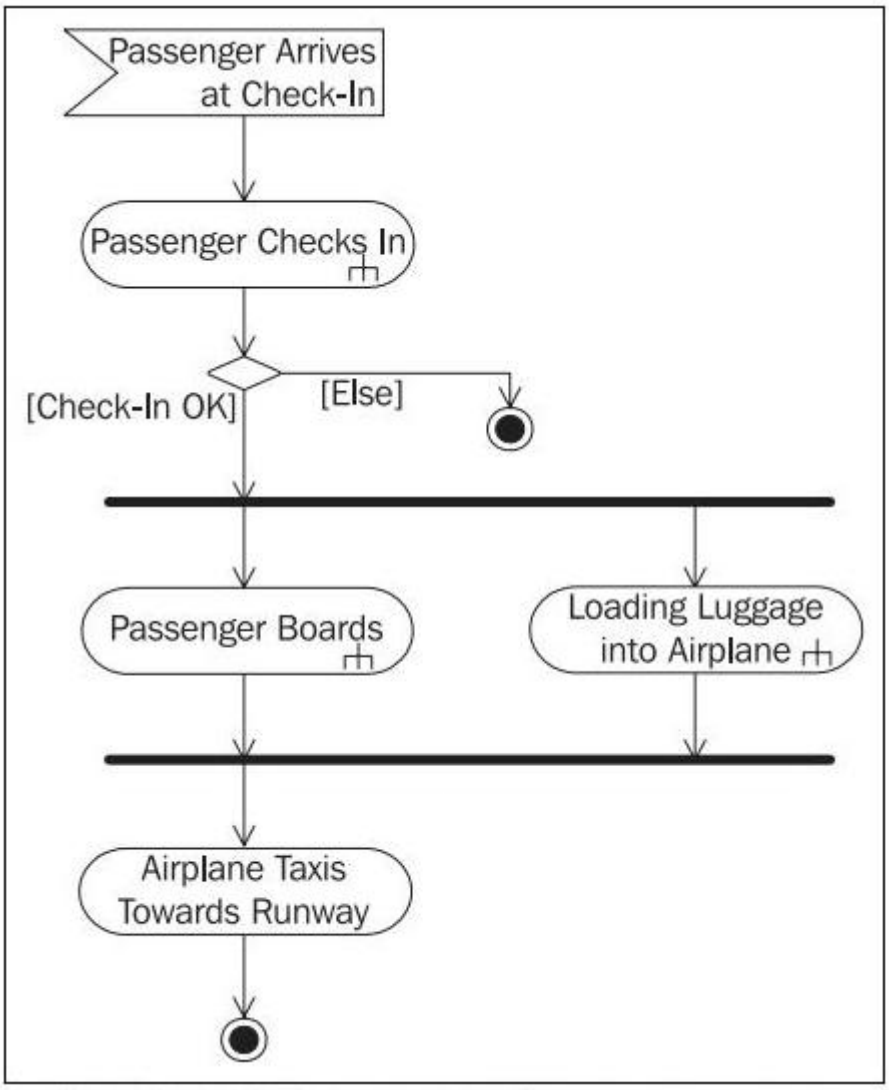
Activity diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus (Munawar, 2005, p109). *Activity* diagram memiliki peran seperti halnya *flowchart*, akan tetapi *flowchart* tidak mendukung perilaku paralel.

Berikut adalah simbol-simbol yang digunakan pada *activity* diagram

Tabel 2.3 simbol *activity* diagram

Simbol	Keterangan
	Titik awal
	Titik akhir
	*Activity*
	Pilihan
	Fork; digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau digunakan untuk menggabungkan dua kegiatan paralel menjadi satu
	Rake; menunjukkan adanya dekomposisi
	Tanda waktu

	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*flow final*)

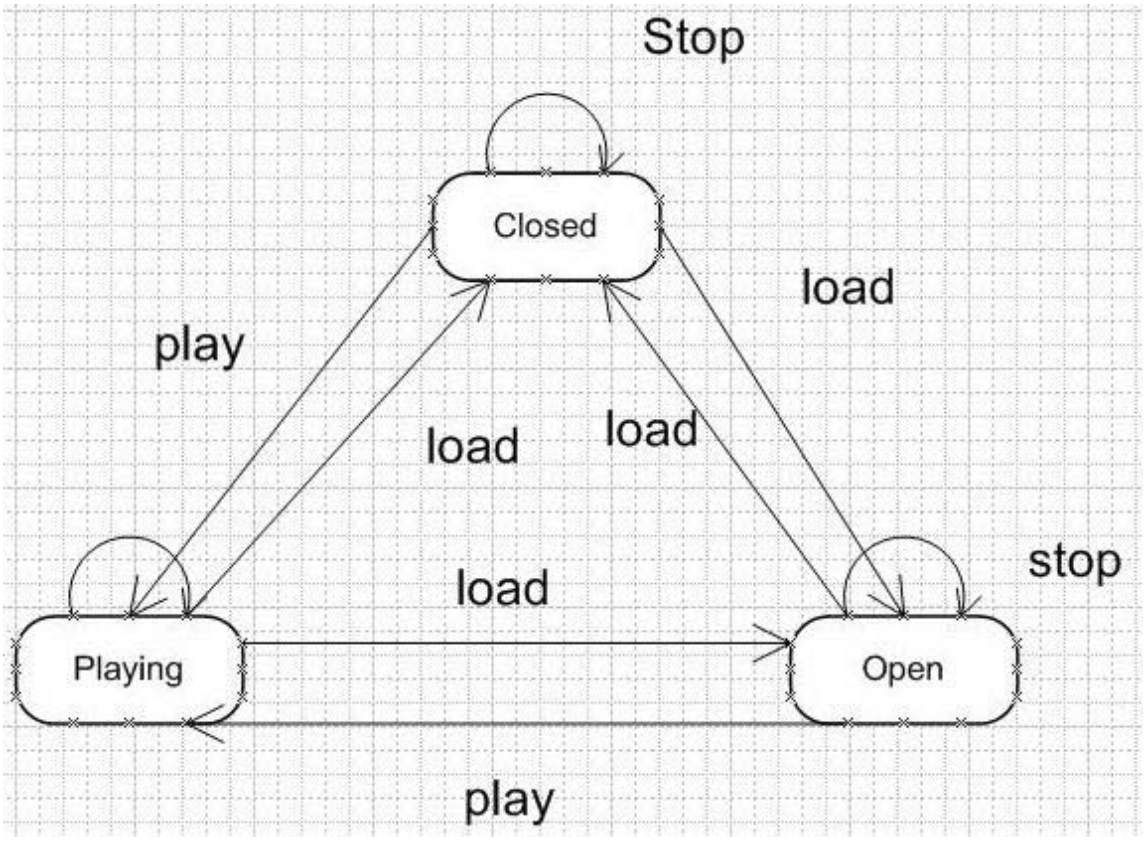


Gambar 2.12 Contoh *Activity* Diagram (Baumann et al)

2.1.27 *State Transition Diagram*

Diagram ini digunakan untuk membantu analisis, perancang dan pengembang untuk memahami perilaku objek di sistem(Munawar, 2005, p73). Harel *statechart* adalah variasi STD yang paling terkenal, yang kemudian diadaptasi oleh Booch ke dalam Unified Modeling Language (UML). *Statechart* diagram menampilkan state-state yang

mungkin dari sebuah objek, event yang bisa dideteksi dan respon atas event-event tersebut (Munawar, 2005, p75).



Gambar 2.13 Contoh *State Transition* Diagram (Munawar)

2.2 Teori Khusus

2.2.1 Penjualan

"Kegiatan penjualan terdiri dari transaksi penjualan barang atau jasa, baik secara kredit maupun tunai" (Mulyadi, 2001, p202). Kegiatan penjualan terdiri dari transaksi penjualan barang dan jasa baik secara kredit maupun tunai. Penjualan adalah peningkatan jumlah aktiva atau penurunan kewajiban suatu badan usaha yaitu : timbul dari penyerahan barang, jasa atau aktiva usaha lainnya di dalam suatu periode.

Penjualan tunai dilakukan oleh perusahaan dengan cara mewajibkan pembeli melakukan pembayaran harga terlebih dahulu sebelum barang diserahkan oleh perusahaan kepada pembeli. Setelah uang diterima oleh perusahaan, barang kemudian diserahkan kepada pembeli dan transaksi penjualan tunai kemudian dicatat oleh perusahaan (Mulyadi, 2001, p202). Sedangkan penjualan kredit adalah penjualan yang pembayarannya dilakukan beberapa waktu kemudian setelah pembeli menerima barang yang dipesannya. Pembayaran biasanya dilakukan dalam jangka waktu yang telah disepakati oleh kedua belah pihak. Penjualan kredit dilaksanakan oleh perusahaan dengan cara mengirimkan barang sesuai dengan pesanan yang diterima dari pembeli dan untuk jangka waktu tertentu perusahaan menagih kepada pembeli tersebut (Mulyadi, 2001, p203).

2.2.2 Pemasaran

Pemasaran adalah suatu proses sosial dan manajerial yang membuat individu dan kelompok memperoleh yang mereka butuhkan dan inginkan melalui penciptaan dan pertukaran timbal balik produk dan nilai dengan orang lain (Kotler, 2001, p6). Tujuan pemasaran adalah untuk mengetahui dan memahami para pelanggan dengan baik sehingga produk atau jasa yang dihasilkan perusahaan cocok dengan mereka dan dapat terjual dengan sendirinya. Kotler (2001, p18), konsep pemasaran mengatakan bahwa, untuk mencapai tujuan organisasi tergantung pada penentuan kebutuhan dan keinginan pasar sasaran dan memuaskan pelanggan secara lebih efektif daripada yang dilakukan oleh pesaing. Konsep pemasaran memusatkan perhatian pada kebutuhan pembeli dan bagaimana caranya untuk memuaskan kebutuhan pelanggan melalui produk dan segala

sesuatu yang berkaitan dengan penciptaan, pengiriman dan pengkonsumsian produk tersebut.

2.2.3 Aturan Asosiasi

Aturan asosiasi (*association rule*) adalah metode data mining untuk mencari suatu hubungan yang menunjukkan kondisi di dalam satu set data, yang beberapa nilai atribut akan muncul secara bersamaan (Han dan Kamber, 2006, p23). Metode ini cukup dikenal dengan istilah *market basket analysis*.

Bentuk umum aturan (*rule*) : "A → B"

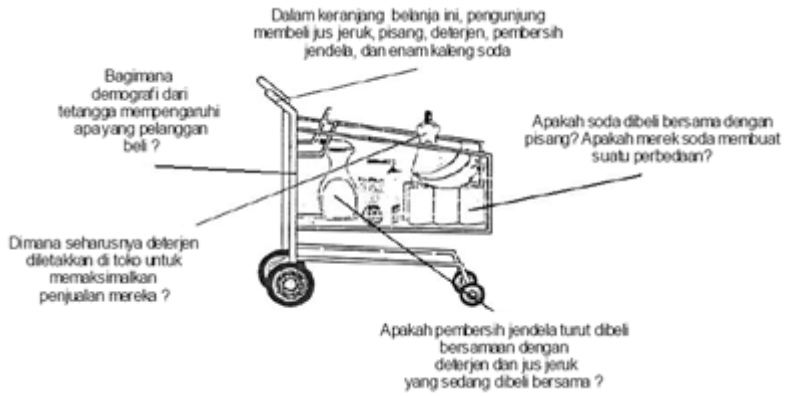
Penjelasan : A = *antecendent*

B = *consequent*

Contoh aturan dalam penjualan :

a. Jika x membeli roti tawar, maka x akan membeli selai
b. Jika x membeli popok, maka x akan membeli bir

Gambar 2.16 menggambarkan permasalahan yang dapat dijawab aturan asosiasi



Gambar 2.14 aturan Asosiasi

Aturan asosiasi didefinisikan suatu proses untuk menemukan semua aturan asosiasi yang memenuhi syarat minimum untuk *support* (*minimum support*) dan syarat minimum untuk *confidence* (*minimum confidence*).

Metodologi dasar analisis asosiasi terbagi menjadi dua tahap (Kusrini dan Luthfi, 2009, p150):

1. Analisis pola frekuensi tinggi

   Mencari *item* yang memenuhi syarat minimum dari nilai *support* dalam database. Rumus untuk memperoleh nilai support :

   Support (A) = ∑ transaksi berisi A / ∑ transaksi

2. Pembentukan aturan asosiasi

   Setelah semua pola frekuensi tinggi didapatkan, kemudian dicari aturan asosiasi yang memenuhi syarat minimum untuk *confidence* dengan menghitung *confidence*.

   Nilai confidence dari aturan A → B diperoleh dari rumus berikut :

   Confidence=p(B|A)= ∑ transaksi berisi A dan B / ∑ transaksi yang berisi A

2.2.4 Algoritma Apriori

Algoritma apriori adalah algoritma yang digunakan dalam teknik asosiasi. Apriori dirancang untuk beroperasi pada *database* yang mengandung transaksi. Ada dua proses utama dalam algoritma tersebut :

1. *Join step* : C_{k+1} dibangun dengan menggabungkan L_k dengan dirinya sendiri.
2. *Prune step* : setiap *(k-1) itemset* yang bukan *frequent* tidak dapat menjadi subset dari suatu *frequent k-itemset*.

Tabel-tabel dibawah ini merupakan gambaran cara kerja algoritma apriori

Tabel 2.4 Daftar Transaksi

ID transaksi	Daftar belanja
1	Beer, diaper, baby powder, bread, umbrella
2	Diaper, baby powder
3	Beer, diaper, milk
4	Diaper, beer, detergent
5	Beer, milk, coca cola

Langkah 1 : dari tabel 2.4 barang dibatasi dengan minimum support = 40% (2/5).

C1

Tabel 2.5 Tabel C1

Barang	*Support*
Beer	4/5
Diaper	4/5
Baby powder	2/5
~~Bread~~	1/5
~~Umbrella~~	1/5
Milk	2/5
~~Detergent~~	1/5
~~Coca cola~~	1/5

Tabel 2.5 berisi barang dengan *support* yang dimilikinya

L1

Tabel 2.6 Tabel L1

Barang	*Support*
Beer	4/5
Diaper	4/5
Baby powder	2/5
Milk	2/5

Barang yang berada pada tabel C1 (tabel 2.5) yang memiliki *support* kurang dari 40% dihilangkan sedangkan barang yang memiliki *support* lebih dari 40% akan dimasukkan ke dalam tabel L1 (tabel 2.6).

Barang – barang yang ada pada tabel 2.6 kemudian dikombinasikan dan dimasukkan ke dalam tabel C2 (tabel 2.7)

Tabel 2.7 Tabel C2

Barang	*Support*
Beer, diaper	3/5
~~Beer, baby powder~~	1/5
Beer, milk	2/5
Diaper, baby powder	2/5
~~Diaper, milk~~	1/5
~~Baby powder, milk~~	0

Barang pada tabel C2 (tabel 2.7) dihapus dari tabel jika memiliki *support* kurang dari 40%. Jika barang memiliki *support* lebih dari 40% akan dimasukkan ke tabel L2 (tabel 2.8)

Tabel 2.8 Tabel L2

Barang	*Support*
Beer, diaper	3/5
Beer, milk	2/5
Diaper, baby powder	2/5

Barang yang ada pada Tabel 2.8 kemudian dikombinasikan dan dimasukkan ke dalam tabel C3 (tabel 2.9)

Tabel 2.9 Tabel C3

Barang	*Support*
~~Beer, diaper, milk~~	1/5
~~Beer, diaper, baby powder~~	1/5
~~Beer, milk, baby powder~~	0/5
~~Milk, diaper, baby powder~~	0/5

Kombinasi barang yang ada pada tabel C3 tidak memenuhi *minimum support,* sehingga tabel L2 lah yang diberikan *minimum confidence* = 70%.

Tabel 2.10 C2 Dengan Nilai *Confidence*

Barang	*Support* (A, B)	*Confidence*
Beer, diaper	60%	75%
~~Beer, milk~~	40%	50%
~~Diaper, baby powder~~	40%	50%
Diaper, beer	60%	75%
Milk, beer	40%	100%
Baby powder, diaper	40%	100%

Tabel 2.11 berisi *strong rules,* yaitu kombinasi barang yang memenuhi kriteria *minimum support* dan *minimum confidence*

Tabel 2.11 Daftar *Strong Rules*

Strong rules	*Support*	*Confidence*
Beer => diaper	60%	75%
Diaper => beer	60%	75%
Milk => beer	40%	100%
Baby powder => diaper	40%	100%